RCP Plus Reference

RCP Version 3

Specification

BOSCH

Technik fürs Leben

Copyright

This manual is the intellectual property of Bosch Security Systems and is protected by copyright. All rights are reserved. No part of this document may be reproduced or transmitted for any purpose, by whatever means, be they electronic or mechanical, without the express written permission of Bosch Security Systems. Bosch Security Systems 2001 2021

Note

This manual was compiled with the greatest of care and all information double checked. At the time of printing the description was complete and correct. Because of the further development of products, the content of the manual might change without prior notice. Bosch Security Systems will not be liable for damage which is directly or indirectly due to errors, incompleteness, or discrepancies between the manual and the product described.

Trade marks

All names used in this manual for hardware and software are very probably registered trade marks and must be treated as such.

1 Functional Description

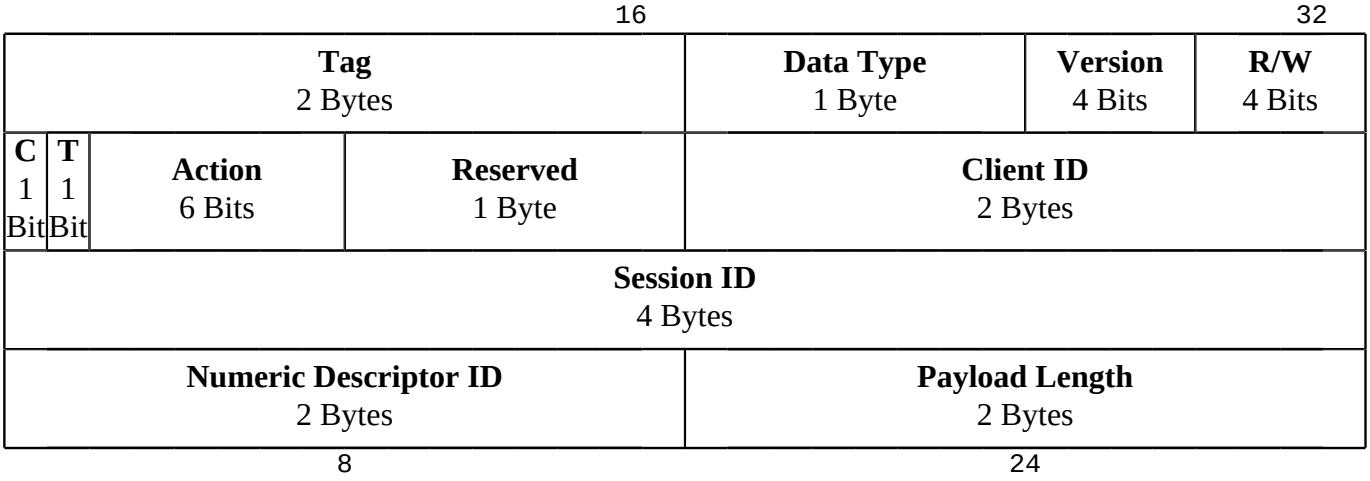
© Copyright 2001-2021 Bosch Security Systems

1.1 Transport Protocol

The transport protocol for this remote control must be TCP. All VideoJets will establish a TCP listen socket on port 1756. Any remote control must be sent to this port. Multiple RCP connections from the same endpoints are allowed. The maximum number of RCP connections on a single endpoint is limited to 50 connections at the same time. Alternatively, a connection can be made using a HTTP tunnel. See this chapter for details.

Remote Control ProtocolPlus Protocol Header Layout

Version 3 (VIP, VideoJet and VipX Series and VIP110Version 6.0 and higher). The RCP Plus protocol header consists of 16 Bytes. The begin of the payload section is now on DWORD boundary.



Tag

Each tag is represented by two octets. It identifies the command which should be processed by the VideoJet.

Data Type

Specifies the data type of the payload section. These are the currently available data types.

Values:

F_FLAG	0x00
T_OCTET	0x01
T_WORD	0x02
T_INT	0x04
T_DWORD	0x08
P_OCTET	0x0C
P_STRING	0x10
P_UNICODE	0x14

Version

The current RCP version is 3. Backward compatibility to version 2 or version 0 is NOT provided.

© Copyright 2001-2021 Bosch Security Systems

R/W

Specifies whether the command should read or write. The Read/Write field is coded in the lower nibble of byte 4.

Values:

Read	0x00
Write	0x01

C

Continuation. This bit signals, when set, that this RCP+ packet is not terminated in the payload; additional packets with the full RCP+ header will follow immediately in the stream as long as this bit is cleared. The reassembly of the complete payload is up to the application and is beyond the scope of this document.

T

StringTable available. This bit signals, when set, that there is a string table appended to this RCP+ packet which contains IPv6 addresses or host names.

Action

Specifies the kind of the packet.

Values:

Request	0x00
Reply	0x01
Message	0x02
Error	0x03

Reserved

This byte is returned by the VideoJet unchanged. It is up to the user to setup a request ID here to assign the replies to multiple pending requests.

Client ID

Each RCP client register results in a Client ID; this ID has to be provided in all following RCP commands.

Session ID

This ID is used for implementations which need to identify a once registered user in other applications or RCP sessions.

Numeric Descriptor ID

The Numeric Descriptor specifies an attribute for components which are installed more than one time inside the VideoJet, e.g. inputs or relays. The first component is always counted as 1. If this field is not applicable to the command in this packet, a value of zero should be inserted.

© Copyright 2001-2021 Bosch Security Systems

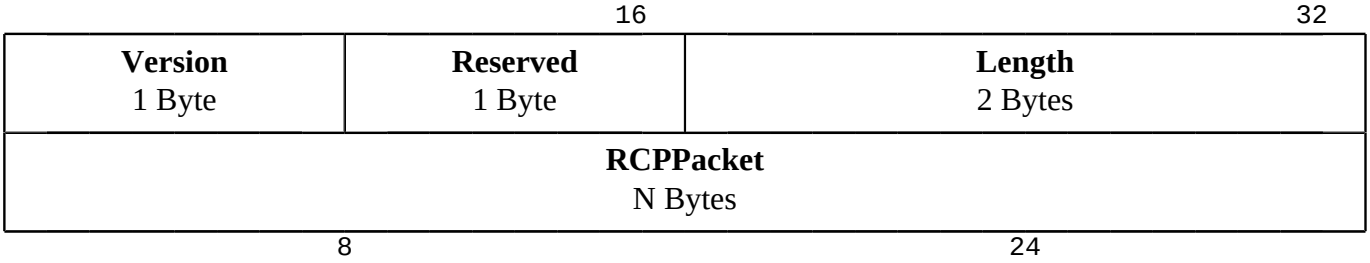
Payload Length

The number of data bytes inside the payload section. The length field itself is not counted.

© Copyright 2001-2021 Bosch Security Systems

1.2 Packetizing in the TCP Stream

As TCP is a stream oriented protocol, packetizing information has to be inserted to achieve proper packet reconstruction. For this purpose the TPKT structure is used.



Version

TPKT version 3.

Reserved

Should be set to zero.

Length

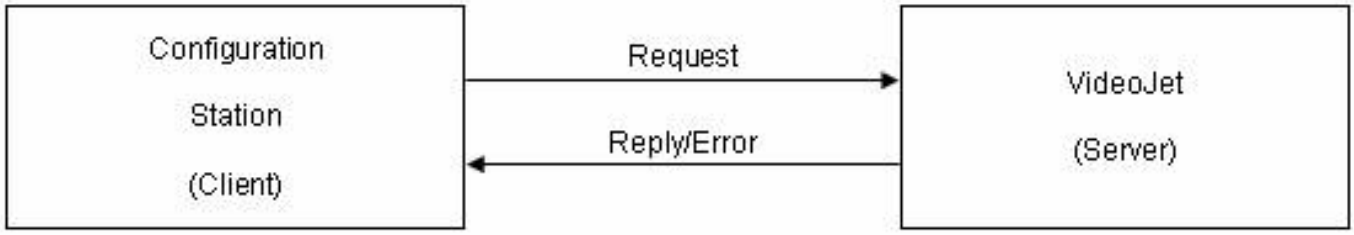
Byte count of the complete packet; RCP packet length + TPKT length (4 Bytes).

© Copyright 2001-2021 Bosch Security Systems

1.3 RCP Protocol Procedure

Request / Reply

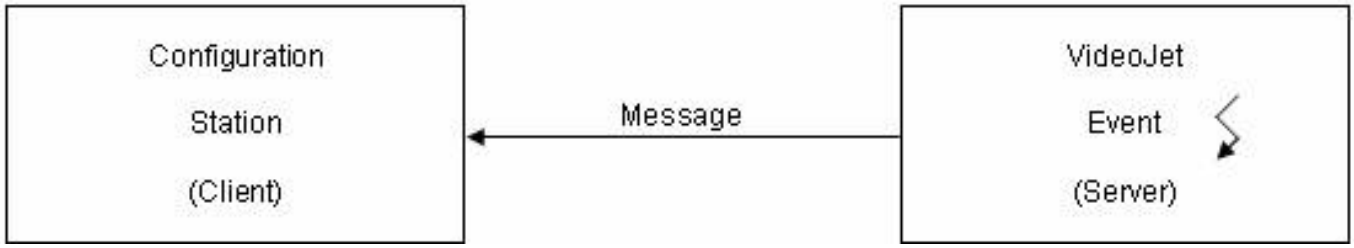
The VideoJet unit will send back a reply on each incoming request. In some protocol functions the reply will carry no data.



Note: A reply from a write command which is not readable will be an RCP_ERROR_READ_NOT_SUPPORTED (error code 0x90) This means that it is not possible to determine correct setting of the command.

Messages

A message will be generated on the VideoJet on certain events (see RCP command for details). If an RCP client is registered to receive a certain message (e.g. an input state has changed), the VideoJet will generate it. The message has the same payload format as the response to a Read command of the same tag, except the reserved byte. The reserved byte is used as sequence number.



RCP is capable of sending messages on certain events. Before an RCP client can receive these messages, a registration at the VideoJet is necessary.

RCP Errors

The packet will have the standard layout with the method field set to ‘Error’. The first bytes of the payload section contains error cause. If the error code is RCP_ERROR_COMMAND_SPECIFIC, then the command specific error (see RCP command for details) is included in the second byte. The following generic error codes are defined:

RCP_ERROR_INVALID_VERSION	0x10
RCP_ERROR_NOT_REGISTERED	0x20
RCP_ERROR_INVALID_CLIENT_ID	0x21
RCP_ERROR_INVALID_METHOD	0x30
RCP_ERROR_INVALID_CMD	0x40
RCP_ERROR_INVALID_ACCESS_TYPE	0x50
RCP_ERROR_INVALID_DATA_TYPE	0x60
RCP_ERROR_WRITE_ERROR	0x70
RCP_ERROR_PACKET_SIZE	0x80
RCP_ERROR_READ_NOT_SUPPORTED	0x90

© Copyright 2001-2021 Bosch Security Systems

RCP_ERROR_INVALID_AUTH_LEVEL	0xa0	
RCP_ERROR_INVAILD_SESSION_ID	0xb0	
RCP_ERROR_TRY_LATER	0xc0	
RCP_ERROR_TIMEOUT	0xd0	
RCP_ERROR_NO_LICENCE	0xe0	Used by NVR only
RCP_ERROR_COMMAND_SPECIFIC	0xf0	
RCP_ERROR_ADDRESS_FORMAT	0xf1	
RCP_ERROR_NOT_SUPPORTED_ON_THAT_PLATFORM	0xf2	
RCP_ERROR_UNKNOWN	0xff	

Note the error code 0xc0 RCP_ERROR_TRY_LATER indicates, that the VideoJet recognizes the command, but it cannot be processed immediately. The client should repeat this command later.

A list of all defined command specific error codes can be found in the Appendix.

© Copyright 2001-2021 Bosch Security Systems

1.4 RCP over a HTTP tunnel

Many network installations includes firewalls to protect network segments against unauthorisized access. In many cases, the HTTP port (80) is enabled by default to gain Internet access. For this purpose, RCP offers a second network port. As the VideoJet has a built in webserver, the connection can be made to the webserver, the token

```
GET /rcp_tunnel HTTP1.0\r\n\r\n
```

advices the webserver to pass this socket to the RCP server. From that point on, the client is connected to the RCP server and can immediately cointinue with RCP registration.

Note: The delimiters '\r' and '\n' are written here in C notation; in the TCP packet to the server these delimiters corresponds with the ASCII codes 0x0d and 0x0a.

Example:

- Establish a TCP connection to port 80 at your Videjet
- Send the string "GET /rcp_tunnel HTTP/1.0\r\n\r\n" (as HEX string: 0x47 0x45 0x54 0x20 0x2f 0x72 0x63 0x70 0x5f 0x74 0x75 0x6e 0x6e 0x65 0x6c 0x20 0x48 0x54 0x54 0x50 0x31 0x2e 0x30 0x0d 0x0a 0x0d 0x0a)
- Proceed in the same manner as a normal RCP connection

Receiving media data

The VideoJet is capable for tunneling media data through HTTP. In some network environments this might be helpful.

When using this mechanism, the connect primitive must set the "RTP over TCP" for the media encapsulation protocol. The response for Media host and port can be ignored. After that, a TCP connection to the HTTP port must be established. By sending the string

```
GET /media_tunnel/SSSSSSSS/CC/DD/LL/CO HTTP/1.0\r\n\r\n
```

SSSSSSS	The session ID returned from the connect primitive (padded with leading zeros to 8)
CC	The media type (see connect primitive; 01=video, 02=audio)
DD	Direction of media (01=receive media, 00=transmit media)
LL	The line input number
CO	The relative coder number

When the HTTP server inside the VideoJet received this string, it passes this socket to the RCP. From that point on, media data will be transferred over this socket using TPKT encapsulation (see chapter Packetizing in the TCP Stream).

The coder numbers that have to be used to establish the media tunnels is a relative one that isn't unique when using video as media type. The video and meta data can have the same relative coder number and have the same media type. In that case the optional index parameter might be used to distinguish the media channels:

```
GET /media_tunnel/SSSSSSSS/CC/DD/LL/CO?index=n HTTP/1.0\r\n\r\n
```

© Copyright 2001-2021 Bosch Security Systems

where n is the number of the initial connect primitive channel position starting at 1.

E.g the ConnectPrimitive request contains a Get Video, a Get Audio and a Get Meta section, then the index for video would be 1, the index for audio 2 and the index for meta would be 3.

When all sockets of a session are in data sending mode, a RCP message MEDIA_SOCKETS_COMPLETE with the associated session ID will be sent out to the initiator of the session. This is useful when a session is used for replay; the replay PLAY command can securely be applied when the MEDIA_SOCKETS_COMPLETE message has been arrived.

© Copyright 2001-2021 Bosch Security Systems

1.5 Autodetecting Devices

Bosch Security Systems products are being detected on a network using an IP broadcast to either the static UDP port 1757 or to the configurable UDP discover port. The default port number for the configurable discover port is 1800, and it can be modified with the RCP command CONF_DISCOVER_PORT (0x0976). In addition to the possibility to send a scan request packet to the broadcast destination address of 255.255.255.255, the discover port of the devices listens for scan requests sent to the configurable multicast group, which can be set by means of the RCP command CONF_AUTODETECT_REPLY_GROUP (0x0956). The default multicast detect group is 225.86.67.83.

Devices generate and transmit their replies to an autodetect request with a random delay of up to 2 seconds. When implementing your own detection software, please bear in mind that, when calculating the necessary waiting period until all replies are received, additionally to the possible span of max 2 seconds, you need to consider the time necessary to process the reply as well as the time the packet spends on the wire and in propagation devices like switches or routers on its way to and back from the device to be detected.

The scanned devices send their replies either to the source IP address of the request, or to the broadcast address 255.255.255.255. In case the source IP address of the request resides inside the same subnet, that is defined by the scanned device's IP address and it's subnet mask, the reply is addressed to the source IP address in the request. If the originator is not located in the same subnet as the scanned device, the reply is addressed to the broadcast address 255.255.255.255. In case of a scan request sent to the device via the multicast scan group, the reply is addressed to the same multicast group.

An autodetect scan request packet must consist of an autodetect request header, that may or may not be preceeded by an RCP plus protocol header for command id CONF_RCP_AUTODETECT (0xffdc). In case of an existing RCP plus protocol header in the scan request, the length field of the RCP plus protocol header must contain the length of the autodetect request header.

The format for the autodetect request header as well as for the autodetect reply is shown below:

Autodetect Request Header

16 32

0x99 1 Byte	**0x39** 1 Byte	**0xA4** 1 Byte	**0x27** 1 Byte
Sequence Number 4 Bytes			
0x00 1 Byte	**0x00** 1 Byte	**Reply Port** 2 Bytes	

8 24

Sequence Number

This sequence number must be a randomized number.

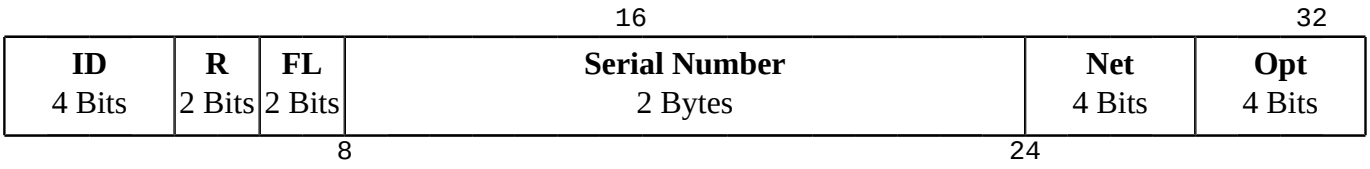
Reply Port

Specifies the reply UDP port e.g. RP1 0x06; RP2 0xDE -> reply port: 1758

© Copyright 2001-2021 Bosch Security Systems

Additional information on hardware versions

32-Bit Hardware version Former Scheme



ID

Values:

NCID_NVR	0x01	
NCID_VJ8000	0x02	
NCID_VIP10	0x03	
NCID_VIP1000	0x04	
NCID_VJ400	0x06	
NCID_VIP100	0x07	
NCID_VJEX	0x08	
NCID_VJ1000	0x09	
NCID_VJ100	0x0A	
NCID_VJ10	0x0B	
NCID_VJ8008	0x0C	
NCID_VJ8004	0x0D	Latest unit carrying the old numbering scheme
EscapeCode	0x0F	Used to switch over to new detection

R

Reserved

FL

Flavour - Values:

HARDWARE_TYPE_VIN	0x00
HARDWARE_TYPE_VOUT	0x01
HARDWARE_TYPE_VIN_OUT	0x02

Net

Values:

	Mask	Name
Bit 2	0x4	HARDWARE_SYSTEM_ETHERNET
Bit 0	0x1	HARDWARE_SYSTEM_OPT_ISDN

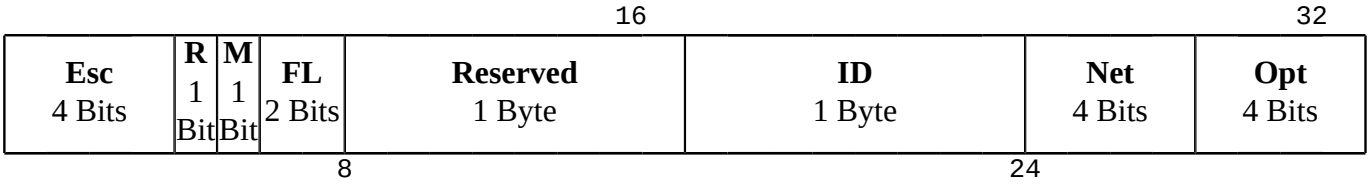
© Copyright 2001-2021 Bosch Security Systems

Opt

Values:

	Mask	Name
Bit 1	0x2	HARDWARE_OPT_HDD
Bit 0	0x1	HARDWARE_OPT_AUDIO

32-Bit Hardware version New Scheme



Esc

Must be set to 0xF

R

Reserved

M

Values:

Master	0x00
Slave	0x01

FL

Flavour - Values:

HARDWARE_TYPE_VIN	0x00
HARDWARE_TYPE_VOUT	0x01
HARDWARE_TYPE_VIN_OUT	0x02

ID

Note: this Hardware ID will not be further maintained for future products. To distinguish different device variants please use the Product- and Variant ID, see command CONF_DEVICE_TYPE_IDS

HARDWARE_ID_VIPX1	0x01
HARDWARE_ID_VIPX2	0x02
HARDWARE_ID_VIPXDEC	0x03
HARDWARE_ID_VJ_X10	0x05
HARDWARE_ID_VJ_X20	0x06

© Copyright 2001-2021 Bosch Security Systems

HARDWARE_ID_VJ_X40	0x07
HARDWARE_ID_VJ_X40_ECO	0x08
HARDWARE_ID_VJ_X10_ECO	0x09
HARDWARE_ID_VJ_X20_ECO	0x0A
HARDWARE_ID_IP_PANEL	0x0d
HARDWARE_ID_GEN4	0x0e
HARDWARE_ID_M1600	0x0f
HARDWARE_ID_FLEXIDOME	0x11
HARDWARE_ID_M1600_DEC	0x13
HARDWARE_ID_M1600_XFM	0x15
HARDWARE_ID_AUTODOME	0x16
HARDWARE_ID_NBC_225P	0x1a
HARDWARE_ID_VIPX1_XF	0x1e
HARDWARE_ID_CPP13	0x1f
HARDWARE_ID_NBC_255P	0x20
HARDWARE_ID_NBC_255W	0x21
HARDWARE_ID_AUTODOME_EASY_II	0x23
HARDWARE_ID_AUTODOME_EASY_II_E	0x24
HARDWARE_ID_VIPX1_XF_E	0x25
HARDWARE_ID_VJT_X20XF_E_2CH	0x26
HARDWARE_ID_VJT_X20XF_E_4CH	0x27
HARDWARE_ID_VIPX1_XF_W	0x28
HARDWARE_ID_VG5_AUTODOME_700	0x29
HARDWARE_ID_NDC_455_P	0x2a
HARDWARE_ID_NDC_455_P_IVA	0x2b
HARDWARE_ID_NBC_455_P	0x2c
HARDWARE_ID_NBC_455_P_IVA	0x2d
HARDWARE_ID_VG4_AUTODOME	0x2e
HARDWARE_ID_NDC_225_P	0x2f
HARDWARE_ID_NDC_255_P	0x30
HARDWARE_ID_VOT	0x32
HARDWARE_ID_NDC_274_P	0x35
HARDWARE_ID_NDC_284_P	0x36
HARDWARE_ID_NTC_265_PI	0x37
HARDWARE_ID_NDC_265_PIO	0x38
HARDWARE_ID_DINION_720P	0x39
HARDWARE_ID_NDN_822	0x3a
HARDWARE_ID_FLEXIDOME_720P	0x3b
HARDWARE_ID_NDC_265_W	0x3c
HARDWARE_ID_NDC_265_P	0x3d
HARDWARE_ID_NBC_265_P	0x3e
HARDWARE_ID_NDC_225_PI	0x3f
HARDWARE_ID_NTC_255_PI	0x40
HARDWARE_ID_JR_DOME_HD	0x41
HARDWARE_ID_JR_DOME_HD_FIXED	0x42
HARDWARE_ID_EX30_IR	0x43
HARDWARE_ID_GEN5_HD_PC	0x45
HARDWARE_ID_EX65	0x46

© Copyright 2001-2021 Bosch Security Systems

HARDWARE_ID_DINION_1080P	0x47
HARDWARE_ID_FLEXIDOME_1080P	0x48
HARDWARE_ID_HD_DECODER	0x49
HARDWARE_ID_GEN5_HD	0x4a
HARDWARE_ID_NER_L2	0x4b
HARDWARE_ID_VIP_MIC	0x4c
HARDWARE_ID_GEN5_A5_800	0x4d
HARDWARE_ID_NEVADA_TRANSCODER	0x4e
HARDWARE_ID_TESLA_BOXED	0x4f
HARDWARE_ID_TESLA_DOME	0x50
HARDWARE_ID_GEN5_A5_700	0x52
HARDWARE_ID_VJ_GENERIC_TRANSCODER	0x53
HARDWARE_ID_HD_DECODER_M	0x54
HARDWARE_ID_OASIS	0x55
HARDWARE_ID_GALILEO_BOXED	0x56
HARDWARE_ID_GALILEO_DOME	0x57
HARDWARE_ID_HUYGENS_KEPPLER	0x58
HARDWARE_ID_TESLA_KEPPLER	0x59
HARDWARE_ID_GALILEO_KEPPLER	0x5a
HARDWARE_ID_NUC_20002	0x5b
HARDWARE_ID_NUC_20012	0x5c
HARDWARE_ID_NUC_50022	0x5d
HARDWARE_ID_NUC_50051	0x5e
HARDWARE_ID_NPC_20002	0x5f
HARDWARE_ID_NPC_20012	0x60
HARDWARE_ID_NPC_20012_W	0x61
HARDWARE_ID_NIN_50022	0x62
HARDWARE_ID_NII_50022	0x63
HARDWARE_ID_NDN_50022	0x64
HARDWARE_ID_NDI_50022	0x65
HARDWARE_ID_NTI_50022	0x66
HARDWARE_ID_NIN_50051	0x67
HARDWARE_ID_NDN_50051	0x68
HARDWARE_ID_NAI_90022	0x69
HARDWARE_ID_NCN_90022	0x6a
HARDWARE_ID_NEVADA_DECODER	0x6c
HARDWARE_ID_NBN_50022_C	0x6d
HARDWARE_ID_NBN_40012_C	0x6e
HARDWARE_ID_NBN_80052	0x71
HARDWARE_ID_MIC_7000	0x72
HARDWARE_ID_EX65_HD	0x73
HARDWARE_ID_NBN_80122	0x74
HARDWARE_ID_NPC_20012_L	0x75
HARDWARE_ID_MIC_NPS	0x76
HARDWARE_ID_NBN_50051_C	0x78
HARDWARE_ID_NIN_40012	0x79
HARDWARE_ID_NII_40012	0x7a
HARDWARE_ID_NDN_40012	0x7b

© Copyright 2001-2021 Bosch Security Systems

HARDWARE_ID_NDI_40012	0x7c
HARDWARE_ID_NTI_40012	0x7d
HARDWARE_ID_NII_50051	0x7e
HARDWARE_ID_NDI_50051	0x7f
HARDWARE_ID_NEZ_4000	0x80
HARDWARE_ID_VEGA_3000_HD	0x81
HARDWARE_ID_VEGA_4000_HD	0x82
HARDWARE_ID_VEGA_5000_HD	0x83
HARDWARE_ID_VEGA_5000_MP	0x84
HARDWARE_ID_NIN_70122_180	0x85
HARDWARE_ID_NIN_70122_360	0x86
HARDWARE_ID_NEZ_5000	0x88
HARDWARE_ID_NEZ_5000_IR	0x89
HARDWARE_ID_ROLA	0x8b
HARDWARE_ID_NBN_80122_CA	0x8c
HARDWARE_ID_ROLA_VANDAL	0x8d
HARDWARE_ID_ROLA_MSI	0x8e
HARDWARE_ID_SVO_1601	0x8f
HARDWARE_ID_VJD_8000	0x90
HARDWARE_ID_PIXIE_BOXED	0x91
HARDWARE_ID_SVI_1609	0x92
HARDWARE_ID_PIXIE_DOME	0x94
HARDWARE_ID_NBN_80052_FIRE	0x95
HARDWARE_ID_ALTAIR3	0x96
HARDWARE_ID_PLANCK	0x97
HARDWARE_ID_NDP_4000	0x99
HARDWARE_ID_NDP_5000	0x9a
HARDWARE_ID_NDP_5000_IR	0x9b
HARDWARE_ID_NDE_8502	0x9e
HARDWARE_ID_NDE_8503_4	0x9f
HARDWARE_ID_NBE_4000	0xa0
HARDWARE_ID_NBE_5000	0xa1
HARDWARE_ID_NBE_6000	0xa2
HARDWARE_ID_MIC_9000	0xa3
HARDWARE_ID_MIC_7000_7_3	0xa9
HARDWARE_ID_GEN5_7_3	0xaa
HARDWARE_ID_NTI_51022	0xad
HARDWARE_ID_NDP_5523	0xae
HARDWARE_ID_NDP_5523_IR	0xaf
HARDWARE_ID_CHOPPER	0xb0
HARDWARE_ID_NDE_8502_RX_8503_RX	0xb4
HARDWARE_ID_ISCSI_TARGET	0xf6
HARDWARE_ID_VRM_PROXY_16	0xf7
HARDWARE_ID_VRM_UL_APP	0xf8
HARDWARE_ID_VRM_LE_APP	0xf9
HARDWARE_ID_CAMNETWORK	0xfa
HARDWARE_ID_VRM_PROXY	0xfb

© Copyright 2001-2021 Bosch Security Systems

HARDWARE_ID_VRM	0xfc
HARDWARE_ID_VIDOS_SERVER	0xfd
HARDWARE_ID_VIDOS_MONITOR	0xfe

Net

Values:

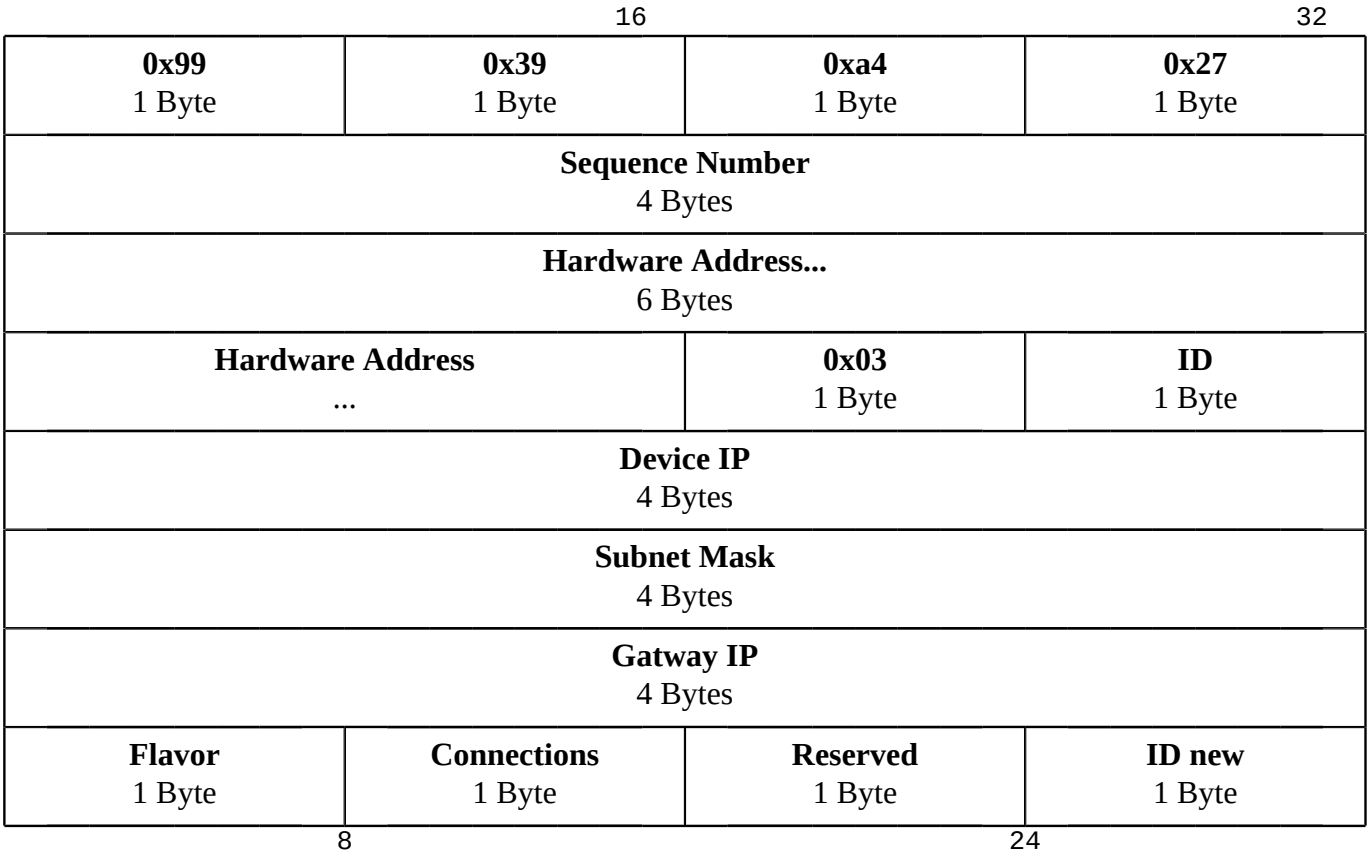
	Mask	Name
Bit 2	0x4	HARDWARE_SYSTEM_ETHERNET
Bit 0	0x1	HARDWARE_SYSTEM_OPT_ISDN

Opt

Values:

	Mask	Name
Bit 1	0x2	HARDWARE_OPT_HDD
Bit 0	0x1	HARDWARE_OPT_AUDIO

1st Reply Packet



Sequence Number

Returns the sequence number from the request packet.

© Copyright 2001-2021 Bosch Security Systems

Hardware Address

MAC address of the device.

ID

Unit Hardware ID returned from request (will be 0xff when the new version layout is used).

Device IP

Current IP address of the device.

Subnet Mask

Current Subnet mask of the device.

Gatway IP

Current Gateway of the device.

Flavor

Flavor of this device.

Connections

Number of active connections.

ID new

The ID filed of the new version layout.

2nd Reply Packet

```
<?xml version="1.0" encoding="UTF-8"?>
<root xmlns="OEMExtXX:device-1-0">
<device>
 <deviceType>OEMExtXX:device:deviceIDYY</deviceType>
 <friendlyName>VideoJet</friendlyName>
 <unitName>UnitName</unitName>
 <deviceGUID>XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
deviceGUID>
 <serialNumber>XXXXXXXX:YYYYYYYY</serialNumber>
 <physAddress>XX-XX-XX-XX-XX-XX</physAddress>
 <sessionKey>XXXXXXXXXXXXXXXX</sessionKey>
 <unitIPAddress>XXX.XXX.XXX.XXX</unitIPAddress>
 <unitSubnetMask>XXX.XXX.XXX.XXX</unitSubnetMask>
 <unitGatewayIp>XXX.XXX.XXX.XXX</unitGatewayIp>
 <unitIPv6Address>XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX</
unitIPv6Address>
 <activeConnections>XX</activeConnections>
```

© Copyright 2001-2021 Bosch Security Systems

```
 <RCPPort>XXXX</RCPPort>
 <HTTPPort>XXXX</HTTPPort>
 <HTTPSPort>XXXX</HTTPSPort>
 <dhcp>X</dhcp>
 <cameraFrontendID>XXXX</cameraFrontendID>
 <Flags>X</Flags>
 <Lines>X</Lines>
 <BcJpeg>X</BcJpeg>
 <ClusterMember>XXX.XXX.XXX.XXX</ClusterMember>
 <ClusterMemberMAC>XX-XX-XX-XX-XX-XX</ClusterMemberMAC>
 <ClusterID>X</ClusterID>
 <deviceCapabilities>XXXXXXXXXX...XX</deviceCapabilities>
</device>
</root>

<root xmlns="OEMExtXX:device-1-0">
XX is the OEM Extension; 0x00 for Bosch Security Systems; OEM
 specific

<deviceType>OEMExtXX:device:deviceIDYY</deviceType>
XX is the OEM Extension; 0x00 for Bosch Security Systems; OEM
 specific
YY is the VideoJet ID from the request packet

<friendlyName>VideoJet</friendlyName>
The name and type of the VideoJet; type and OEM specific

<unitName>UnitName</unitName>
The name of the unit coded in UTF-8

<serialNumber>XXXXXXXX:YYYYYYYY</serialNumber>
XXXXXXX is the Hardware version of the VideoJet
YYYYYYY is the Softwareware version of the VideoJet

<physAddress> XX-XX-XX-XX-XX-XX </physAddress>
XX-XX-XX-XX-XX-XX is the MAC address of the VideoJet

<sessionKey>XXXXXXXXXXXXXXXX</sessionKey>
No common use

<unitIPAddress> XXX.XXX.XXX.XXX </unitIPAddress>
XXX.XXX.XXX.XXX is the current IP address of the VideoJet

<unitSubnetMask>XXX.XXX.XXX.XXX</unitSubnetMask>
XXX.XXX.XXX.XXX is the subnet mask of the VideoJet

<unitGatewayIp>XXX.XXX.XXX.XXX</unitGatewayIp>
XXX.XXX.XXX.XXX is the current gateway IP address of the VideoJet
```

© Copyright 2001-2021 Bosch Security Systems

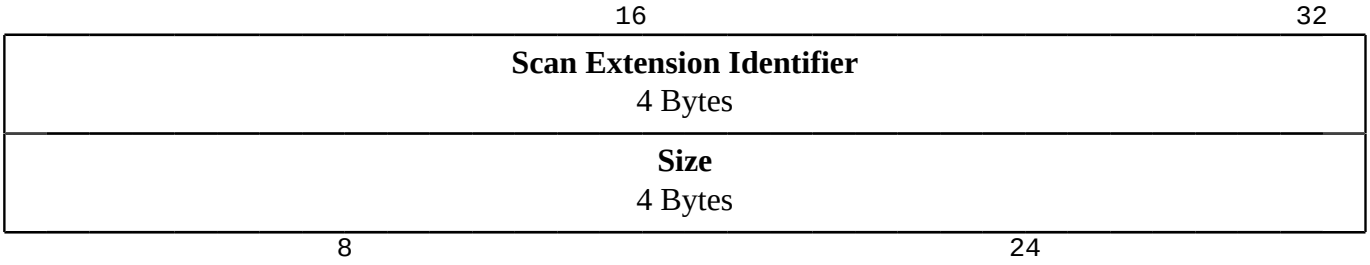
```
<unitIPv6Address>XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX</
unitIPv6Address>
XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX is the current IPv6 address of
 the VideoJet e.g. FD00::1234:5678

<activeConnections>XX</activeConnections>
XX is the number of active media connections of the VideoJet

<RCPPort>XXXX</RCPPort>
XXXX is the current RCP port number the VideoJet is listening to

<HTTPPort>XXXX</HTTPPort>
XXXX is the current HTTP port number the VideoJet is listening to

<HTTPSPort>XXXX</HTTPSPort>
XXXX is the current HTTPS port number the VideoJet is listening to

<dhcp>X</dhcp>
X indicates if dhcp is switched on (1) or off (0)

<cameraFrontendID>XXXX</cameraFrontendID>
XXXX is the ID of the (internal) camera frontend (see also Bicom
 documentation)

<Flags>X</Flags>
X is the flag value returned by RCP command CONF_CM_DEFAULT_CONFIG

<Lines>X</Lines>
X is the number of video lines the device has

<BcJpeg>X</BcJpeg>
X indicates the ability to deliver JPEGs via the broadcast JPEG
 retrieval mechanism

<ClusterMember>XXX.XXX.XXX.XXX</ClusterMember>
XXX.XXX.XXX.XXX is the ip address of one of the cluster members; there
 might be multiple entries including this device

<ClusterMemberMAC>XX-XX-XX-XX-XX-XX</ClusterMemberMAC>
XX-XX-XX-XX-XX-XX is the MAC address of one of the cluster members;
 there might be multiple entries including this device

<ClusterID>XX</ClusterID>
XX is index number of the above ClusterMember list of this device

<deviceCapabilities>XXXXXXXXXX...XX</deviceCapabilities>
```

© Copyright 2001-2021 Bosch Security Systems

```
XXXXXXXXXX...XX is the binary payload of the reply to the command
 CONF_DEVICE_CAPABILITIES
(each byte is printed with %2x) and can help to identify the
 availability of features or the capabilities of this device.
The number of bytes given in this section can vary depending on the
 device and the firmware version.
```

Optional Extension: Targeting Autodetect Scans at devices matching certain criteria

The autodetect process can be targeted at a certain group of devices identified by a defined criteria. This sort of autodetect is device-wise only handled, when sent to the configurable UDP port, whose default port number is 1800. It is not possible via the static autodetect port 1757. To perform a targeted autodetect scan, an extension needs to be added to the usual autodetect scan command. For these targeted kinds of autodetect scans, it is obligatory that the scan packet consists of an RCP plus protocol header, the autodetect request header and the extension itself. The bytes of this extension are considered in the length field of a preceeding RCP header.



Scan Extension Identifier

The scan extension identifier determines the criteria that needs to be matched by the scanned devices in order to reply to the request.

Currently the following possible criteria are defined:	
MAC address	0x01

Size

The Size field of the extension informs about the number of bytes used in the extension. The number includes the 4 bytes of the Scan Extension Identifier field, the 4 bytes of the Size field itself and the length of the following payload in bytes.

In case of the MAC address extension, the Size field is followed by one or more MAC addresses of 6 bytes length each. A scan extension limiting the autodetect scan to 2 devices, the size field of the scan extension should contain a value of 20. (4 bytes Scan Extension Identifier + 4 bytes Length + 6 bytes first MAC address + 6 bytes second MAC address)

Retrieving thumbnail JPEGs using the Autodetecting Broadcast Mechanism

The Bosch Security Systems products support a mechanism that allows to retrieve thumbnail JPEGs via a broadcast autodetect mechanism. The mechanism requires to send a UDP packet similar to the autodetect

© Copyright 2001-2021 Bosch Security Systems

scan packet to the configurable autodetect port of the device, whose default port number is 1800. The device replies with a packet containing the thumbnail JPEG. The JPEG retrieval can not be accomplished via the static autodetect port 1757.

Format of the request packet of an autodetect based JPEG request

The request packet triggering the autodetect based JPEG retrieval consists of an RCP plus protocol header and an autodetect based JPEG request header, followed by a MAC list scan extension.

RCP plus protocol header

The RCP plus protocol header's command id field is set to 0x099e (CONF_JPEG). The data type field is set to 0x0c (P_OCTET). The numeric descriptor field identifies the line of the targeted device, from which the JPEG is to be retrieved. The length field of the request indicates the number of payload bytes and therefore is to be set to the length of the autodetect based JPEG request header plus the length of the scan extension (e.g. 12 bytes for autodetect based JPEG request header, 8 bytes for scan extension header and 6 bytes for MAC address = 26 bytes).

Autodetect based JPEG Request Header

The first four bytes of the autodetect based JPEG request header consist of a static magic number, that is always set to the values given in the table below. The sequence number given in the request is returned in the reply, so that replies can be matched to the respective request.

		16	32
0x4B 1 Byte	**0xA5** 1 Byte	**0xC3** 1 Byte	**0x55** 1 Byte
Sequence Number 4 Bytes			
0x00 1 Byte	**0x00** 1 Byte	**Reply Port** 2 Bytes	
8		24	

Sequence Number

This sequence number must be a randomized number.

Reply Port

Specifies the reply UDP port e.g. RP1 0x06; RP2 0xDE -> reply port: 1758

MAC list scan extension

The autodetect based JPEG request can only be targeted at selected devices. To announce the scope of the request, extending the request packet by a MAC list scan extension, like it is defined in the chapter "Autodetecting Devices" of this document, is obligatory. As the intended use of this JPEG retrieval mechanism is limited to situations where the regular JPEG retrieval mechanisms (HTTP request for the JPEG, RCP request of CONF_JPEG within a regular RCP+ client session, etc.) do not work, the MAC list of the scan extension is limited to 1 MAC address, i.e. only one JPEG from one device can be retrieved at a time.

Format of the reply packet to an autodetect based JPEG request

© Copyright 2001-2021 Bosch Security Systems

The reply packet to the request for an autodetect based broadcast JPEG consists of an RCP plus protocol header and an autodetect based JPEG request header, followed by the JPEG data.

RCP plus protocol header

The RCP plus protocol header's command id field is set to 0x099e (CONF_JPEG). The data type field is set to 0x0c (P_OCTET). The numeric descriptor field identifies the line of the targeted device, from which the JPEG was retrieved. The length field of the reply indicates the number of payload bytes and therefore is to be set to the length of the autodetect based JPEG request header plus the length of the JPEG thumbnail.

Autodetect based JPEG Request Header

The autodetect based JPEG request header in the reply is byte for byte a copy of the same header in the respective request.

JPEG thumbnail

The JPEG thumbnail data follows the autodetect based JPEG request header.

© Copyright 2001-2021 Bosch Security Systems

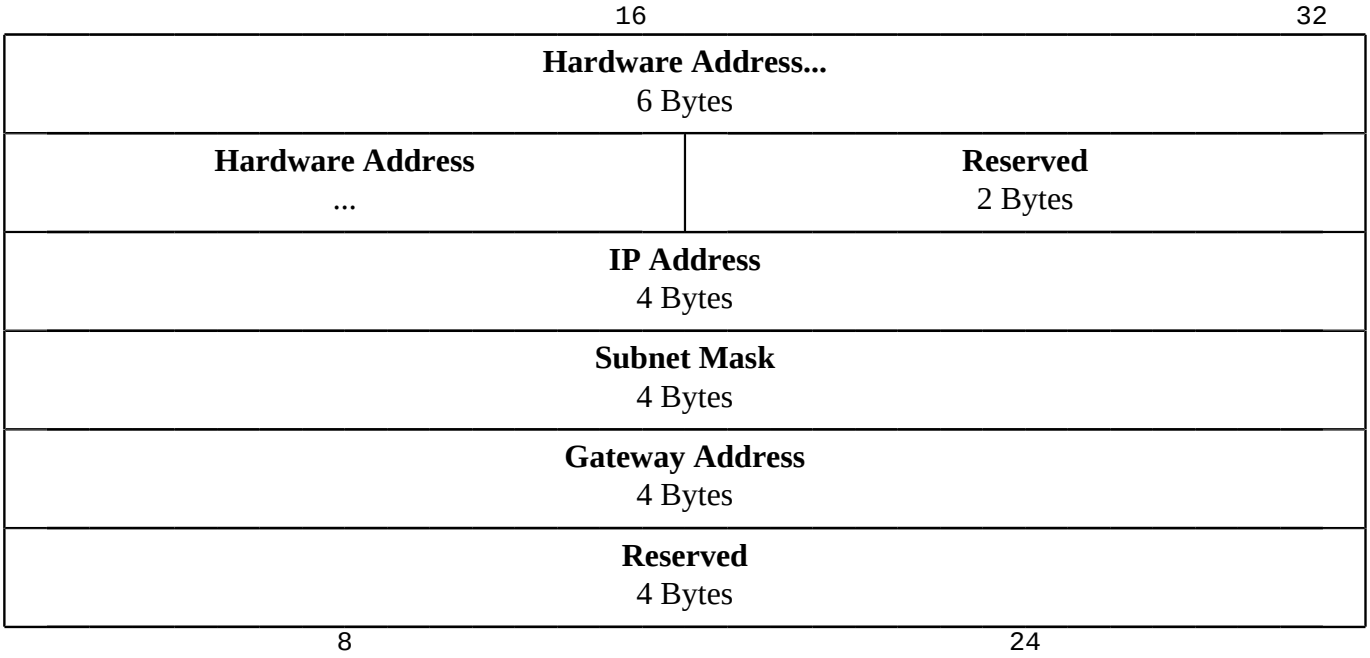
1.6 Setting an IP Address using Broadcast Mechanism

If the current IP address of a device doesn’t match the subnet settings, this unit cannot be reached using the standard Remote Control Protocol. All IP parameters of a VideoJet can be set using a UDP broadcast message to port 1759.

This is supported for IPv4 broadcast 255.255.255.255

And also for IPv6 multicast to (link-scope all-hosts multicast) ff02::1

Payload Structure (Basic Version IPv4 only)



Hardware Address

MAC address of the device.

Reserved

Should be zero

IP Address

IP address to set.

Subnet Mask

Subnet mask to set.

Gateway Address

Gateway IP address to set.

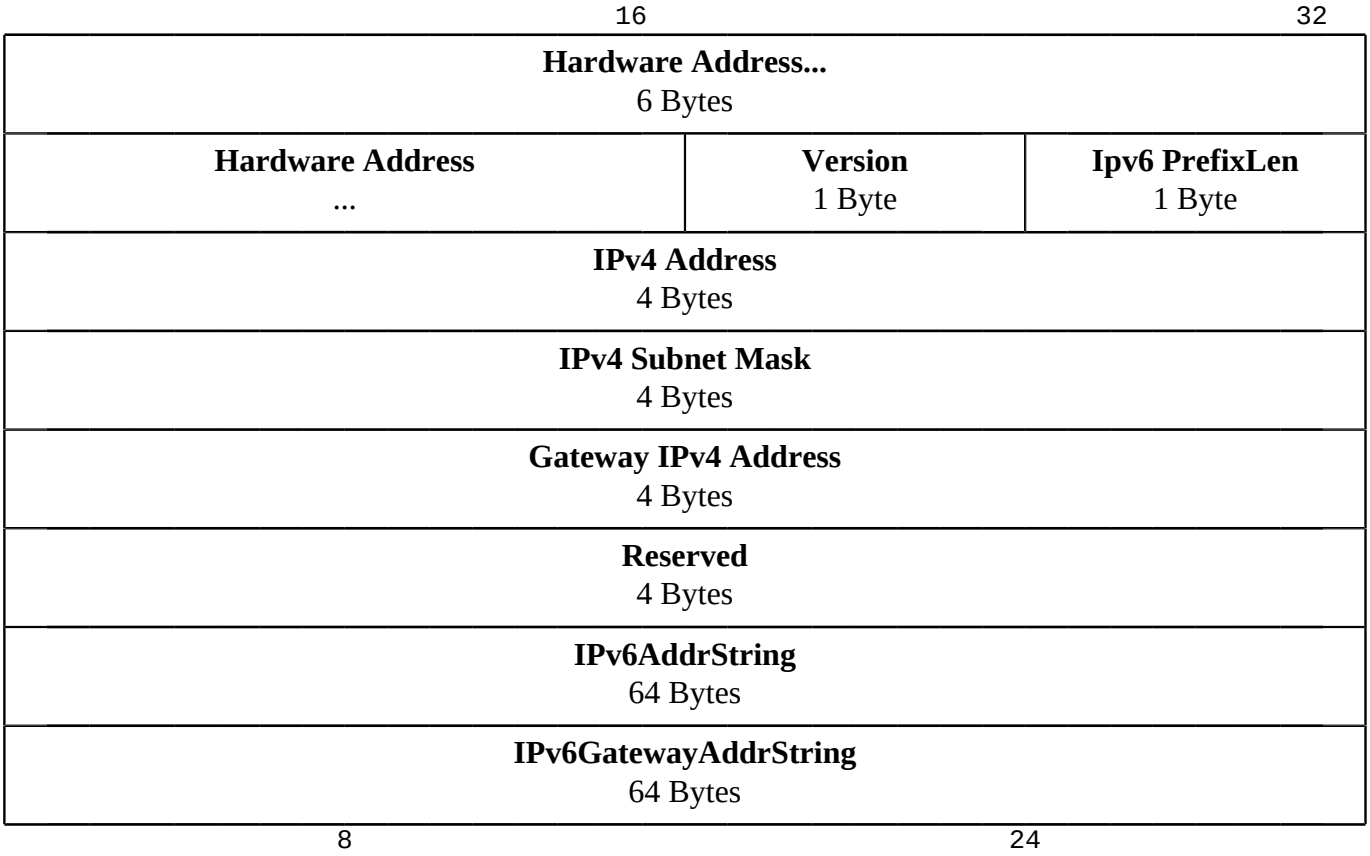
© Copyright 2001-2021 Bosch Security Systems

Note: As all VideoJets will receive this message, only the unit matching the hardware address (HW1-HW6) will update its IP parameters.

Note: There will be no reply to this command.

Note: This command will only work if NO service password is set.

Payload Structure (Extended Version IPv4 and IPv6 settings)



Hardware Address

MAC address of the device.

Version

Version of this command (must be 2)

Ipv6 PrefixLen

IPv6 Prefix length used for the IPv6 Device Address

IPv4 Address

IP address to set.

IPv4 Subnet Mask

Subnet mask to set.

© Copyright 2001-2021 Bosch Security Systems

Gateway IPv4 Address

Gateway IP address to set.

IPv6AddrString

Zero terminated string containig the IPv6 address to set. Format: e.g. FD00::1234:5678

IPv6GatewayAddrString

Zero terminated string containig the IPv6 gateway address to set. Format: e.g. FD00::1234:5678

Note: As all VideoJets will receive this message, only the unit matching the hardware address (HW1-HW6) will update its IP parameters.

Note: There will be no reply to this command.

Note: This command will only work if NO service password is set.

© Copyright 2001-2021 Bosch Security Systems

1.7 RCP over CGI

RCP+ command my also be encapsulated in CGI (Common Gateway Interface) using a standard web browsers URL. All needed parameters are passed to the HTTP server of the video server which tunnels the RCP+ commands to the RCP server. The reply returned is a text based XML structure.

The CGI interface may also be used by a HTTP client written from the scratch based on RFC 1945 (HTTP 1.0) using 'GET' method. The unit's HTTP server will only examine 'Connection: Keep-Alive' and optional 'Authorization: ' header fields. See RFC for implementing details.

Valid CGI parameters:

Command

The RCP+ command tag number

Data Type

Values:

See 'Data Types'

Direction

Values:

READ	0
WRITE	1

Num

The numeric parameter.

Payload

Values:

- The payload as readable string for payload type P_STRING
- The payload as octet array with no spaces preceeded with 0x for payload type P_OCTET and P_UNICODE
- The payload value in hex or decimal notion for all other payload types

Optional CGI parameters:

idstring

User defined parameter which will not be processed by the unit; will return unchanged in the reply

sessionid

Context specific session ID for this command

Multiple parameters are passed in CGI manner with '&' in between two parameter-value pairs.

© Copyright 2001-2021 Bosch Security Systems

Example:

A sample request to readout the units hardware version will look

```
http://ip.ip.ip.ip/rcp.xml?command=0x002e&type=P_STRING&direction=READ
```

The reply XML structure will then have this format:

```
<rcp>
 <command>
   <hex>0x002e</hex>
   <dec>46</dec>
 </command>
 <type>T_DWORD</type>
 <direction>READ</direction>
 <num>0</num>
 <idstring />
 <payload />
 <result>
   <hex>0xf0000f43</hex>
   <dec>4026535747</dec>
 </result>
</rcp>
```

The upper part contains the request, the <result> contains the reply. Numerical values are presented in <hex> and <dec>, strings, unicodes and p_octet in <str> sections. <

Message handling

RCP+ messages will be processed and received by the CGI client using a poll mechanism. A CGI command set with the requested messages command tag numbers need to be issued. After issuing a request, the reply returns immediately after a message has been sent or the default timeout of 1000 ms has been expired. When the timeout expires, a message count of 0 is returned. Otherwise the number of received messages is signaled. The default timeout can be altered by setting the 'collectms' CGI value to the appropriate number of milliseconds. In order to ensure correct assignment of messages in case several clients are polling for messages or if a polling client uses different socket connections, a unique message domain ID (CGI parameter 'msgdomainID') should be provided. Messages are then collected separately for each ID and can be uniquely assigned to the polling client even if requesting via different socket connections.

Valid CGI commands to control messages:

Message

One or a list of requested message RCP+ command tag numbers separated by '$'

CollectMS

Time in milliseconds to collect messages before returning with 'No messages'; default is 1000 ms

© Copyright 2001-2021 Bosch Security Systems

MsgDomainID

Unique ID (maximum 4 Bytes) to identify a certain polling client and to e.g. ensure correct message assignment to clients sending requests via different socket connections

The <msgcnt> section contains the number of veiled <msg> sections inside the reply.

A buffer mechanism (depth 64) ensures that no messages will be lost during two consecutive poll cycles. The <over> sections counts the number of lost messages. <clip> signals too small internal buffers; this should never occur.

The <poll> section counts the number of issued poll requests

A sample request to receive all connection related messages (CONF_CONNECT_TO) will look like

```
http://ip.ip.ip.ip/rcp.xml?message=0xffcc&collectms=5000
```

The reply XML structure will then have this format:

```
<message_list>
 <stats>
   <msgcnt>2</msgcnt>
   <over>0</over>
   <clip>0</clip>
   <poll>1</poll>
 </stats>
 <msg>
   <no>1</no>
   <command>0xffcc</command>
   <num>0</num>
   <sessionID>0x7063001f</sessionID>
   <hex>0xa00a00340000000001010000001010001</hex>
 </msg>
 <msg>
   <no>2</no>
   <command>0xffcc</command>
   <num>0</num>
   <sessionID>0x70630020</sessionID>
   <hex>0xa00a00340000000001010000001010001</hex>
 </msg>
</message_list>
```

Authentication

The RCP+ server needs proper authentication to process protected commands. The CGI interface provides two basic authentication schemes to be used.

In this case, HTTP header authentication (basic or digest) must be present in the request. The internal HTTP server will pass the granted authorization level to the RCP+ server.

For each successful HTTP authentication (for all available resources like html pages, images...) a session cookie will be returned by the HTTP server. When this cookie is present in any further HTTP connections,

© Copyright 2001-2021 Bosch Security Systems

the same authorization level will be granted as for the originating connection. The session cookie will remain active as long as at least one HTTP connection remains open. In this case the internal HTTP server will pass the granted authorization level to the RCP+ server.

IPv6

For RCP commands which are defined as T_DWORD or P_OCTET and contain DWORD IP-Addresses there is a extention to read or write also IPv6 adresses or Hostnames, e.g. URL:

```
http://ip.ip.ip.ip/rcp.xml?
command=0x0a1f&type=T_DWORD&direction=WRITE&payload=0x00000000&ip-
item0=fd00::55:77
```

or

```
http://ip.ip.ip.ip/rcp.xml?
command=0x0a1f&type=T_DWORD&direction=WRITE&payload=0x00000000&ip-
item0=MyHostname.de
```

where payload must contain a 4 byte index to the trailing list of ip-item0..ip-itemn

The reply XML structure will then have this format:

```
<rcp>
 <command>
   <hex>0x0a1f</hex>
   <dec>2591</dec>
 </command>
 <type>T_DWORD</type>
 <direction>READ</direction>
 <num>0</num>
 <idstring />
 <payload />
  <result>
   <hex>0x00000000</hex>
   <dec>0</dec>
   <ip-table>
    <item>
     <idx>0</idx>
     <type>IPV6</type>
     <str>fd00::55:77</str>
    </item>
   </ip-table>
 </result>
</rcp>
```

where payload contains a 4 byte index to the ip-table list for each address contained in the command.

If the command datatype is P_OCTET and the command does contain more IP-Adresses then the result will contain multipe item blocks.

Alternative formats for the item blocks are also possible:

© Copyright 2001-2021 Bosch Security Systems

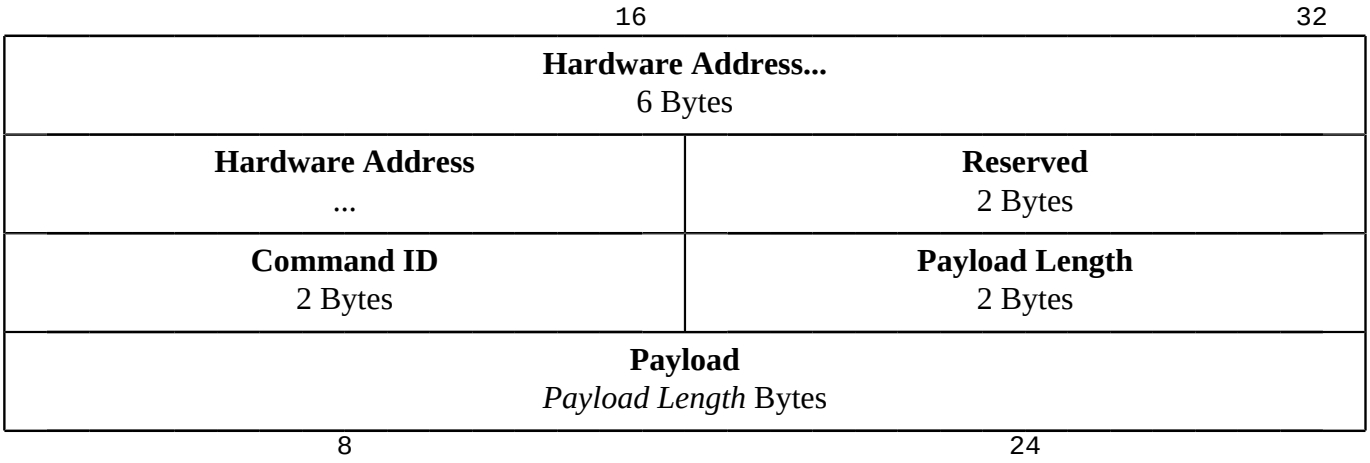
```
<item>
 <idx>1</idx>
 <type>IPV4</type>
 <str>192.168.1.2</str>
</item>
<item>
 <idx>2</idx>
 <type>NAME</type>
 <str>MyHostname.de</str>
</item>
```

If a list is supplied the ALL addresses MUST be in the list also in the case that they are IPv4 type

© Copyright 2001-2021 Bosch Security Systems

1.8 Request or setting of system or network parameters using Broadcast Mechanism

In addition to the SETTING_AN_IP_ADDRESS_USING_BROADCAST, an additional way to setup non network configured devices has been introduced. The device itself is selected using the ethernet mac address as the first parameter inside the command structure. A list of commands including payload length information may be added. All commands must be addressed using UDP broadcast message to port 1760.



Hardware Address

MAC address of the unit to configure.

Command ID

ID of broadcast setting command

Payload Length

ID of broadcast setting command"

LED_BLINK	0x0000
SWITCH_DHCP	0x0001

Payload

Payload; depending on Command ID

Payload length 4 bytes. Requests the unit blink the power led; payload containin the number of seconds for blinking. NOTE duration is limited to 60 sec.

Payload length 4 bytes. Switches DHCP on (payload = 1) or off (payload=0).

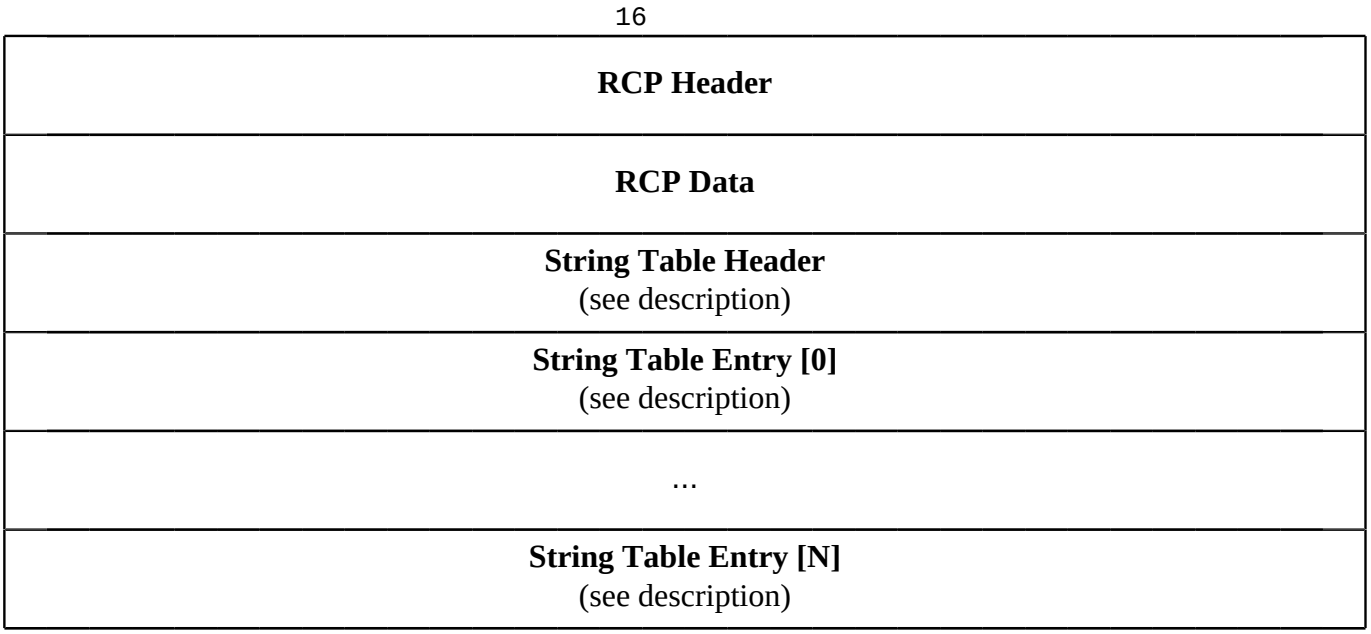
Note: There will be no reply to this command.

© Copyright 2001-2021 Bosch Security Systems

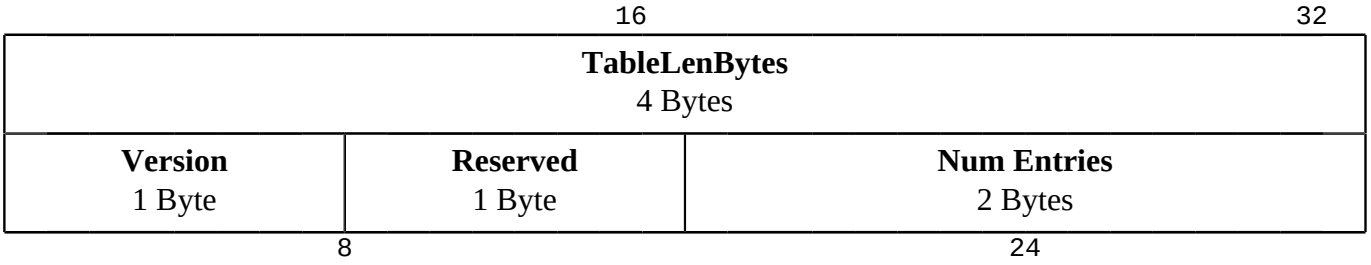
1.9 Extension for RCP+ commands containing IPv6 addresses or host names

For transmitting IPv6 addresses or host names in RCP+ Commands there is a string table extension in firmware version 5.50 and later. The layout of the RCP+ commands remains the same as it was in earlier firmware versions and it is compatible to older clients as long as only IPv4 Addresses are transmitted. If IPv6 Addresses are contained in the command then this extension has to be used.

RCP+ Command with string table



String Table Header



TableLenBytes

Length of hole string table including this header.

Version

Must be 1.

© Copyright 2001-2021 Bosch Security Systems

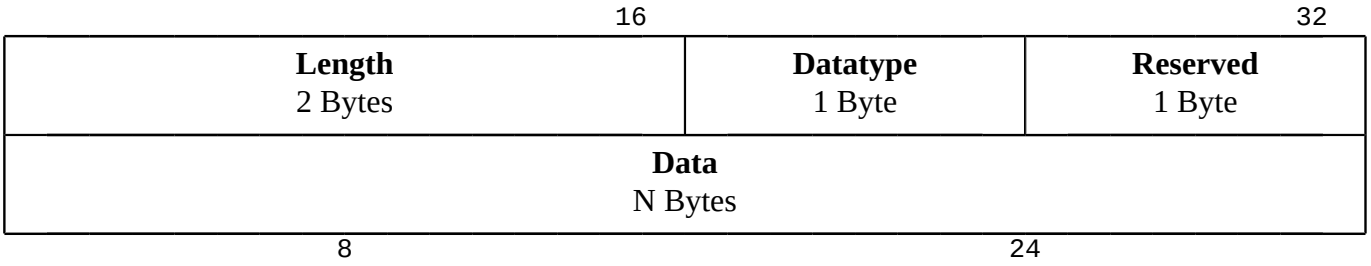
Reserved

Must be 0.

Num Entries

Number of entries in the table

String Table Entry



Length

Length of this entry including this header in bytes.

Datatype

Type of content:

IPv4	0	IPv4 binary encoded value (always 4 bytes)
IPv6	1	IPv6 binary encoded value (always 16 bytes)
HostName	2	HostName or IPv4/IPv6 Address String (encoded as zero terminated string)

Reserved

Must be 0.

Data

N bytes of entry data.

Rules for using the table

- The "StringTable" extension is an option.
- If the RCP-Data contain any IP-Address which current(!) value is other than an IPv4 type then the StringTable option must be used.
- If the RCP-Data does not contain any IP-Address which current(!) value is other than an IPv4 type then the StringTable option should NOT be used to be compatible with older firmware versions and with commands which might not (yet) support this feature.
- If the StringTable is used then:
- The string_table_avail flag must be set to 1 in the RCP+ Header
- All IP-values within "RCP-Data" must be stored in the string table (also when they are IPv4 values)

© Copyright 2001-2021 Bosch Security Systems

- The DWORD-field in "RCP-Data" (which normally would store the IPv4 value) has to contain an offset-pointer to the location in the string table where the actual IP-Address-value can be found. Offset has to be calculated from "Head of StringTable Header" to "Head of StringTable Entry Header" in Bytes.
- There is only one StringTable allowed in an RCP+ Packet.
- The len field of the RCP+ Header must NOT include the size of the StringTable.
- The len field of the TPKT Header (Framing Header) MUST include the size of the StringTable. (This means: RCP-Packet and StringTable are both content of the same TPKT Packet)

© Copyright 2001-2021 Bosch Security Systems

1.10 RCP Command Notification

A RCP client can register for a special message CONF_RCP_CMD_NOTIFICATION (0xff23) in order to be informed when any other RCP+ command is written to the device.

The message is sent out at a frequency of maxium 100ms and contains the tag codes of all RCP+ commands with the direction 'write' that have been processed by the device since the last message. If no RCP+ command was written within this time, no message will be sent. Besides the tag codes the message contains some more information about the commands as described below.

The payload of the message contains one or multiple elements of the following structure (one element for each RCP command). The maxium number of elements is 32.

Message payload element

16		32
Command Tag 2 Bytes	**Datatype** 1 Byte	**Direction** 1 Byte
Action 1 Byte	**Reserved** 1 Byte	**Client ID** 2 Bytes
Reserved 4 Bytes		
NumDes 2 Bytes	**Count** 2 Bytes	
8	24	

Command Tag

Tag code of the RCP command.

Datatype

Datatype of the RCP command (e.g. P_OCTET).

Direction

Direction of the RCP command (currently only WRITE).

Action

Action of the RCP command (currently only REPLY).

Client ID

Client Id of the RCP client that sent the RCP command.

NumDes

Numeric Descriptor (NumDes) parameter value of the RCP command.

© Copyright 2001-2021 Bosch Security Systems

Count

Number of times that this RCP command has been processed.

© Copyright 2001-2021 Bosch Security Systems

2 RCP Command Reference

The following tables will group rcp commands by their corresponding api.

account

alarm

alarm connections

© Copyright 2001-2021 Bosch Security Systems

app management

audio

© Copyright 2001-2021 Bosch Security Systems

autotracker

backup

bicom

browser

calibration

Cbs

cert store

© Copyright 2001-2021 Bosch Security Systems

CloudWatch

commissioning

debug

decoder/videoout

dyndns

© Copyright 2001-2021 Bosch Security Systems

eap

encoder

© Copyright 2001-2021 Bosch Security Systems

ethernet

© Copyright 2001-2021 Bosch Security Systems

extension

GB28181

http live streaming

http settings

I/Os

© Copyright 2001-2021 Bosch Security Systems

iscsi

Keyboard

lldp

© Copyright 2001-2021 Bosch Security Systems

local time and time server settings and configuration

multicast

network services

© Copyright 2001-2021 Bosch Security Systems

onvif

rcp.connection

rcp.control

rcp.discovery

recording settings

© Copyright 2001-2021 Bosch Security Systems

© Copyright 2001-2021 Bosch Security Systems

replay control

roi

© Copyright 2001-2021 Bosch Security Systems

rtsp settings

security

serial port

snmp

© Copyright 2001-2021 Bosch Security Systems

Socket Knocker

stamping

storage

© Copyright 2001-2021 Bosch Security Systems

stratocast

streaming gateway

syslog

system.settings

© Copyright 2001-2021 Bosch Security Systems

system.status

© Copyright 2001-2021 Bosch Security Systems

transcoder

upnp

user management

video input

© Copyright 2001-2021 Bosch Security Systems

viproc

© Copyright 2001-2021 Bosch Security Systems

wlan

© Copyright 2001-2021 Bosch Security Systems

2.1 CONF_ACCESS_LUN_MGMT_FILE

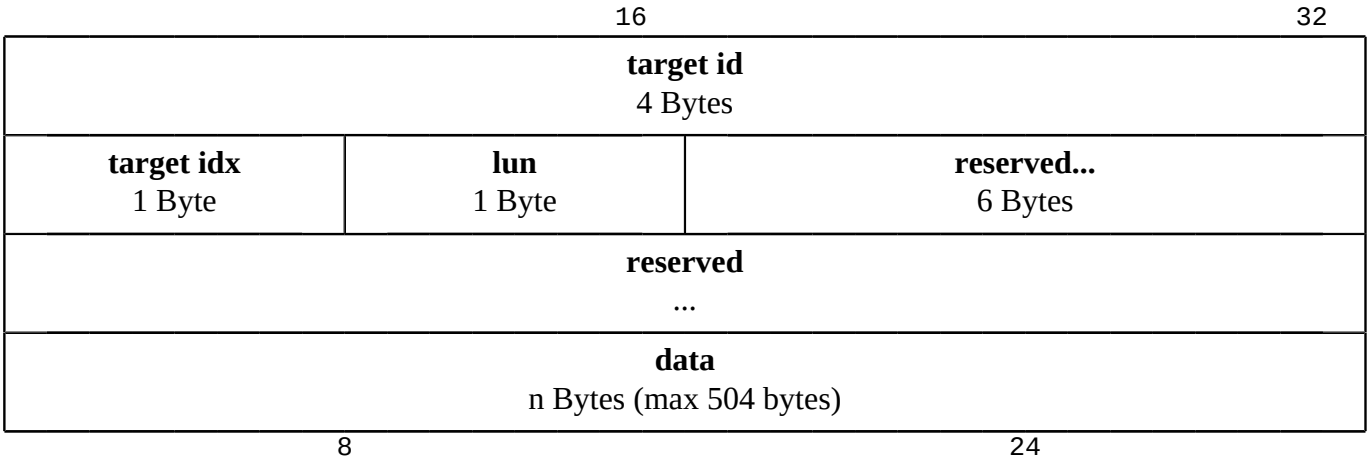
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Access lun management information.	
Write	p_octet	service	Access lun management information.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to acces the lun managment file. This exists only on span formatted luns, which were formatted by firmware /generic dll versions, that are able to create this file. Older versions cannot do this. The file name is "lunmgmt.bin". the max size which can be stored is 504 bytes. The in payload for reading or writing this command has to be at least a size of 12 bytes and includes the lun address for read/ write from/to lun mgmt file. The response on the read direction includes the user data from the lun mgmt file. The data length will be the remayning bytes from the respons payload size minus the lun address and the reserved bytes (12 bytes together). Same in write direction. The payload size determines the number of bytes to write to the lun mgmt file(max 504 bytes).

Payload Structure



target id

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

© Copyright 2001-2021 Bosch Security Systems

target idx

Target index of the lun

lun

Lun

reserved

data

User data to read/write from/to the lun mgmt file

© Copyright 2001-2021 Bosch Security Systems

2.2 CONF_ACCESS_OPTION

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c7f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	always	Read the current access options. 0=normal access, Bit1: camera is in 'force initial password' mode. I.e. before further usage an initial service password must be set.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

	Mask	Name
Bit 0	0x01	Force Initial Password

Combined Values

Mask	Name
0x00	Normal Access

© Copyright 2001-2021 Bosch Security Systems

2.3 CONF_ACCOUNT_LIST

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1d	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Returns the response from the ftp/ dropbox LIST command	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.4 CONF_ACCOUNT_LOGIN_TEST

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b36	account info (1...MAX_ACCOUNT	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	user	Checks if a login to the specified server is possible. Return values: #ValueList: 0=ok; 1=Connect failed; 2=invalid user or password; 3=set directory failed; 4=general error; 5=List failed;	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.5 CONF_ACCOUNT_SETTINGS

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b5e	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Obsolete use CONF_ACCOUNT_SETTINGS_V2	
Write	p_octet	service	Obsolete use CONF_ACCOUNT_SETTINGS_V2	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.6 CONF_ACCOUNT_SETTINGS_V2

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cc0	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns account settings in a tagged format	
Write	p_octet	service	Writes account settings in a tagged format	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

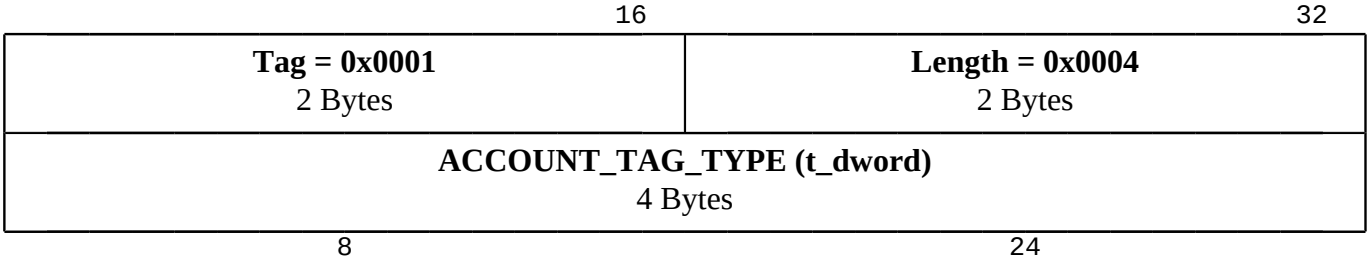
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

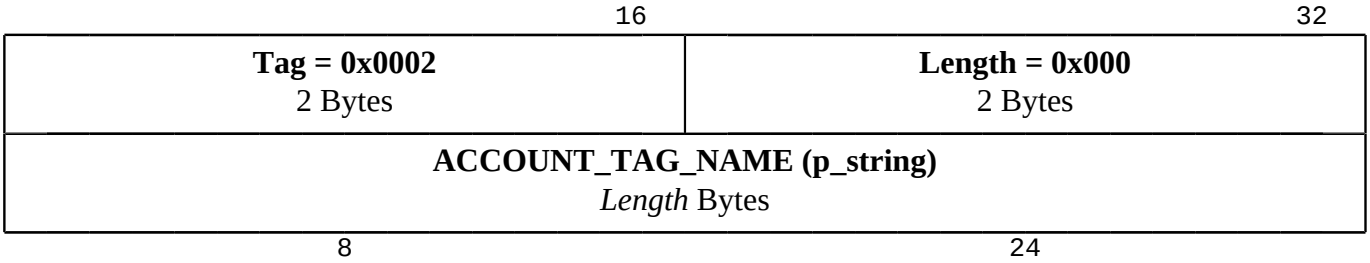
Tag 1: ACCOUNT_TAG_TYPE

0: not configured, 1: ftp, 2: dropbox, 5: amazon s3, 6: kinesis



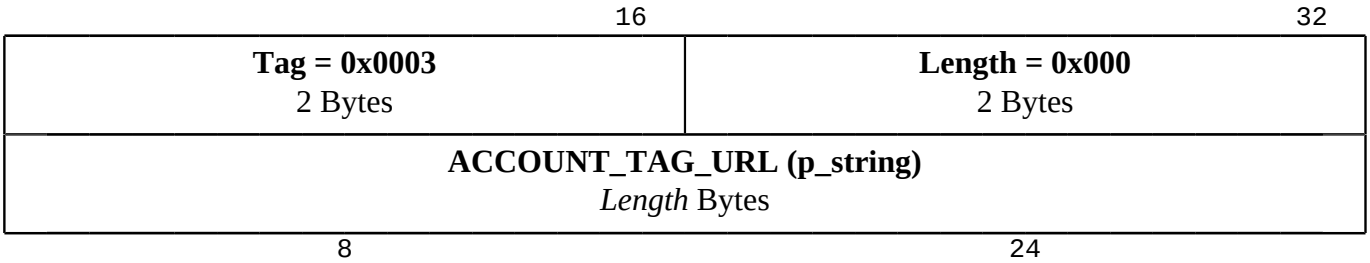
Tag 2: ACCOUNT_TAG_NAME

account name



Tag 3: ACCOUNT_TAG_URL

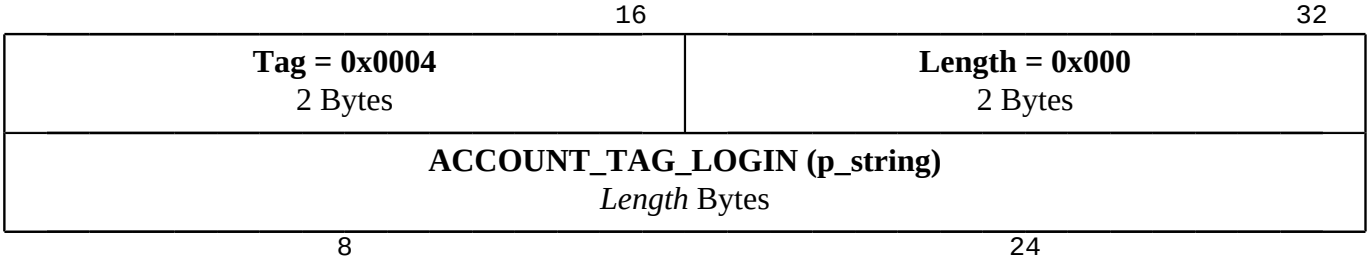
url, e.g url to the ftp server



© Copyright 2001-2021 Bosch Security Systems

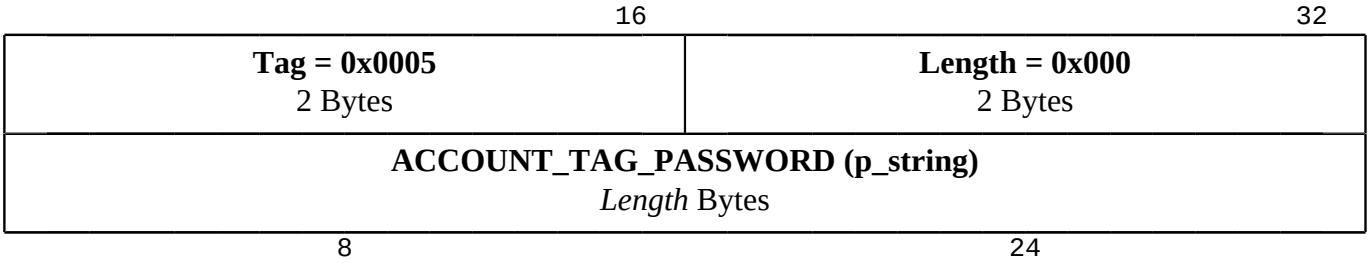
Tag 4: ACCOUNT_TAG_LOGIN

username



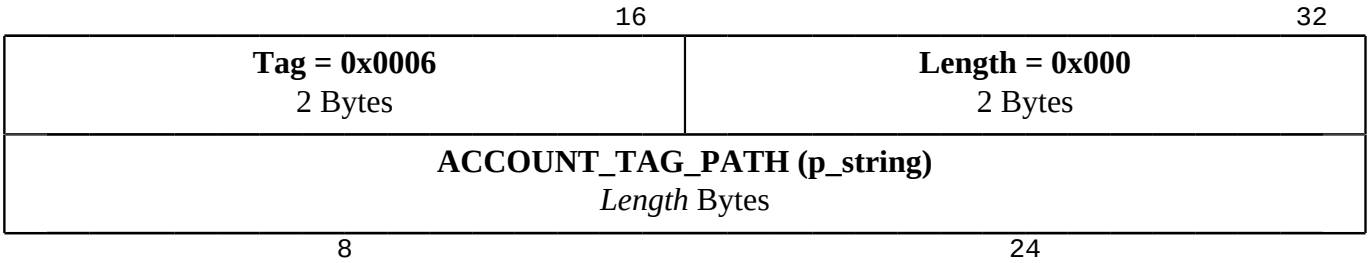
Tag 5: ACCOUNT_TAG_PASSWORD

password



Tag 6: ACCOUNT_TAG_PATH

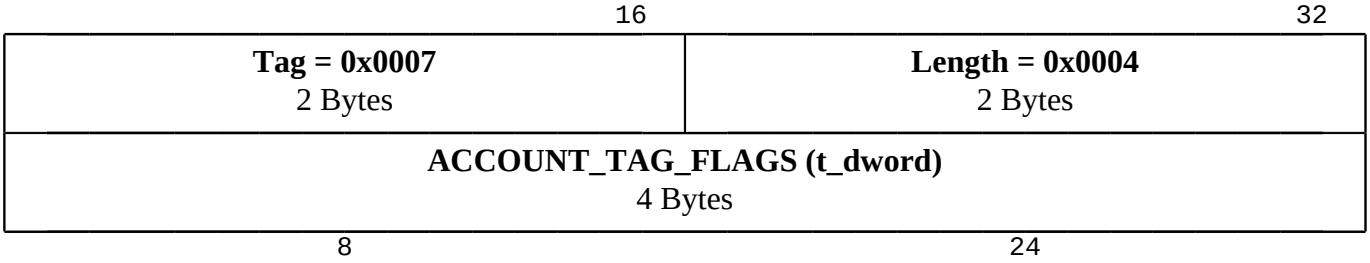
path



© Copyright 2001-2021 Bosch Security Systems

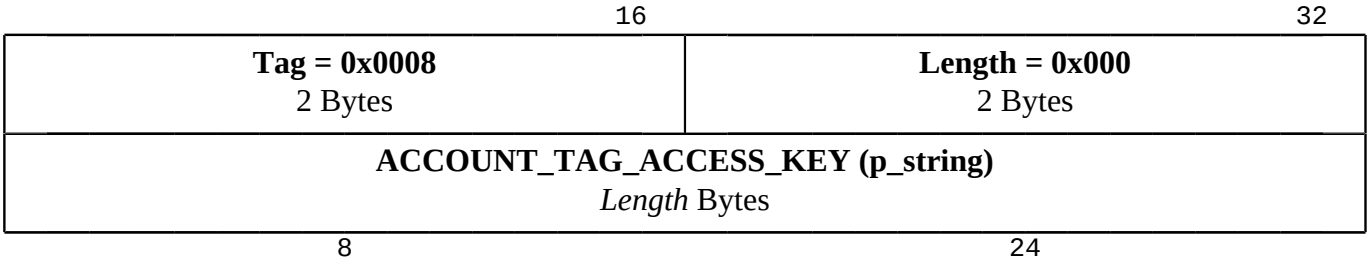
Tag 7: ACCOUNT_TAG_FLAGS

optional flags (e.g. FTP Encryption mode 0=off,1=TLS)



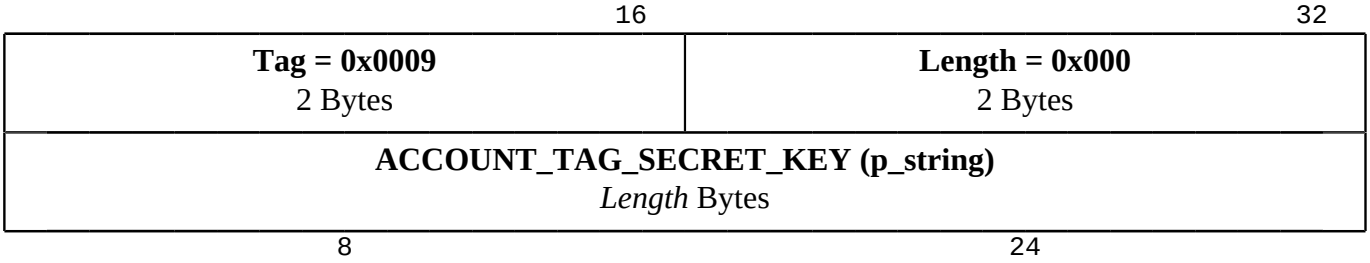
Tag 8: ACCOUNT_TAG_ACCESS_KEY

aws access key (max 128 bytes)



Tag 9: ACCOUNT_TAG_SECRET_KEY

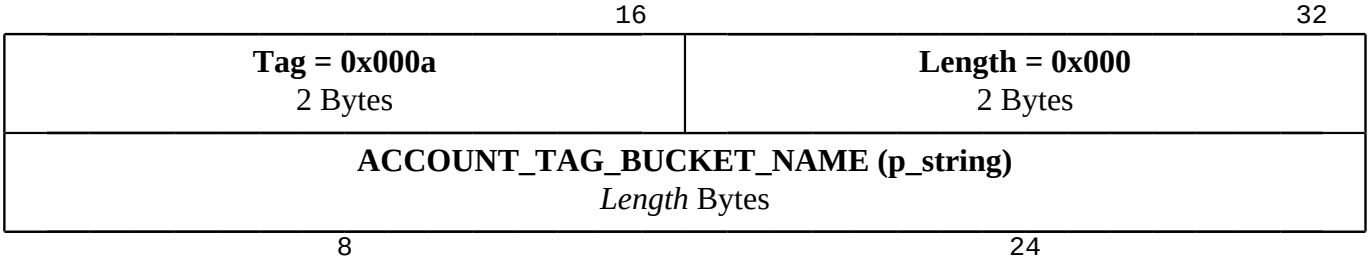
aws secret access key (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

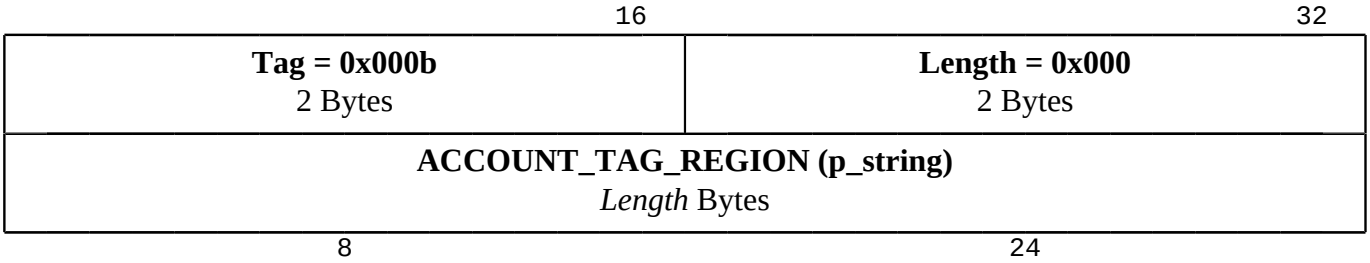
Tag 10: ACCOUNT_TAG_BUCKET_NAME

bucket name (amazon s3), max 128 bytes



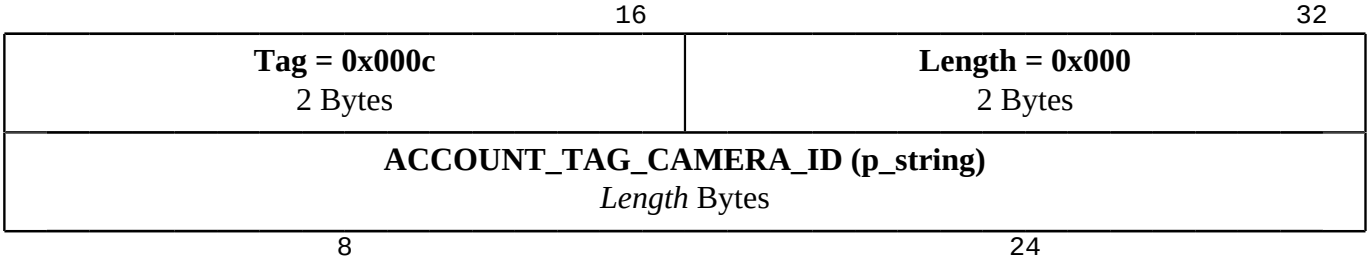
Tag 11: ACCOUNT_TAG_REGION

aws region



Tag 12: ACCOUNT_TAG_CAMERA_ID

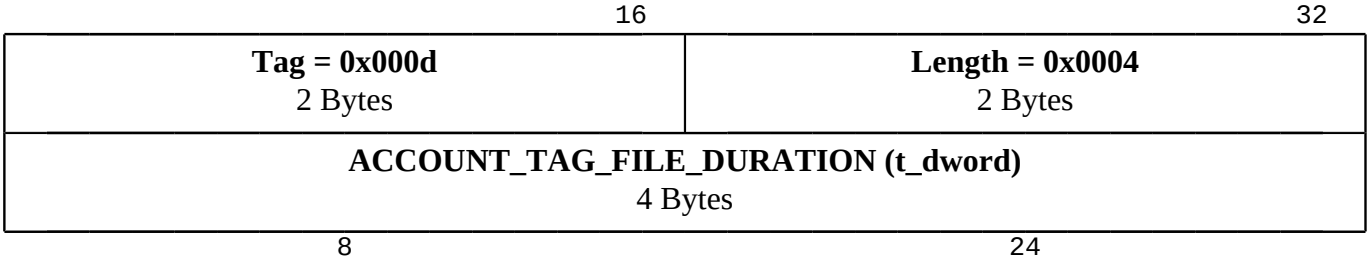
s3 camera id, string will be added to the uploaded filename (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

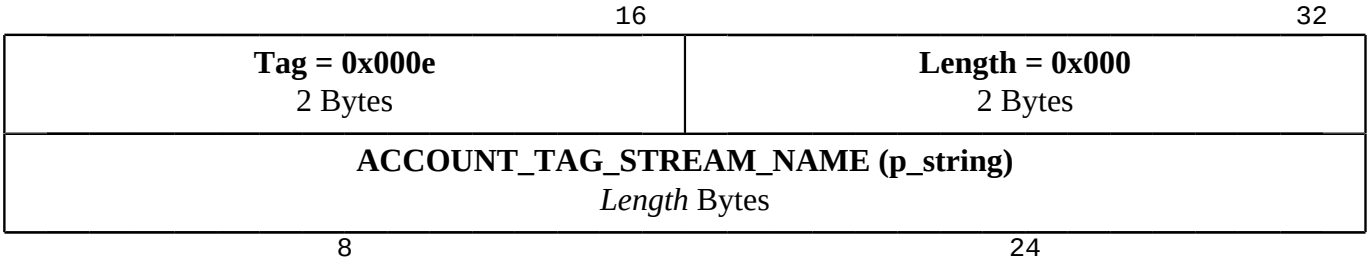
Tag 13: ACCOUNT_TAG_FILE_DURATION

s3 file duration in seconds



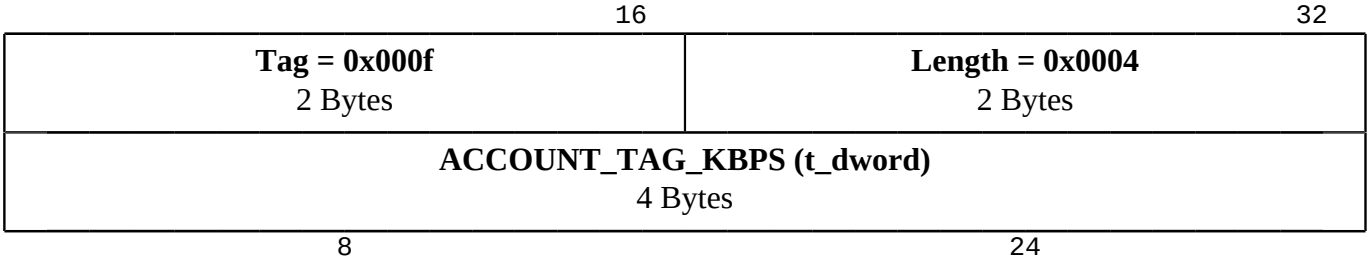
Tag 14: ACCOUNT_TAG_STREAM_NAME

stream name



Tag 15: ACCOUNT_TAG_KBPS

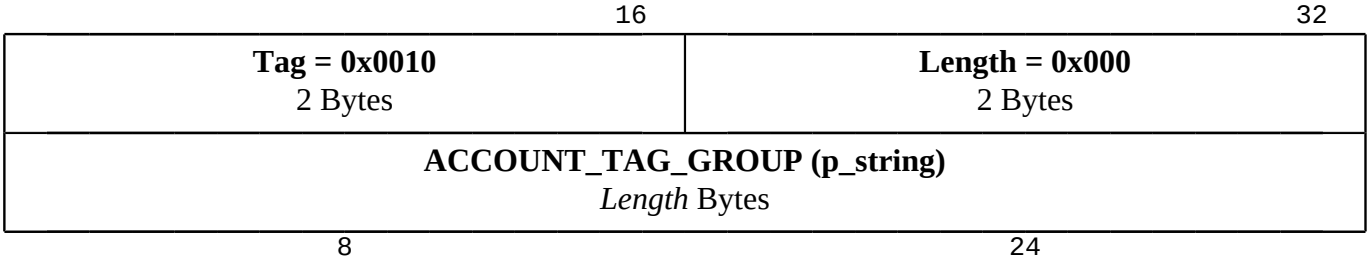
maximum datarate for the backup, see CONF_BACKUP_MAX_KBPS



© Copyright 2001-2021 Bosch Security Systems

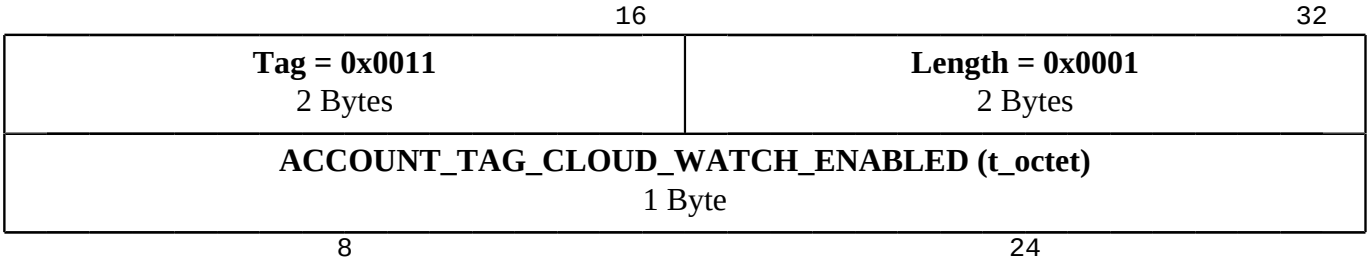
Tag 16: ACCOUNT_TAG_GROUP

cloud watch log group



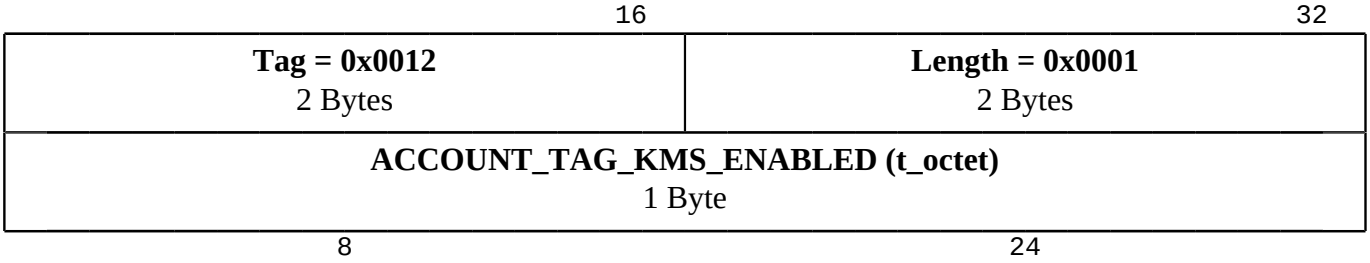
Tag 17: ACCOUNT_TAG_CLOUD_WATCH_ENABLED

cloud watch enabled/disabled



Tag 18: ACCOUNT_TAG_KMS_ENABLED

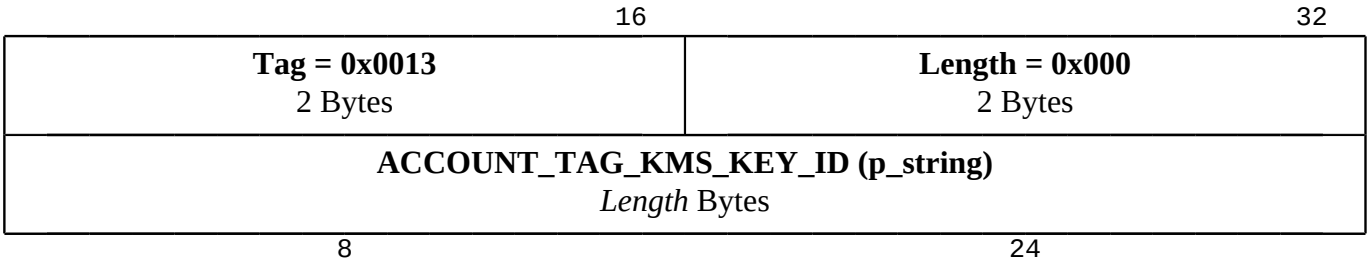
kms key encryption enabled/disabled



© Copyright 2001-2021 Bosch Security Systems

Tag 19: ACCOUNT_TAG_KMS_KEY_ID

aws kms key id (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

2.7 CONF_ACCOUNT_STATUS

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bde	account number	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.8 CONF_ACCOUNTS_CREATE_FOLDER

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b6c	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Creates a new folder. Argument: folder name	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.9 CONF_ACCOUNTS_DELETE_FOLDER

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b6f	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Deletes a folder. Argument: folder name	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.10 CONF_ACTIVE_CONNECTION_LIST

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xffc1	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

ConnectionEntry [0] (see description)
...
ConnectionEntry [N] (see description)

ConnectionEntry

16		32
Destination IP Address 4 Bytes		
Local Coder 1 Byte	**Local Line** 1 Byte	**Flags** 2 Bytes
Session ID 4 Bytes		
Remote Coder 1 Byte	**Remote Line** 1 Byte	**Reserved** 2 Bytes
TX Channels 4 Bytes		
RX Channels 4 Bytes		
8		24

Destination IP Address

IP address to which the unit is connected.

© Copyright 2001-2021 Bosch Security Systems

Local Coder

The local connected coder (relative to line).

Local Line

The remote connected line

Flags

	Mask	Name	Description
Bit 11	0x0800	H.265	Is set when connection is H.265
Bit 10	0x0400	Transcoder	Is set when connection is using transcoder
Bit 9	0x0200	Jpeg	Is set when connection is Jpeg
Bit 8	0x0100	H.264	Is set when connection is H.264
Bit 7	0x0080	Encrypted	Is set when datastream is encrypted
Bit 6	0x0040	TCP Transportation	Is set when connection is using TCP for transportation
Bit 5	0x0020	Multicast	Is set when connection is using multicast
Bit 4	0x0010	Streaming	Is set when connection is started as streaming
Bit 3	0x0008	HDD Replay	Is set when connection is to HDD replay
Bit 2	0x0004	MPEG4	Is set when connection is MPEG4
Bit 1	0x0002	MPEG2 PRG	Is set when connection is MPEG2 PRG
Bit 0	0x0001	MPEG2 VES	Is set when connection is MPEG2 VES

Session ID

Session identifier.

Remote Coder

The remote connected coder (relative to line), (absolute) transcoder if connection is using transcoder

Remote Line

The remote connected line.

Reserved

TX Channels

See CONNECT_TO command for bit mask.

RX Channels

See CONNECT_TO command for bit mask.

Note: If a session ID is provided in the rcp header, then only the connection list of this specific session ID is supplied.

© Copyright 2001-2021 Bosch Security Systems

2.11 CONF_ACTIVE_VIPROC_CONFIG

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3f	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the config id activated by bool engine.	
Write	t_dword	iva	Sets the specified config ID as active. The command has only an effect, if the VIPROC_MODE is set to SCRIPT mode (0xff). this command is not available on a gen4 dome.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.12 CONF_ADAPT_ENC_PROFILE_BITRATES

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ce1	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Adapt bitrates of the camera	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.13 CONF_ADD_REMOTE_DEVICE

[API: transcoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b52	entry index(1...4)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live	Read entry of remote devices, see detailed description	
Write	p_octet	iva	Add entry of remote devices, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Description

Adds a device with forwarder ports, all data to/from the forwarder ports will be forwarded to/from the configured device port. It is allowed to store complete zeroed forwarder/remote port entries. For entries which are referenced by CONF_RCP_CONNECT_SALVO command num param, it it nessessary, that the first forward port entry contains the http port and the second one the https port. If one of these ports shall not be used, at least a zeroed entry (remote/forwarder port to zero) has to be placed. The max lines and coder per lines are just information, which will be stored together in the config. To delete an entry, send an empty or zeroed payload.

Payload Structure

16	32
mac... 6 Bytes	
mac ...	**reserved** 2 Bytes
line cnt 2 Bytes	**max coder per line** 2 Bytes
nbr of forwarder entries 4 Bytes	
forwarder entry [0] (see description)	
...	
forwarder entry [N] (see description)	

© Copyright 2001-2021 Bosch Security Systems

url len 2 Bytes	**url** *url len* Bytes
pwd len 2 Bytes	**pwd** *pwd len* Bytes
device name len 2 Bytes	**device name** *device name len* Bytes

8 24

mac

Mac of the device (will be stored only as information in config)

reserved

line cnt

Max lines of the device (will be stored only as information in config)

max coder per line

Max number of coder instances per line (will be stored only as information in config)

nbr of forwarder entries

Number of forward port entries following this field (actual limited to 2)

forwarder entry

16 32

remote port 2 Bytes	**forwarder port** 2 Bytes

8 24

remote port

Port of the remote device

forwarder port

Local port which forwards all data to the remote port

url len

Length of the following url (actual limit is 64)

url

Url for the connection to the remote device, ascii string including zero termination

© Copyright 2001-2021 Bosch Security Systems

pwd len

Length of the following password (actual limit is 64)

pwd

Password for the connection, ascii string including zero termination

device name len

Length of the following device name (actual limit is 64)

device name

Device name of the remote device, ascii string including zero termination

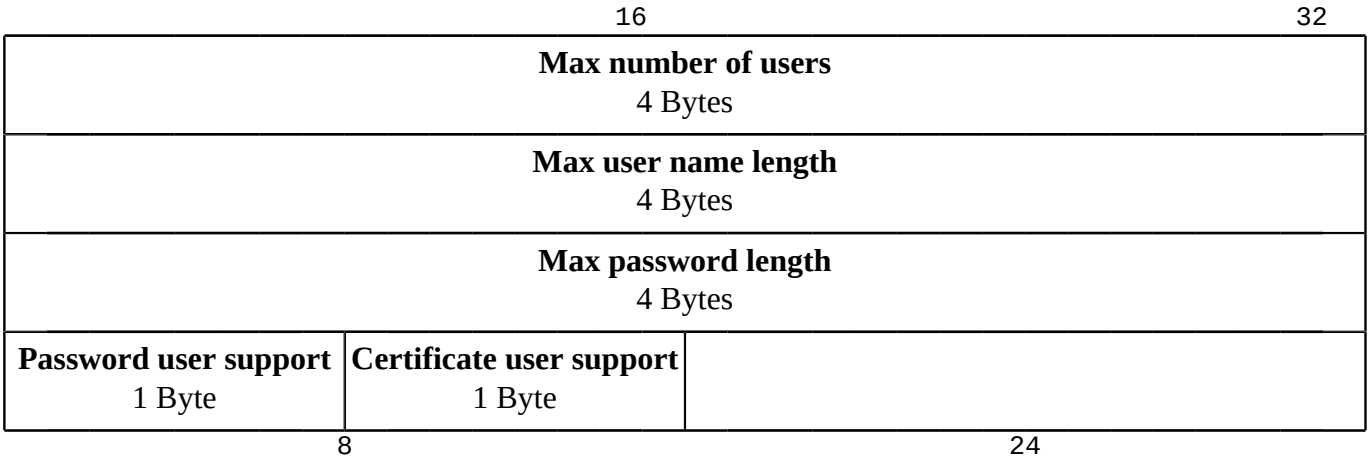
© Copyright 2001-2021 Bosch Security Systems

2.14 CONF_ADV_USER_OPTIONS

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bd9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Max number of users

In addition to the standard users 'user', 'service', 'live'

Password user support

no	0
Yes	1

Certificate user support

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

2.15 CONF_ADV_USER_SETTINGS

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bda	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	no_pwd		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_ADV_USER_SETTINGS allows to query or configure user entries for camera access control. User entries are indentified by the user name, the user name of each entry must be unique. Supported lengths, types and restrictions can be queried using CONF_ADV_USER_SETTINGS_OPTIONS.

In read direction only the user name is used to find the user entry to query the user configuration.

In write direction an already existing user with the same name is modified or a new user with the given configuration is created.

Payload Structure

User Entry

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

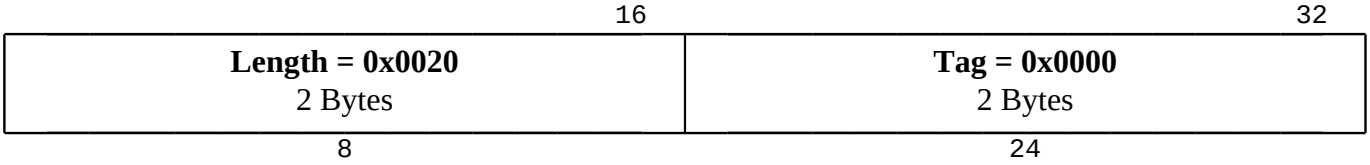
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

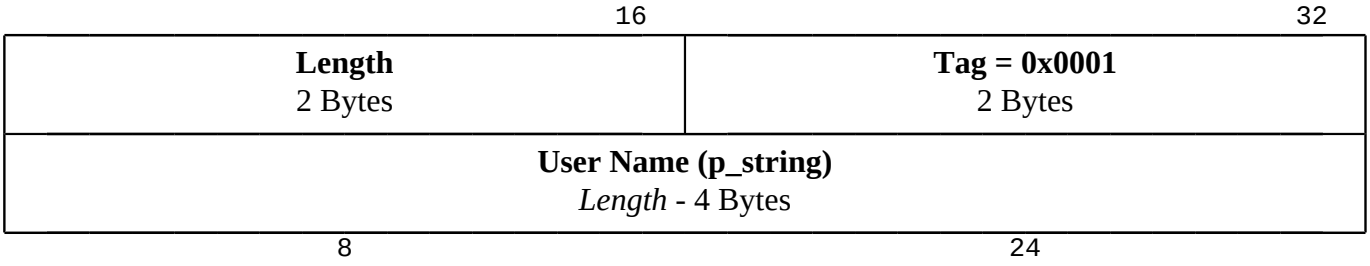
Tag 0: User Entry

The payload of tag user entry contains further tagged values described below.

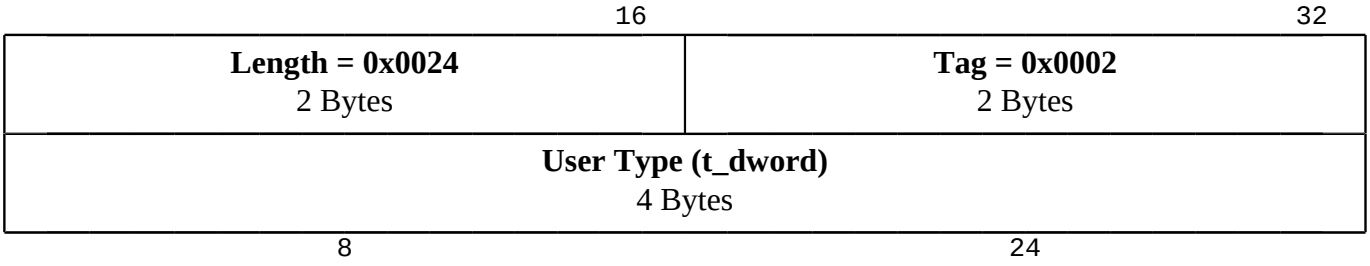


Tag 1: User Name (Required)

Name of the user in ascii chars, no zero termination



Tag 2: User Type



The payload of tag user type contains the user type, possible values are:

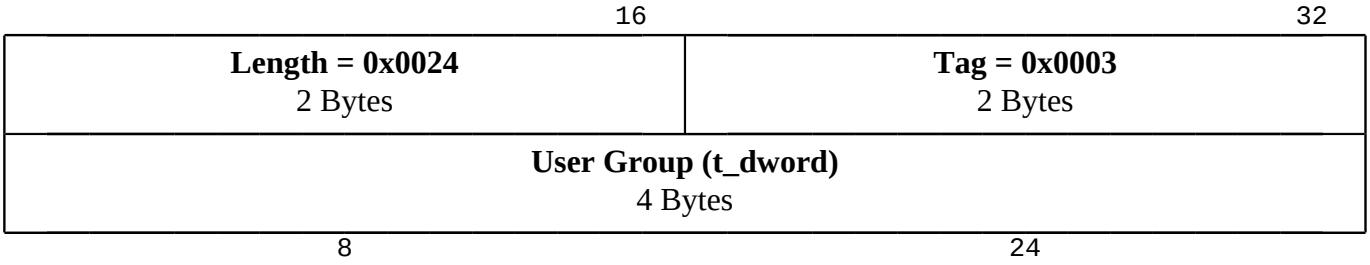
Deleted	0	
Password	1	
Certificate	2	
Temporary with password	3	
Cloud	4	(read only)

The following rules apply to a user entry:

- Changing the type of an existing user to Deleted will delete the user.
- A User Entry of type Password must contain a password entry.
- A User Entry of type 'Temporary with password' must contain a password entry and an 'Expires' entry.

© Copyright 2001-2021 Bosch Security Systems

Tag 3: User Group

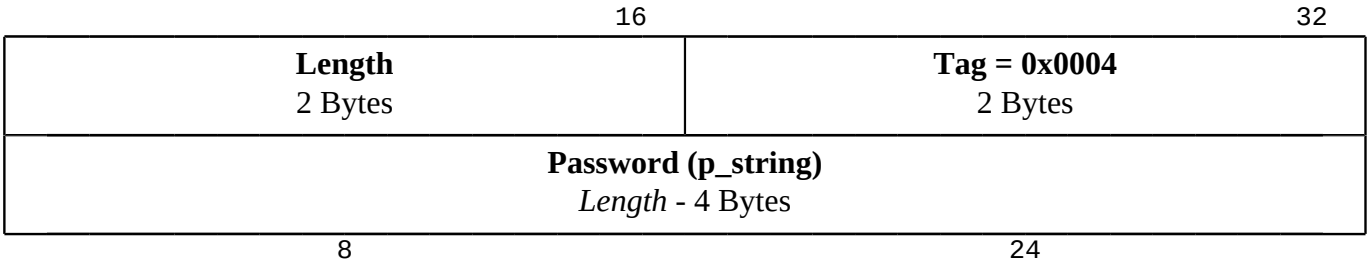


Numeric unique ID of the user group the user should belong to:	
Reserved do not use	0
Live	1
User	2
Service	3
IVA	6

Tag 4: Password

The payload of tag user password contains the user password.

This field is only mandatory when configuring a user of type Password and will be empty when read if there is no password or just '********' when a password is set.



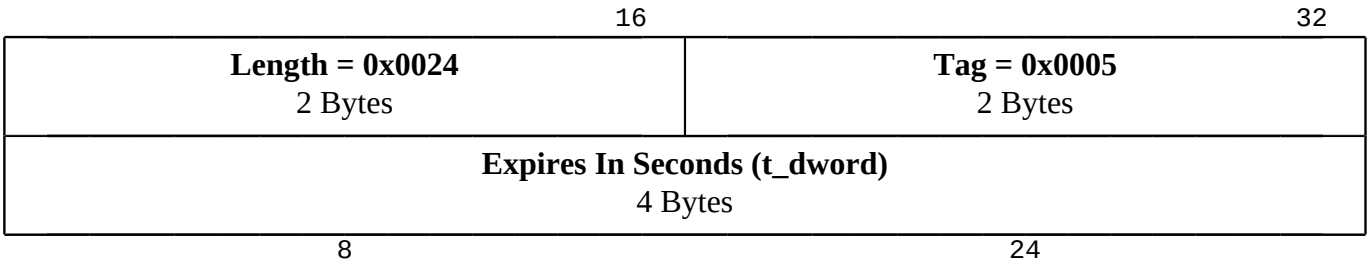
Tag 5: Expires In Seconds

The 'Expires' tag indicates the life time of the user in seconds. So if the user should life for 1 hour pass the value 3600 here.

The field is mandatory when you create a user of type 'Temporary with password' (3). For all other types this field should not be present or have the payload 0.

© Copyright 2001-2021 Bosch Security Systems

On read, this field will indicate the remaining time in seconds until the user will expire.



© Copyright 2001-2021 Bosch Security Systems

2.16 CONF_ADV_USER_SETTINGS_LIST

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bdc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live	Returns a list of tagged values(User Entries).	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

User Entry [0]
...
User Entry [N]

The payload of this command consists of a list of 'User Entry' items of the command CONF_ADV_USER_SETTINGS.

© Copyright 2001-2021 Bosch Security Systems

2.17 CONF_ALARM_BACKUP_REC_SPEED_LIMIT

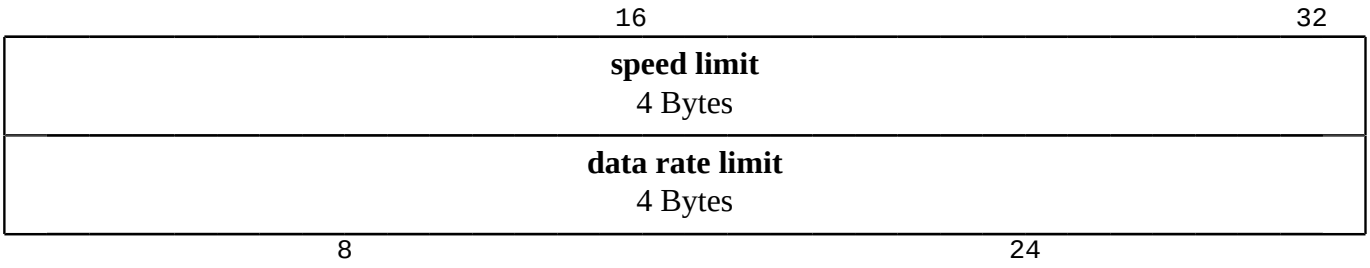
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb0	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Get the speed limit parameter, see detailed description	
Write	p_octet	service	Set the speed limit parameter, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to limit the data traffic caused by backuped recording per line (num parameter 1 to n). This can be done by limiting the data rate and/or the speed in percent(100 percent means backup speed like replay in normal speed), based on the time information of the video data. The backup can be stopped completly, by sending a speed limit by 0 and continued again by sending a non zero speed later. It cannot be stopped by the data rate, a value of 0 for the data rate means to disable the limit for the data rate. The data rate parameter isn't applied immediately, but when the next backup starts.

Payload Structure



speed limit

Speed limit in percent (0 to 9999, default 200)

data rate limit

Data rate limit in Kbps (0 - no limit, default 0)

© Copyright 2001-2021 Bosch Security Systems

2.18 CONF_ALARM_CONNECT_TO_IP

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0041	destination IP number	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the connect on alarm event IP address	
Write	t_dword	service	Specify the connect on alarm event IP address	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.19 CONF_ALARM_CONNECT_TO_IP_STR

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0081	destination IP number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the alarm IP using string notation (xxx.xxx.xxx.xxx)	
Write	p_string	service	Set alarm IP using string notation (xxx.xxx.xxx.xxx)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.20 CONF_ALARM_CONNECTION_DESTINATION_PORT

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a15	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Returns the network destination port on outgoing alarm connections	
Write	t_word	service	Selects the destination network port for outgoing alarm connections; all other than 1756 (default RCP port) will use the HTTP tunneling protocol. In this case, this must be the HTTP/HTTPS port of the called host.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.21 CONF_ALARM_CONNECTION_USE_SSL

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a16	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Use RCP SSL for outgoing alarm connections	
Write	f_flag	service	Use RCP SSL for outgoing alarm connections	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.22 CONF_ALARM_DISP_VAL

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x008e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read current state	
Write	t_octet	service	Set alarm display state	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

disable alarm display	1	
enable alarm display	2	
alarm stamping with custom attributes	3	can be set with the CONF_STAMP_ATTR_ALARM

© Copyright 2001-2021 Bosch Security Systems

2.23 CONF_ALARM_INPUT_CAPABILITIES

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read capabilities of the available alarm inputs.

Payload Structure

16	32
Count 4 Bytes	
Capabilities [0] 4 Bytes	
...	
Capabilities [N] 4 Bytes	
8	24

Count

Number of available alarm inputs

Capabilities

Capabilities of the alarm input

	Mask	Name
Bit 0	0x00000001	Supervised mode possible

© Copyright 2001-2021 Bosch Security Systems

2.24 CONF_ALARM_INPUT_LH_VAL

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x008d	alarm input	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	1=alarm pin high active; 2=alarm pin low active	
Write	t_octet	service	1=set alarm pin high active; 2=set alarm pin low active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.25 CONF_ALARM_INPUT_SUPERVISED

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b87	alarm input	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	1= alarm pin is set to supervised mode; 0= alarm pin is set to normal mode (non-supervised)	
Write	t_octet	service	1=set alarm pin to supervised mode; 0=set alarm pin to normal mode (non-supervised)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.26 CONF_ALARM_OVERVIEW

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c38	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Messages are sent if a change has happened (event, state change, alarm added, alarm deleted). The readout command contains all registered alarm entries.

Payload Structure

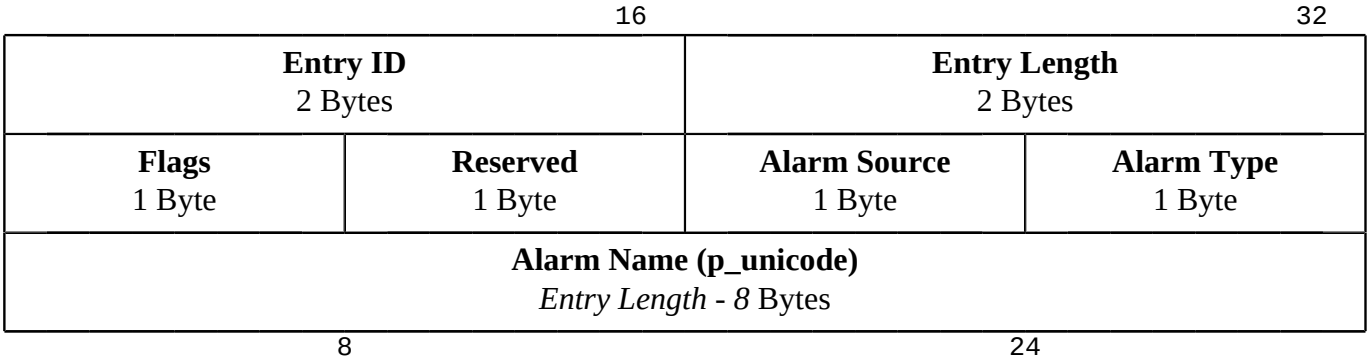
16		32
Flags 1 Byte	**Reserved** 1 Byte	**Number of alarm entries** 2 Bytes
Alarm Entry [0] (see description)		
...		
Alarm Entry [Number of alarm entries - 1] (see description)		
8		24

Flags

	Mask	Name	Description
Bit 7	0x80	All entries included	(read out)

© Copyright 2001-2021 Bosch Security Systems

Alarm Entry



Entry ID

Unique ID for each entry. May be reassigned to entry if entry is deleted.

Entry Length

Length of the entry payload, inclusive header.

Flags

	Mask	Name	Description
Bit 7	0x80	Add Flag	Alarm Entry has been registered
Bit 6	0x40	Delete Flag	Alarm Entry has been unregistered
Bit 5	0x20	State Flag	Alarm Entry is a state
Bit 4	0x10	State Set Flag	Alarm Entry State is set (states only)

Alarm Source

Values:

Unknown	0
Relais	2
Digital Input	3
Audio	4
Virtual Input	5
Tamper	6
Motion	7
Flow	8
Intelligent Video Analytics	9
Fire	10
Man Over Board	11
storage	13
recorder	14

© Copyright 2001-2021 Bosch Security Systems

Alarm Type

Values:

Unknown	0
VCA	1
Relais	2
Digital Input	3
Audio	4
Virtual Input	5
storage medium failure	6
storage recording state	7
disk mounted	8
disk unmounted	9
Default Task	16
Global Change	17
Signal too bright	18
Signal too dark	19
Signal too noisy	20
Signal too blurry	21
Signal loss	22
Reference Image Check Failed	23
Invalid Configuration	24
Flame Detected	25
Smoke Detected	26
Object in field	32
Crossing line	33
Loitering	34
Condition change	35
Following route	36
Tampering	37
Removed object	38
Idle object	39
Entering field	40
Leaving field	41
Similarity search	42
Crowd detection	43
Flow in field	44
Counter flow in field	45
Motion in field	46
Man over board	47
Counter	48
BEV People Counter	49
Occupacy	50

© Copyright 2001-2021 Bosch Security Systems

Alarm Name

String encoded in unicode.

© Copyright 2001-2021 Bosch Security Systems

2.27 CONF_ALARM_STRING

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0090	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Get the alarm string	
Write	p_unicode	service	Set the alarm string (max 32 unicode characters)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.28
CONF_ALLOW_BASIC_HTTP_AUTH_ON_NON_SSL_SOCK

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Allow basic http authentication on non SSL sockets,#ValueList: 0= not allowed (default); 1= allowed	
Write	f_flag	service	Allow basic http authentication on non SSL sockets, #ValueList: 0= not allowed (default), 1= allowed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.29
CONF_ALLOW_OVERWRITE_TIMESRVIP_BY_DHCP

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c0e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Allow or deny that dhcp overwrites time/ntp srv ip, 0 = deny, 1 = allow	
Write	t_octet	service	Allow or deny that dhcp overwrites time/ntp srv ip, 0 = deny, 1 = allow	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.30 CONF_AMBIENT_LIGHT_LEVEL

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c45	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Get the normalized ambient light level (value range: 0-1000; typical values between 0-400); Only available on certain devices	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.31 CONF_APP_MGMT_GET_DEVICE_IDS

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	GetDeviceIDs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.32 CONF_APP_MGMT_INSTALL_LICENSE

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	Install license	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.33 CONF_APP_OPTION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	DEPRICATED: Use CONF_APP_OPTION_EXT. Returns a bit field containing 256 bits (32 byte). Each bit represents one active license on the device. Bit order is MSB first.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.34 CONF_APP_OPTION_EXT

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c4a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns a bit field containing 512 bits (64 byte). Each bit represents one active license on the device. Bit order is MSB first.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.35 CONF_APP_OPTION_SET

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Returns a readable string with the result code of the last applied set option	
Write	p_string	service	A key, generated by the license server to enable a specific application option; after entering a key, the APP_OPTION command returns the current activated application options	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.36 CONF_APP_OPTION_UNIT_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e1	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read out the unique unit id (installation code) for setting application options	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.37 CONF_APPLICATION_TYPE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c47	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Returns the application type,	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

default application	0
standard application	1

© Copyright 2001-2021 Bosch Security Systems

2.38 CONF_APPLY_NETWORK_SETTINGS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cbe	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	service	Apply the configured network settings at runtime	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.39 CONF_AUDIO_AAC_BITRATE

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b9a	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the bitrate of the AAC encoder (only value of 48000 and 80000 are supported) for the line given by num	
Write	t_dword	service	Set the bitrate for the AAC encoder (only value of 48000 and 80000 are supported) on the line given with the num parameter	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.40 CONF_AUDIO_ENC_STREAMING

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x099b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Audio streaming on coder (audio encoder n: coderbits=1<<(n-1))	
Write	t_dword	service	Enables audio streaming on coder (audio encoder n: coderbits=1<<(n-1))	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.41 CONF_AUDIO_INPUT

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09b8	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the audio input source	
Write	t_dword	service	Set the audio input source	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Line	0	
Mic	1	
Mute	2	only supported by arm based products

© Copyright 2001-2021 Bosch Security Systems

2.42 CONF_AUDIO_INPUT_LEVEL

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x000a	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets current level of audio input. Range: 0 to AUDIO_INPUT_MAX	
Write	t_dword	service	Sets level of audio input. Range: 0 to AUDIO_INPUT_MAX	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.43 CONF_AUDIO_INPUT_MAX

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ba	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the maximum input level that can be adjusted	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.44 CONF_AUDIO_INPUT_PEEK

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09c6	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the max iabs of the audio input signal (clears the value)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.45 CONF_AUDIO_LOUDSPEAKER_ON_OFF

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09be	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Gets the loudspeaker state.	
Write	f_flag	service	Sets the loudspeaker state.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Off	0
On	1

© Copyright 2001-2021 Bosch Security Systems

2.46 CONF_AUDIO_MIC_LEVEL

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09bc	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets level of mic input. Range: 0 to AUDIO_MIC_MAX	
Write	t_dword	service	Sets level of mic input. Range: 0 to AUDIO_MIC_MAX	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.47 CONF_AUDIO_MIC_MAX

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09bd	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the maximum mic level that can be adjusted	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.48 CONF_AUDIO_NOTIFICATION_SOUND

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c77	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Read status of audio notification sound (only applicable on certain devices).	
Write	f_flag	service	Switch audio notification sound on/off (only applicable on certain devices).	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

privacy mode sound 1

Values:

do not play sound	0
play sound	1

© Copyright 2001-2021 Bosch Security Systems

2.49 CONF_AUDIO_ON_OFF

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x000c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Get the audio state	
Write	f_flag	service	Set the audio state	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Off	0
On	1

© Copyright 2001-2021 Bosch Security Systems

2.50 CONF_AUDIO_OPTIONS

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09bf	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets options for Audio.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

	Mask	Name
Bit 3	0x08	Loudspeaker
Bit 2	0x04	Mic
Bit 1	0x02	Line Out
Bit 0	0x01	Line In

© Copyright 2001-2021 Bosch Security Systems

2.51 CONF_AUDIO_OUTPUT

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09b9	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the audio output target	
Write	t_dword	service	Set the audio output target	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Decoder	0
Sine	1
Loop	2
Mute	3
Sample	4

© Copyright 2001-2021 Bosch Security Systems

2.52 CONF_AUDIO_OUTPUT_LEVEL

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09b7	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets level of audio output. Range: 0 to AUDIO_OUTPUT_MAX	
Write	t_dword	service	Sets level of audio output. Range: 0 to AUDIO_OUTPUT_MAX	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.53 CONF_AUDIO_OUTPUT_MAX

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09bb	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the maximum output level that can be adjusted	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.54 CONF_AUDIO_OUTPUT_PEEK

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09c7	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the max iabs of the audio output signal (clears the value)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.55 CONF_AUDIO_REC_FORMAT

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae9	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get audio format for recording	
Write	t_octet	service	Set audio format for recording (running recording has to be stopped and restarted in order to become effective)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

None	0
G711	1
L16	2
AAC	3

© Copyright 2001-2021 Bosch Security Systems

2.56 CONF_AUDIO_STARTUP_SOUND

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09b6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Turn on/off audio startup sound	
Write	f_flag	service	Turn on/off audio startup sound	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.57 CONF_AUDIO_STREAMING_ENCODING

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b8d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Defines the audio encoding that is used for streaming (1: g711, 2: aac)	
Write	t_dword	service	Defines the audio encoding that is used for streaming (1: g711, 2: aac)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

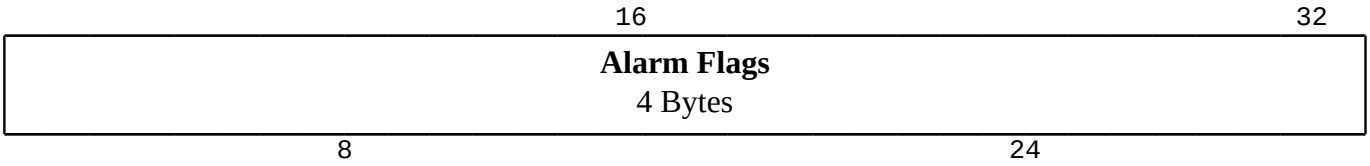
© Copyright 2001-2021 Bosch Security Systems

2.58 CONF_AUPROC_ALARM

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a79	audio line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Alarm Flags

The message is sent whenever any of the bit values changes. Additionally it is sent once per second when any of the bits is set.

Values:

	Mask	Name	Description
Bit 13	0x00002000	Entire	Entire frequency band
Bit 12	0x00001000	Hz2865_4000	Frequency band 2875-4000 Hz
Bit 11	0x00000800	Hz2420_3400	Frequency band 2420-3400 Hz
Bit 10	0x00000400	Hz2030_2875	Frequency band 2030-2875 Hz
Bit 9	0x00000200	Hz1690_2420	Frequency band 1690-2420 Hz
Bit 8	0x00000100	Hz1375_2030	Frequency band 1375-2030 Hz
Bit 7	0x00000080	Hz1110_1690	Frequency band 1110-1690 Hz
Bit 6	0x00000040	Hz890_1375	Frequency band 890-1375 Hz
Bit 5	0x00000020	Hz690_1110	Frequency band 690-1110 Hz
Bit 4	0x00000010	Hz500_890	Frequency band 500-890 Hz
Bit 3	0x00000008	Hz360_690	Frequency band 360-690 Hz
Bit 2	0x00000004	Hz220_500	Frequency band 220-500 Hz
Bit 1	0x00000002	Hz110_360	Frequency band 110-360 Hz
Bit 0	0x00000001	Hz0_220	Frequency band 0-220 Hz

Bit14-31 are unused and should be zero.

© Copyright 2001-2021 Bosch Security Systems

2.59 CONF_AUPROC_CONFIG

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a7a	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command configures the audio alarm analysis. The analysis is based on a 8kHz signal (if the device supports higher sampling rates the signal is downsampled). The analysed frequency bands (there are 13 frequency bands) have the following center frequencies:

100 Hz, 220 Hz, 350 Hz, 500 Hz, 680 Hz, 880 Hz, 1.1 kHz, 1.4 kHz, 1.7 kHz, 2.0 kHz, 2.4 kHz, 2.9 kHz, 3.4 kHz

The freuqnecy bands are overlapping triangles.

Payload Structure

16	32
Version 4 Bytes	
Trigger Level... 16 Bytes	
Trigger Level ...	
Trigger Level ...	
Trigger Level ...	
Sensitivity... 16 Bytes	
Sensitivity ...	
Sensitivity ...	

© Copyright 2001-2021 Bosch Security Systems

Sensitivity ...		
Reserved 1 Byte	**Flags** 1 Byte	**Reserved** 2 Bytes
8		24

Version

Unused (should be zero).

Trigger Level

16 Values between 0 and 255 (1 Byte each), setting the fixed threshold for every frequency band and energy level.

First Byte is for the lowest frequency and the last for the energy level. Trigger level 0 indicates that this frequency band is set off.

14 bytes (13 bands plus entire energie) are used in this version (refer to AUPROC_ALARM command).

Sensitivity

16 Values between 1 and 100 (1 Byte each), setting the sensitivity for every frequency band and energy level.

First Byte is for the lowest frequency and the last for the energy level.

14 bytes (13 bands pus entire energie) are used in this version.

The sensitivity is based on an adaptive threshold and a sliding window.

Flags

1 byte reserved for flags

Values:

	Mask	Name	Description
Bit 7	0x80	OnOff	Indicates if audio alarm is on or off

Reserved

© Copyright 2001-2021 Bosch Security Systems

2.60 CONF_AUPROC_MELPEGEL

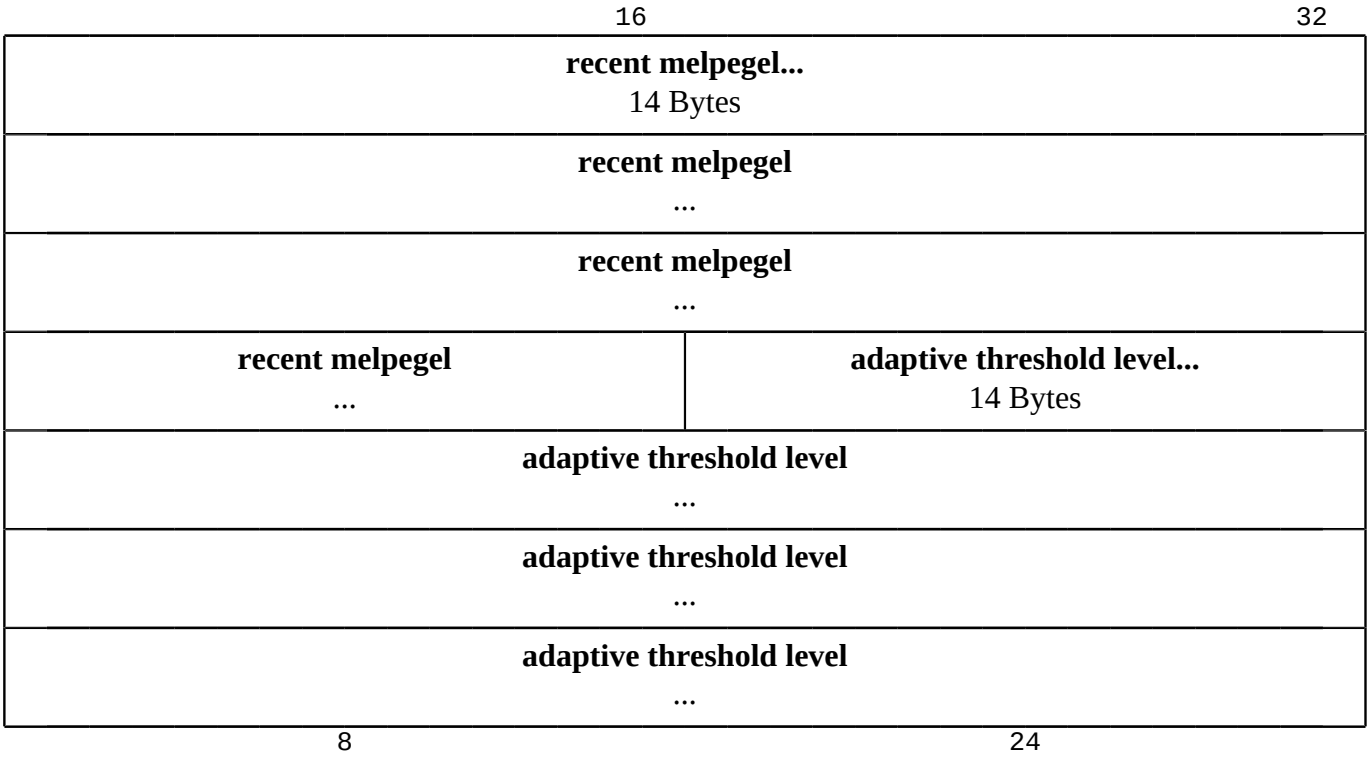
[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a7b	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the recent pegel in the frequency bands: detailed decription	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Each of the 14 bytes corresponds to the frequency bands described within the command AUPROC_ALARM and AUPROC_MELPEGEL.

Payload Structure



© Copyright 2001-2021 Bosch Security Systems

2.61 CONF_AUPROC_NAME

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a7c	audio line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	-	
Write	p_unicode	service	-	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.62 CONF_AUTO_DISCONNECT_TIME

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x030d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	0=no auto disconnect; 0<Time in seconds to disconnect	
Write	t_dword	service	0=no auto disconnect; 0<Time in seconds to disconnect	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.63 CONF_AUTO_TRACKER_TRACK_OBJECT

[API: autotracker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b2d	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the object id of the currently tracked object. Zero is returned if no object is tracked.	
Write	t_dword	user	Send the object id to start tracking	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.64 CONF_AUTO_TRACKER_TRACK_OBJECT_POS

[API: autotracker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c28	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user	Send an image position to start tracking the nearest object. Position is in normalized coordinate system with upper left corner (0x0000 0x0000) and lower right corner (0x8000 0x8000)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.65 CONF_AUTODETECT_REPLY_GROUP

[API: rcp.discovery]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0956	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the multicast group to which (when set) the VJ will listen for multicast autodetect requests	
Write	t_dword	service	Get the multicast group to which (when set) the VJ will listen for multicast autodetect requests	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

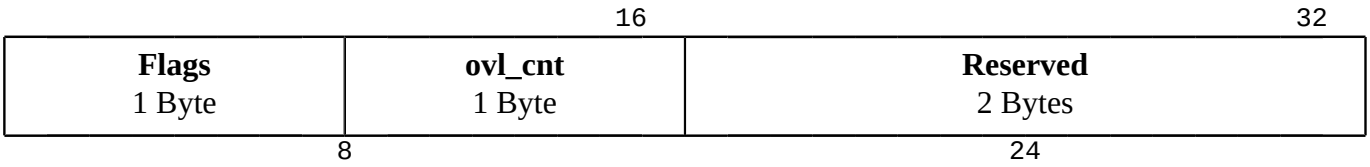
© Copyright 2001-2021 Bosch Security Systems

2.66 CONF_AUXILIARY_POWER

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c55	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns status and availability of auxiliary power, see detailed description	
Write	t_dword	service	Enables or disables auxiliary power, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Flags

	Mask	Name	Description
Bit 3	0x08	ovl	Indicates if auxiliary power is overloaded (read only, write will be ignored)
Bit 2	0x04	has ovl	Indicates if the unit has overload detection (read only, write will be ignored)
Bit 1	0x02	ena power	Indicates if auxiliary power is switched on
Bit 0	0x01	aux power	Indicates if the unit has auxiliary power (read only, write will be ignored)

ovl_cnt

Counts up if an overload occurs

Values:

no overload	0
nr of overload events	1 - 254
more than 254 overload events	255

© Copyright 2001-2021 Bosch Security Systems

2.67 CONF_BACKUP

[API: backup]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af4	cam (backup from device), trackID (backup from vrm)	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Starts an mp4 backup file of the specified replay range and store it on a account. The CONF_BACKUP_STATUS message is sent periodicallay to inform about the backup progress. The command returns an 1 byte backup id that can be used to identify the backup in the message. The backup may include signature generation and video authenticity checks. The check results can be return by an optional addon in CONF_BACKUP_STATUS message. The backup destination can be a dummy backup, the data won't be stored anywhere in that case. The dummy backup is used on the special account number 0xff (255). This is usefull, if the backup is just just for authenticity check of records.

Payload Structure

16 32

From2000 4 Bytes			
To2000 4 Bytes			
Account 4 Bytes			
RecordingIndex 4 Bytes			
RemoteCamera 1 Byte	**Transcode** 1 Byte	**TranscoderPreset** 1 Byte	**Reserved** 1 Byte
ReplaySpeed 4 Bytes			
ObjectID 4 Bytes			

© Copyright 2001-2021 Bosch Security Systems

Reserved 4 Bytes		
Backup filename... 16 Bytes		
Backup filename ...		
Backup filename ...		
Backup filename ...		
Flags 1 Byte	**Signature hash type** 1 Byte	**Reserved** 2 Bytes
8		24

From2000

Start time of the backup in seconds since 2000

To2000

Stop time of the backup in seconds since 2000

Account

Index of the account that should be used for backup (see CONF_ACCOUNT_SETTINGS), special account dummy backup 0xff.

RecordingIndex

Index of the recording to be backuped (0 or 1)

RemoteCamera

0 - local Camera, 1 to 4 refering to entries of CONF_ADD_REMOTE_DEVICE

Transcode

Optional field that defines if the video should be transcoded before backup (used by Transcoder Devices only)

TranscoderPreset

Optional field that defines the preset used by transcoding (used by Transcoder Devices only)

ReplaySpeed

Optional field that defines the replay speed (used by Transcoder Devices only). Use the same values as for CONF_HD_REPLAY_START

© Copyright 2001-2021 Bosch Security Systems

ObjectID

Optional field: if that Id is given a autotracker instance will be started that follows the object (used by Transcoder Devices only)

Backup filename

Backup filename: filename in arbitrary order of: %b begin date/time, %e end date/time, %c camera name, %i export job id, %f file nbr, %a alarm description, %dSubDirectory sub directory to write; optionally seperated by "_" or "-". Exampe: %b_%c would result e.g in a filename 20111017_14-29-26_camera1.mp4

Flags

Flags:

	Mask	Name	Description
Bit 1	0x02	AUTH_MSG_ADDON	Status message CONF_BACKUP_STATUS will contain a message addon containing authenticity check result information
Bit 0	0x01	CHECK_SIGNATURE	Checks the signature of the source records, the generation of a signature of the backup will fail, if the check fails

Signature hash type

Activates the generation of a signature file using the https certificate, works for ftp and dropbox backup

Types:

NO_SIGNATURE	0	Signature generation disabled
SHA1	1	Generates a signature file over the backup file using SHA1
SHA256	2	Generates a signature file over the backup file using SHA256

© Copyright 2001-2021 Bosch Security Systems

2.68 CONF_BACKUP_MAX_KBPS

[API: backup]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af5	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the maximum datarate for the backup (in kbps)	
Write	t_dword	user	Sets the maximum datarate for the backup (in kbps)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.69 CONF_BACKUP_RECORDING_STATUS

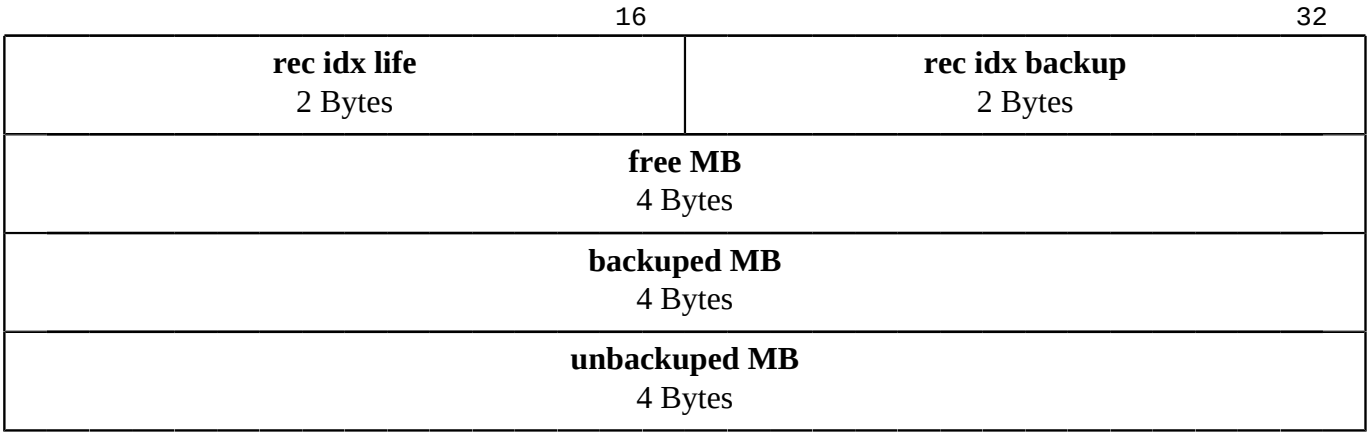
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb4	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Get the backup recording status, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

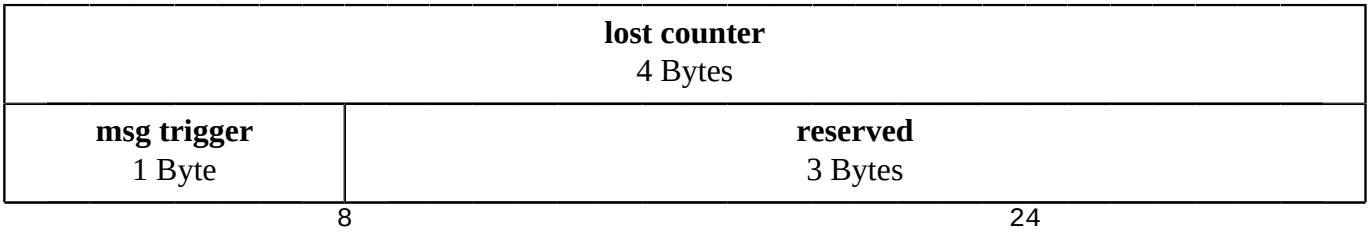
Description

This command returns the backup recording status on a line, specified by the num parameter (1 ... n). First it looks for a valid configuration of backup recording. Valid means, there exists a running recording configured to record the life data and a second recording for backup, which copies the data from the life data recording storage to it's recording storage on alarm or continuously. If both "rec idx life" and "rec idx backup" are none zero, there is a valid configuration. Then the following fields inform about the status and progress of the backuped recording. The utilization of the life data recording storage can be calculated in percent with the equation: utilization = ("unbackuped MB" * 100)/("unbackuped MB" + "backuped MB" + "free MB"). The status will be send as message, if the utilization will cross some thresholds. There are two thresholds, the "All Clear" threshold and the "Warning" threshold. These thresholds can be configured with the command CONF_BACKUP_RECORDING_STATUS_MSG_THRESHOLD (see rcp documentation). Another reason for sending this message will be on deleting or overwriting of records, which had to be backuped but weren't backuped before deletion, or were deleted while still within the maximum retention time range. Every time this event happens, the lost counter will be incremented. This counter will count since device boot.

Payload Structure



© Copyright 2001-2021 Bosch Security Systems



rec idx life

Recording index of the recording, which is configured for life data recording

Values:

None	0
Primary	1
Secondary	2

rec idx backup

Recording index of the recording, which is configured for data backup of the life data recording

Values:

None	0
Primary	1
Secondary	2

free MB

Free Storage space in MB for the life data recording

backuped MB

Storage space in MB with records of the life data recording for this line, which are already backuped or which are not required to be backuped

unbackuped MB

Storage space in MB with records of the life data recording for this line, which have to be backuped or potentially required to be backuped, because they are in range of the max pre alarm time

lost counter

Counter for events of records, which had to be backuped and that couldn't be backuped, on deleting or overwriting by newer records

msg trigger

Reason for sending this status

© Copyright 2001-2021 Bosch Security Systems

Values:

REQUESTED	0	Response on request
ALL_CLEAR	1	All clear threshold passed
WARNING	2	Warning threshold passed
DATA_LOSS	3	New data loss

© Copyright 2001-2021 Bosch Security Systems

2.70
CONF_BACKUP_RECORDING_STATUS_MSG_THRESHOLD

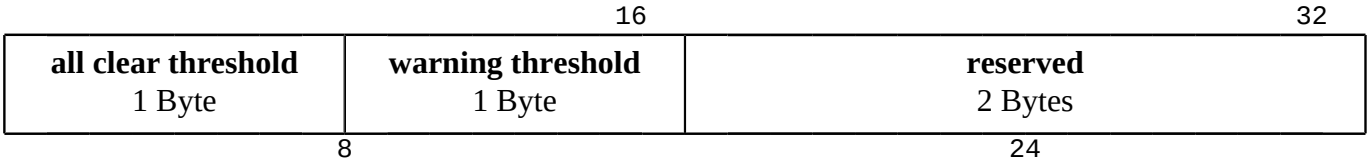
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb5	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Get the thresholds, see detailed description	
Write	p_octet	service	Set the thresholds, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command configures the thresholds for the CONF_BACKUP_RECORDING_STATUS message. The message will be send, when the storage space utilization will cross the "all clear" threshold by changed utilization from higher to lower utilization or when crossing the "warning" threshold by changed utilization from lower to higher utilization.

Payload Structure



all clear threshold

Threshold for all clear in percent

warning threshold

Threshold for warning in percent

© Copyright 2001-2021 Bosch Security Systems

2.71 CONF_BACKUP_STATUS

[API: backup]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af8	Backup Session ID (if set to 0 a list of all backup sessions is returned)	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List the status of a backup sessions:	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command returns a list of the current backups or the status of one single backup if a specific valid backup id was provided. The list contains the following sequence:

Payload Structure

16	32
Backup Session ID 4 Bytes	
RCP Session ID 4 Bytes	
Start Time 4 Bytes	
End Time 4 Bytes	
Transfered Bytes 8 Bytes	
Transfered Bytes ...	
File Count 2 Bytes	**File Nbr** 2 Bytes
File part 2 Bytes	**Reserved** 2 Bytes

© Copyright 2001-2021 Bosch Security Systems

Current time 4 Bytes			
Ppm Current File 4 Bytes			
Backup Errrors 4 Bytes			
RTP packet loss 4 Bytes			
MP4 box errors 4 Bytes			
Account errors 4 Bytes			
Cam 2 Bytes		**RecIdx** 2 Bytes	
Account 1 Byte	**Backup ID** 1 Byte	**Timezone quarter hours** 1 Byte	**Reserved** 1 Byte
Link... 128 Bytes			
Link [...]			
Link ...			
Reserved... 8 Bytes			
Reserved ...			
8		24	

Backup Session ID

Backup Session ID

RCP Session ID

Corresponding RCP Session ID

Start Time

Start Time of the Backup in Seconds since 2000

© Copyright 2001-2021 Bosch Security Systems

End Time

End Time of the Backup in Seconds since 2000

Transfered Bytes

Transfered Bytes

File Count

Number of files to backup

File Nbr

Current file to backup

File part

If a larger Backup file is split into multiple parts this value tells the current part

Current time

The current time of the backup

Ppm Current File

Status in ppm of the current backup

Backup Errrors

The last account error.

Errors:

ERROR	2	Error during backup
CANCELLED	4	Backup canceled
START_FAILED	5	Start of backup failed

RTP packet loss

Number of lost packets

MP4 box errors

Errors in the mp4 converter.

Account errors

The last account error.

Errors:

NO_ERROR	0
NO_IP	1

© Copyright 2001-2021 Bosch Security Systems

CONNECT_FAILED	2
SETDIR_FAILED	3
WRITE_FAILED	4

Cam

Camera

RecIdx

Recording Index

Account

The index of the account used for backup (see CONF_ACCOUNT_SETTINGS and CONF_BACKUP). For backup via http the account is set to 0.

Backup ID

The backup id that is returned by the CONF_BACKUP command. This id can be used to link the backup started by CONF_BACKUP with this status message

Timezone quarter hours

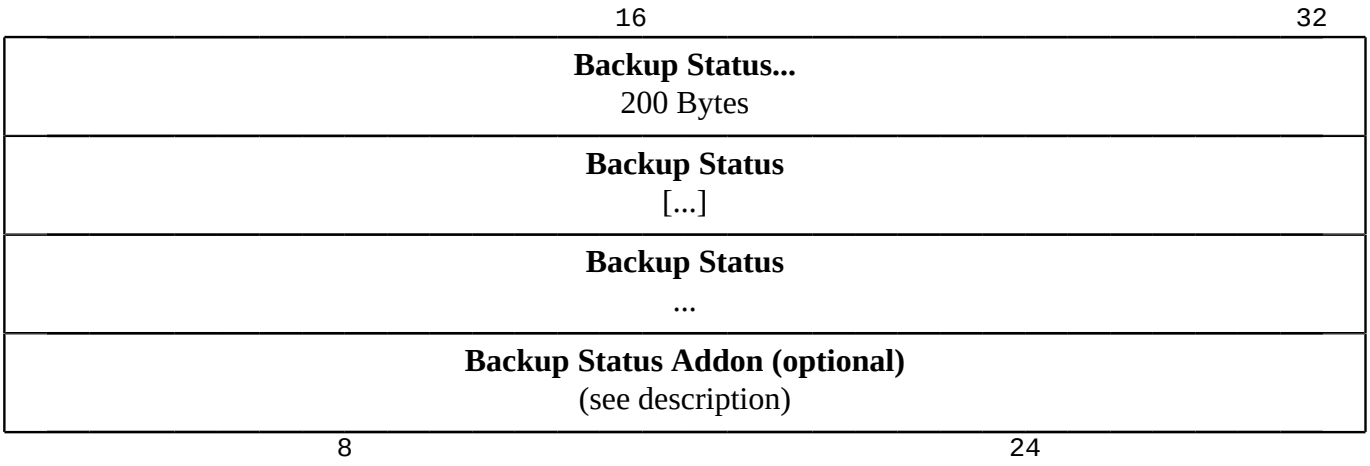
Signed time zone offset in quarter hours

Link

A direct link to the backup mp file. The message contains the link if it is available

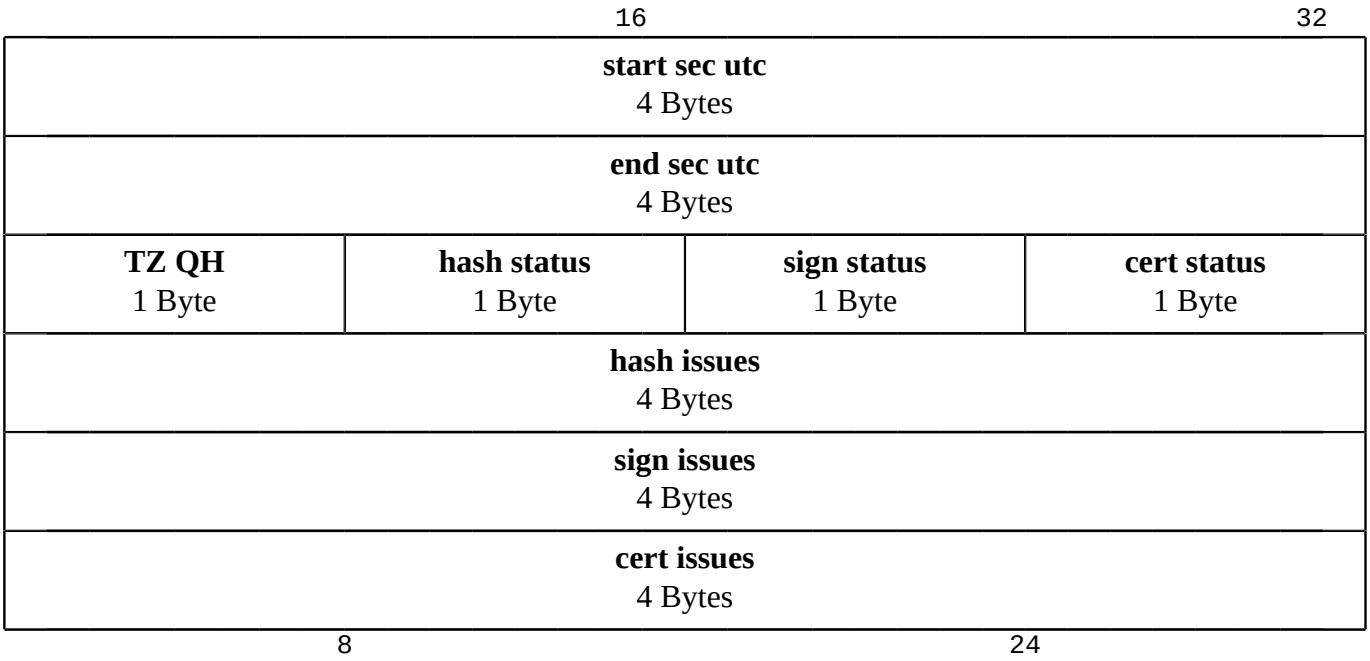
Message Payload Structure

This command is also send as message by an running backup with same payload and an optional addon containing authenticity informations if the check for it and the addon was activated in the backup job



© Copyright 2001-2021 Bosch Security Systems

Backup Status Addon (optional)



start sec utc

Start time in seconds since 2000 utc of the sequence

end sec utc

Last time in seconds since 2000 utc of the sequence

TZ QH

Time time zone offset in quarter hours (7 lowest bits), and sign (highest bit)

hash status

Hash status for authenticity check

Values:

missing	0
not checked	1
invalid	2
valid	3

sign status

Signature status

Values:

missing	0
not checked	1

© Copyright 2001-2021 Bosch Security Systems

missing certificate	2
invalid	3
valid	4

cert status

Signing certificate status

Values:

not checked	0
unknown	1
invalid	2
trusted	3
owned	4

hash issues

Count of hash issues occured since start of authenticity check

sign issues

Count of sign issues occured since start of authenticity check

cert issues

Count of cert issues occured since start of authenticity check

© Copyright 2001-2021 Bosch Security Systems

2.72 CONF_BACKUP_STOP

[API: backup]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b58	Backup Session ID	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	user	Cancels an ongoing backup job	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.73 CONF_BASE_POE_CLASS

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	service	Read the base poe class of the device (w/o considering power adder); 1: class0, 2: class1, 3: class2, etc. 255: no PoE	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.74 CONF_BEST_FACE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b6e	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Allows the upload of detected faces to an ftp server or drop box account.

Payload Structure

16			32
enable 1 Byte	**reserved** 1 Byte	**accountId** 1 Byte	**pictureFormat** 1 Byte
targetObjectWidth 4 Bytes			
time out 1 Byte	**reserved...** 11 Bytes		
reserved ...			
reserved ...			
8		24	

enable

best face is disabled	0
best face is enabled	1

accountId

Select account which is used as target for the detected faces for detailed information’s see CONF_ACCOUNT_SETTINGS

© Copyright 2001-2021 Bosch Security Systems

pictureFormat

Select file format:

Jpeg	0
YUV420	1
Uncompressed tiff	2

targetObjectWidth

Configure maximum face width in pixel 0 means auto no scaling is necessary delivers best performance

time out

Time out in seconds. If the time out is set to 0 the best face of the tracking process will be posted when the face has left the scene.

Defines a timeout for the maximum delay which is introduced till a face is posted. After a face is posted the tracking is restarted.

Example:

Face is in scene for 50 sec t = [0s,50s]

timeout is configured to 10 s

best face of [0s,10s] is posted

best face of [10s,20s] is posted

best face of [20s,30s] is posted

best face of [30s,40s] is posted

best face of [40s,50s] is posted

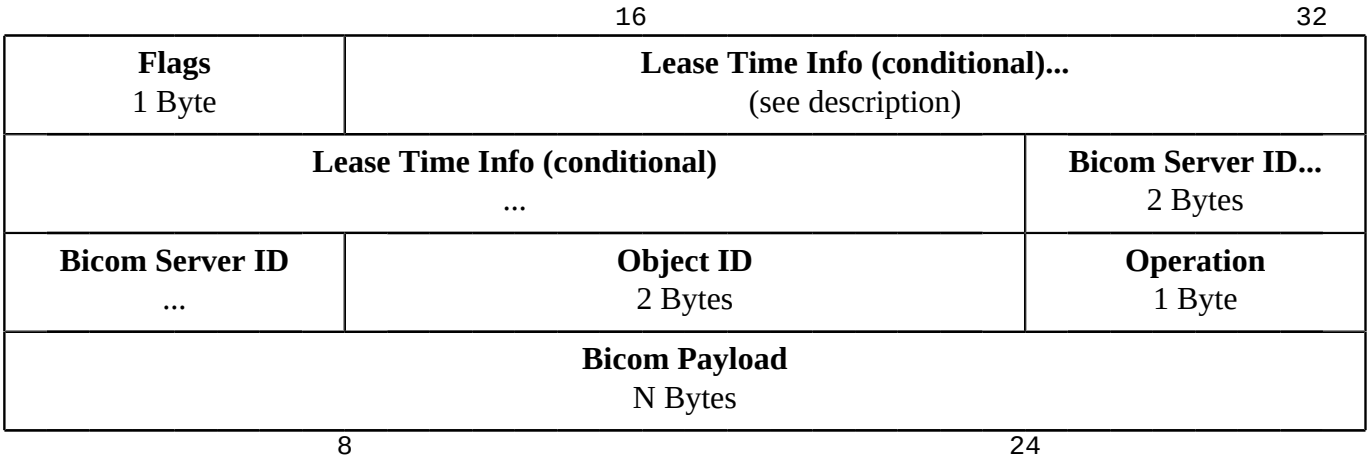
© Copyright 2001-2021 Bosch Security Systems

2.75 CONF_BICOM_COMMAND

[API: bicom]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09a5	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user	Sends a BICOM read or write command to the local camera frontend. Notice: for some BICOM commands an access level higher than 'l_user' is needed. For details about this see Appendix 'Bicom Command Access Levels' of this document.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Flags

Transmission-Flags

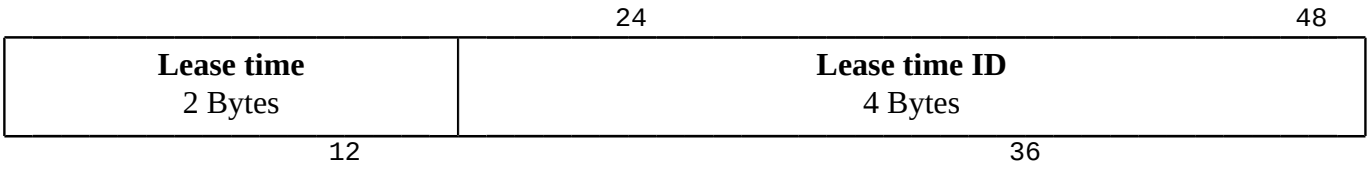
	Mask	Name	Description
Bit 7	0x80	Flags Available	Must be always set to 1
Bit 6	0x40	unsued3	unused set to 0
Bit 5	0x20	unused2	unused set to 0
Bit 4	0x10	unused1	unused set to 0
Bit 3	0x08	Lease Time Available	Set to 1 if a lease time is included in the request

© Copyright 2001-2021 Bosch Security Systems

Bit 2	0x04	Native Errors	Set to 1 to receive the native BICOM errors
Bit 1	0x02	obsolete2	obsolete, don't care
Bit 0	0x01	obsolete1	obsolete, don't care

Lease Time Info (conditional)

Lease Time Info is available when corresponding flag is set.



Lease time

Time period in seconds the access should be blocked for other clients.

Note: This field has to be signaled in little endian mode.

Lease time ID

Random number generated by the client. If a lease time > 0 is provided with the first access, further accesses during the lease time are only possible if the same lease time id is provided.

Bicom Server ID

Server ID, e.g. 0x0002 for "Device Server" (See BICOM application documentation)

Object ID

Object ID, e.g. 0x0100 for "Type" (See BICOM application documentation)

Operation

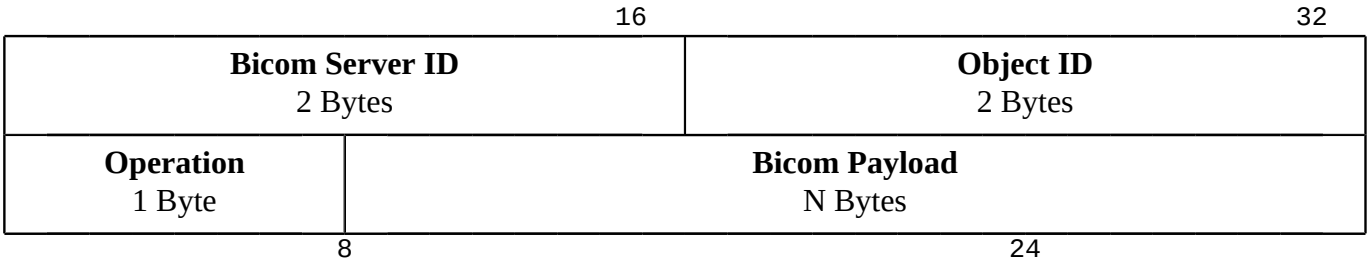
See BICOM application documentation for details.

Values

GET	0x01
SET	0x02
SET_GET	0x03
INC	0x04
INC_GET	0x05
DEC	0x06
DEC_GET	0x07
SET_DFLT	0x08
SET_GET_DFLT	0x09

Message Payload Structure

© Copyright 2001-2021 Bosch Security Systems



Bicom Server ID

Server ID, e.g. 0x0004 for "Camera Server" (See BICOM application documentation)

Object ID

Object ID, e.g. 0x0190 for "Colour" (See BICOM application documentation)

Operation

See BICOM application documentation

Values:

EVENT	0x70 - 0x7F

Example

RCP Request

84	Native Errors
00 02	Server ID = 2 ("Device server")
01 50	Object "IdString"
01	Operation GET

RCP Reply

84	Native Errors
00 02	Server ID = 2 ("Device server")
01 50	Object "IdString"
01	Operation GET
00 68 00 68 00 68 00 68 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20	34 Bytes Unicode ID String

RCP Message

00 04	Server ID = 4 ("Camera server")
01 90	Object "Colour"
70	Operation EVENT
00 00	2 Byte unsigned short: Colour mode is "B/W"

© Copyright 2001-2021 Bosch Security Systems

2.76 CONF_BOARD_RESET

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0811	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Reset the board	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.77 CONF_BOOT_DEFAULT_APP

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x099f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Boot the default application	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.78 CONF_BOOT_STATE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	always		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

booted	100
waiting for camera frontend	50
waiting for DHCP	25
Percent	0 - 100

© Copyright 2001-2021 Bosch Security Systems

2.79 CONF_BOOTLOADER_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ef	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the bootloader version	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.80 CONF_BROWSER_DATETIME_FORMAT_VAL

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01e9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read browser date/time format	
Write	t_octet	service	Set browser date/time format (1=Europe, 2=USA, 3=Japan)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.81 CONF_BROWSER_LANGUAGE_VAL

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01e3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	always	Read current set browser language	
Write	t_octet	service	Set browser language (numbering depends on toolkit implementation)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.82 CONF_BUFFERED_RECORDING_MODE

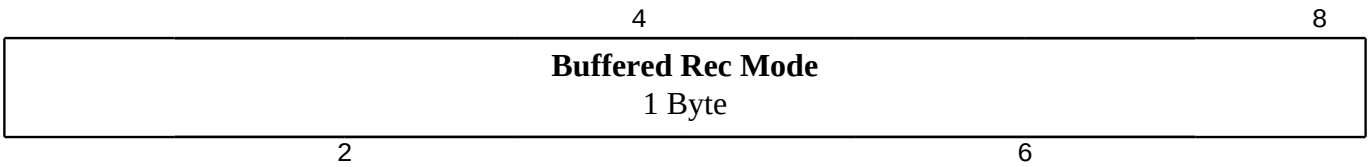
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bcd	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Get the buffered recording mode: 0 off, 1 on	
Write	t_octet	service	Set the buffered recording mode: 0 off, 1 on,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to configure the buffered recording configuration mode. The default is the manual mode, which is used for standard (single/dual) recording or manually configured buffered recording, which means the primary and secondary recording will be setup to do buffered recording. In case of buffered recording configuration mode, the device recording configuration will be changed internally to an buffered recording setup. The recording will then be configured like a single primary recording via recording profiles but internally the configuration is mapped to a coresponding buffered recording setup. When set to this mode, the secondary recording isn't accessable anymore via recording profile configuration (affected comands: CONF_HD_RECORD_PROFILE, CONF_HD_RECORD_PROFILE_V2, CONF_HD_RECORD_PROFILE_SECONDARY, CONF_HD_PROFILE_V2_SECONDARY) and start/stop configuration(affected commands: CONF_START_RECORD, CONF_START_SPAN_RECORD, CONF_HD_MGR_START, CONF_HD_MGR_START_SECONDARY, CONF_HD_MGR_STOP, CONF_HD_MGR_STOP_SECONDARY). When the mode is changed, the actual recording is stopped.

Payload Structure



Buffered Rec Mode

Manual configuration	0
Buffered recording configuration mode	1

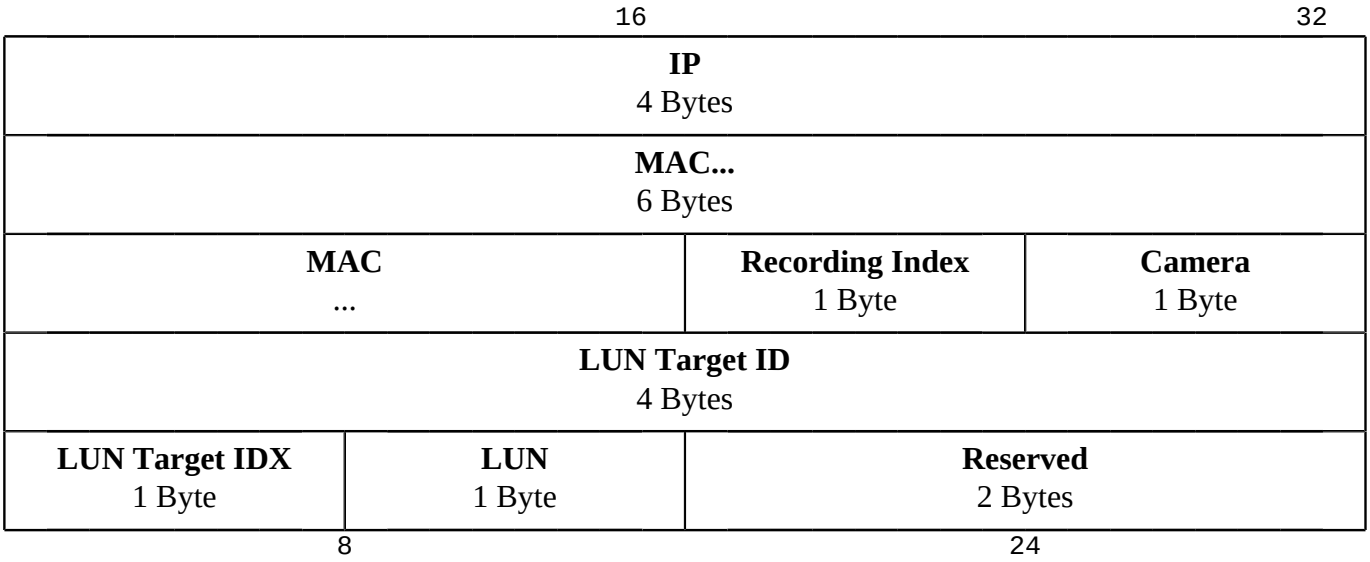
© Copyright 2001-2021 Bosch Security Systems

2.83 CONF_CAM_REC_SPANS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a8f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Retrieve the list of recording spans of a camera from a span manager on a managed lun.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



IP

The ip address of the recording device.

MAC

The hardware address of the recording device. This field may be zero if you search for recordings without considering the mac.

Recording Index

Primary Recording	1
Secondary Recording	2

© Copyright 2001-2021 Bosch Security Systems

Camera

The camera index.

LUN Target ID

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

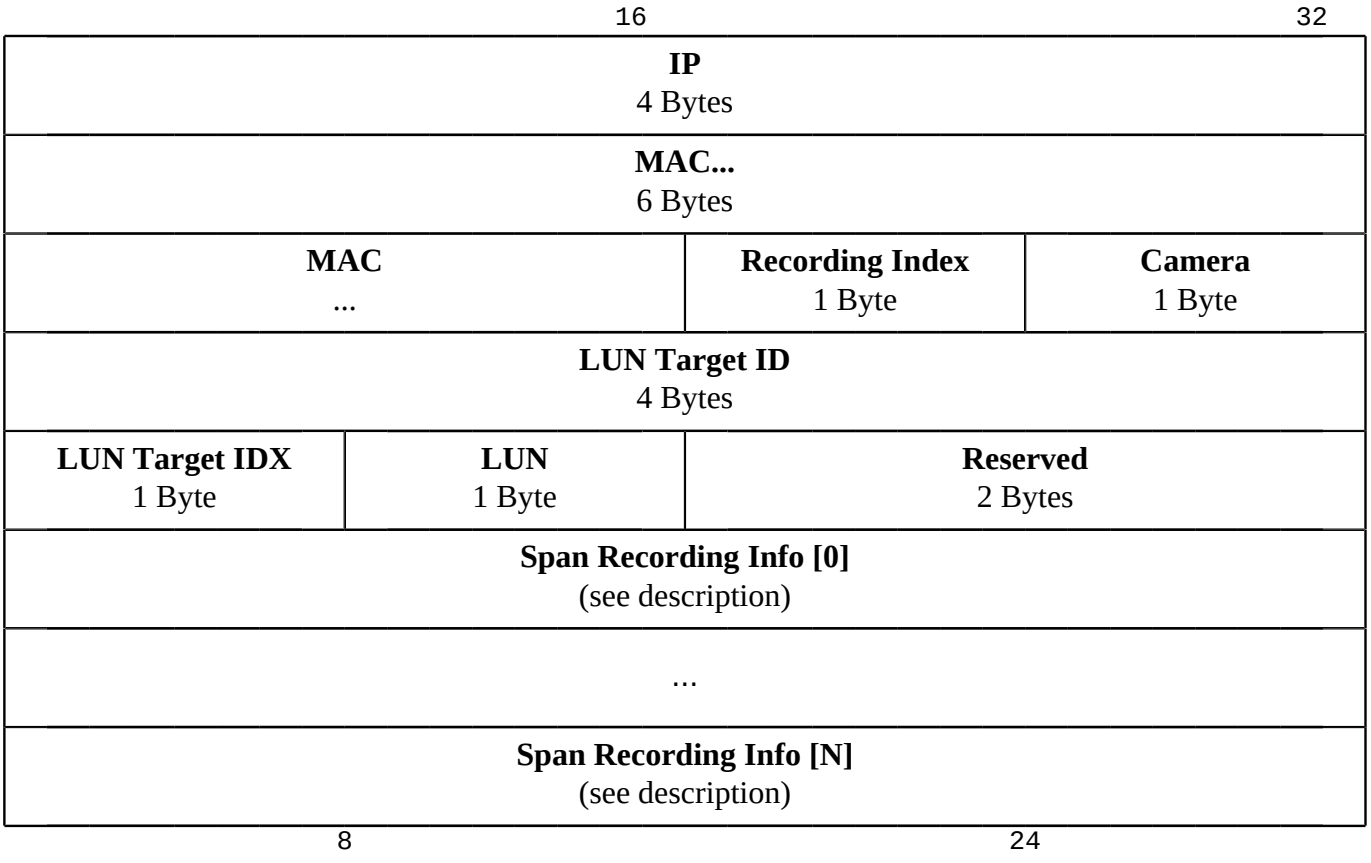
LUN Target IDX

The index of the target.

LUN

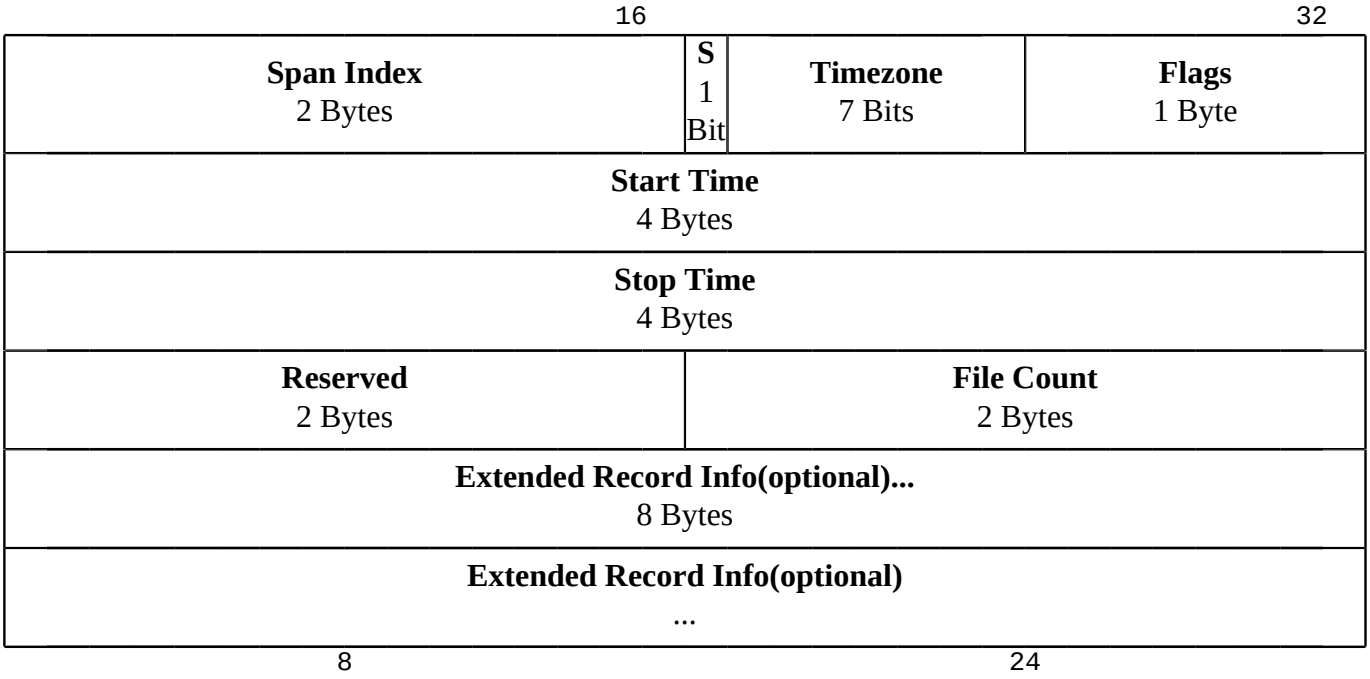
The logical unit number.

Response/Message Payload Structure



© Copyright 2001-2021 Bosch Security Systems

Span Recording Info



Span Index

The span index in the current lun.

S

The timezone sign.

Timezone

Timezone in quarter hours.

Flags

Values:

	Mask	Name
Bit 7	0x80	extended span rec info
Bit 6	0x40	virtual alarm
Bit 5	0x20	video loss
Bit 4	0x10	motion alarm
Bit 3	0x08	input alarm
Bit 2	0x04	reserved (former migrated recording)
Bit 1	0x02	Recording Running
Bit 0	0x01	Continous Recording

© Copyright 2001-2021 Bosch Security Systems

Start Time

Seconds since 2000.

Stop Time

Seconds since 2000.

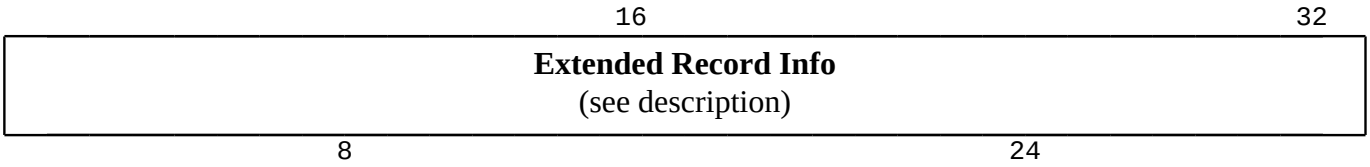
Reserved

File Count

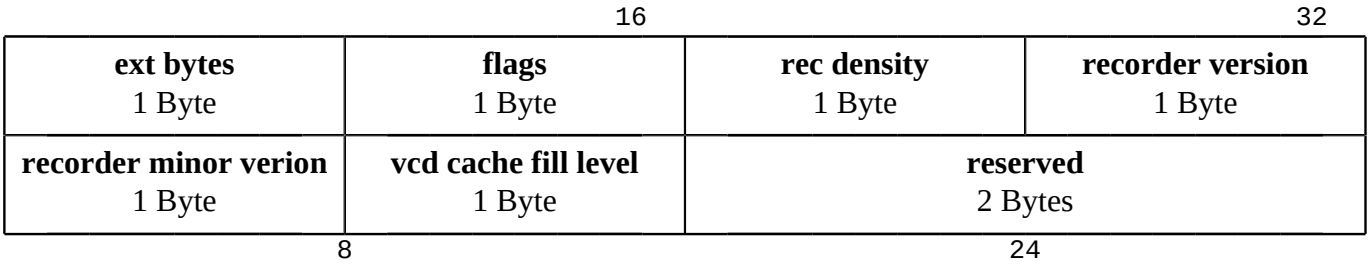
The number of files that are stored in this span.

Extended Record Info(optional)

Additional info, only present, if flag for extended span rec info is set.



Extended Record Info



ext bytes

Size of extension bytes (actual always 8 bytes)

flags

Values:

	Mask	Name
Bit 5	0x20	meta in file
Bit 4	0x10	audio in file
Bit 3	0x08	last file time record
Bit 2	0x04	last file alarm record
Bit 1	0x02	first file time record
Bit 0	0x01	first file alarm record

© Copyright 2001-2021 Bosch Security Systems

rec density

Always 0

recorder version

FW version of the recorder.

recorder minor verion

FW minor of the recorder.

vcd cache fill level

Vcd cache fill level in percent

reserved

Reserved.

© Copyright 2001-2021 Bosch Security Systems

2.84 CONF_CAMERA_CALIBRATION

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c34	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Calculates calibration with calibration elements, payload is beyond the scope of this document (for more details read CameraCalibrationConfig.doc)	
Write	p_octet	iva	Calculates calibration with calibration elements, payload is beyond the scope of this document (for more details read CameraCalibrationConfig.doc)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.85 CONF_CAMERA_LENS_CURVE

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be6	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command allows to query the lens transfer function of the camera. This command is not supported for all cameras. In the case of CONF_DEVICE_CAPABILITIES -> PTZ_ON_CLIENT_TAG = TRUE A lens curve must be present.

```
Coordinate System [Picture]h
(x = 0,y = 0)     ----------------------------------- (x = 0, y =
 32768)
...
...                        (x = 16384, y = 16384)
...
(x = 0,y = 32768) ----------------------------------- (x = 32768,y =
 32768)
```

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

© Copyright 2001-2021 Bosch Security Systems

Length

Length of tagged value including length and tag field

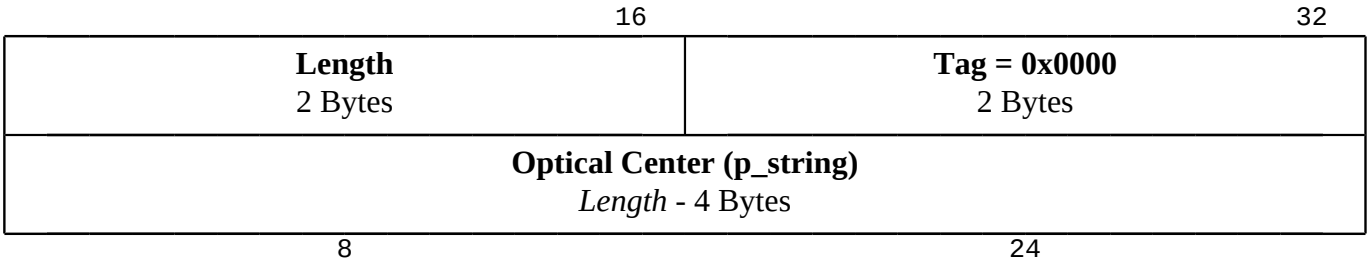
Tag

Tag specifying the encoding and meaning of the value

Tag 0: Optical Center

Defines the horizontal and vertical position of the optical axis of the lens within the sensor plane (full viewed image) in the normalized coordinate system defined in the previous description. This will be called optical center below.

An offset of (16384; 16384) therefore means that the optical center of the circular projection is horizontally and vertically centered in the sensor image (should be the normal case).



Format of Payload String

x;y

	Max. Length	Limits	Description
x	5	[0,32768]	Specify the x position of the optical center
y	5	[0,32768]	Specify the y position of the optical center

Example

```
x;y = "16384;16384"
-> x = 16384, y = 16384 -> lens is centered
```

Tag 1: Lens transfer function

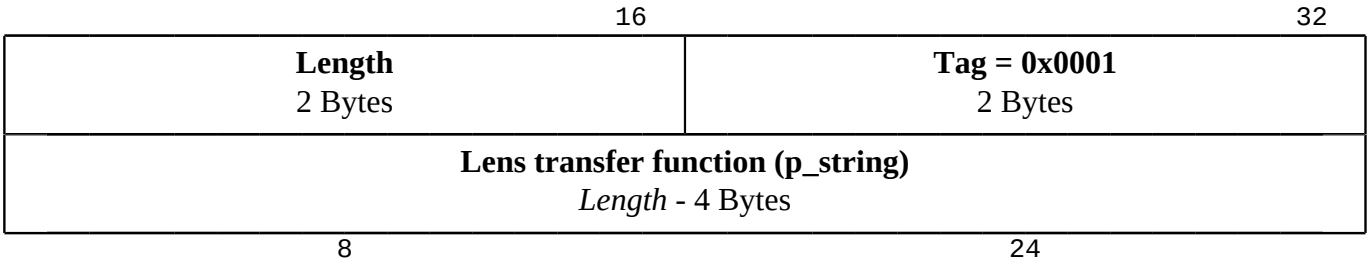
The lens curve is assumed to be rotation symmetric.

Specifices a tuple for a set of points (radius on the sensor, the same coordinate system is used as before and a angle of incidence relativ to the optical axis).

Optimal interpolation between the points can be achived with natural splines.

© Copyright 2001-2021 Bosch Security Systems

For a radius bigger than the last point the behavoiur of the lens is undefined (should be masked out in black on the client)



Format of Payload String

radius[0];angle[0];radius[1];angle[1]

	Max. Length	Limits	Description
radius	5		Specify the radius in the previous defined normalized coordinate system
angle	3.6		Specify the angle in degrees

Example

```
lens transfer function =
 "7189;20.469667;13815;40.049053;17223;50.729707;20151;60.518720;21646;65.859
-> p[0] = (radius = 7189, angle of incidence = 20.469667°)
-> p[1] = (radius = 13815, angle of incidence = 40.049053°)
-> p[2] = (radius = 17223, angle of incidence = 50.729707°)
-> p[3] = (radius = 20151, angle of incidence = 60.518720°)
-> p[4] = (radius = 21646, angle of incidence = 65.859047°)
```

© Copyright 2001-2021 Bosch Security Systems

2.86 CONF_CAMERA_LOCATION_METADATA

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bfa	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_CAMERA_LOCATION_METADATA allows to configure/retrieve several descriptive location metadata fields, e.g. City, Country Code. The command has an internal tagged structure.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

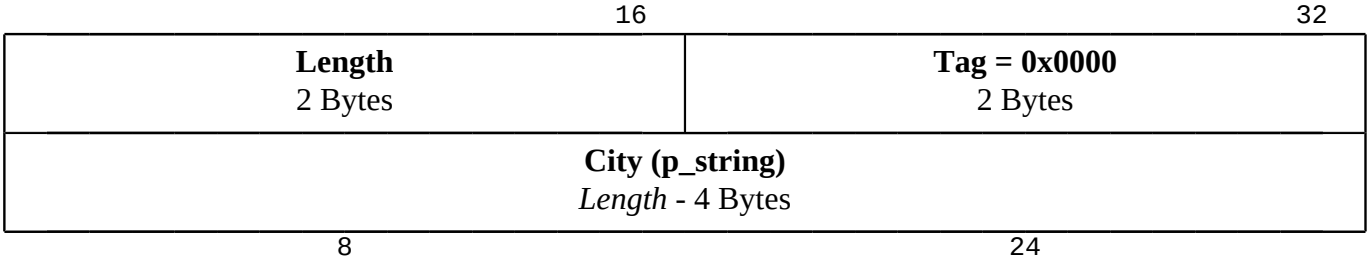
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

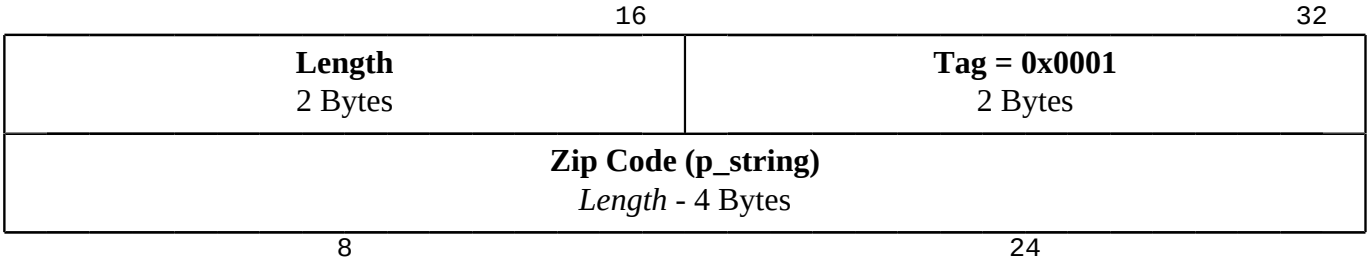
Tag 0: City

Contains the city name as string, no zero termination.



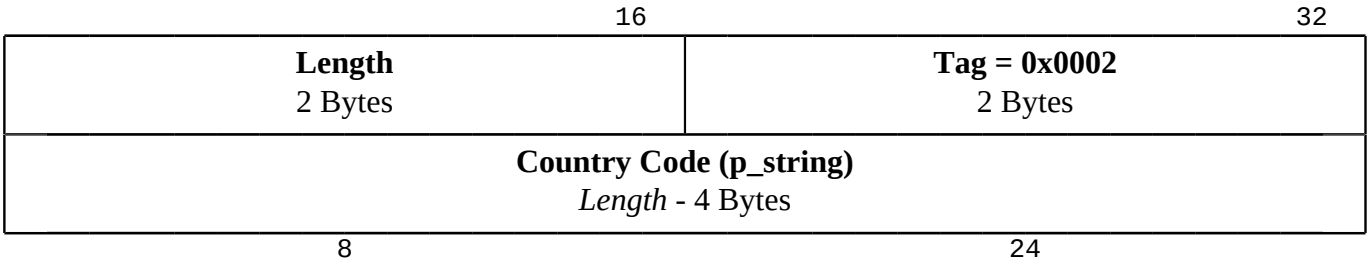
Tag 1: Zip Code

Contains the zip code as string, no zero termination.



Tag 2: Country Code

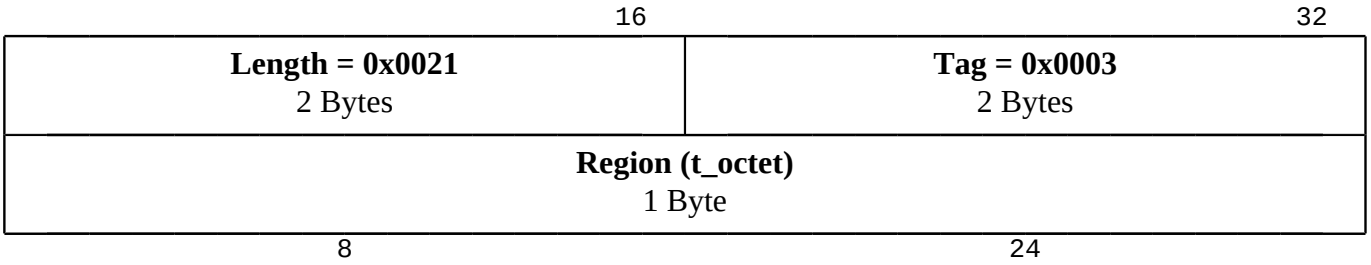
Contains the country code as string, no zero termination.



© Copyright 2001-2021 Bosch Security Systems

Tag 3: Region

Specifies the region.



© Copyright 2001-2021 Bosch Security Systems

2.87 CONF_CAMERA_ORIENTATION

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be5	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The format has an internal tagged structure, the allowed tags can be queried via CONF_CAMERA_POSITION_OPTIONS. (Support orientation calibration [0 no support, 1 pan only, 2 full support])

1: pan only: Tag 1 and 3 are supported

2: full support: Tag 1,2,3 and 4 are supported

All commands can be send in binary or string format. Only one configuration is valid.

Tag 1 (binary) and 3 (string) configure only pan angle for configuration tag 1 or 3 can used.

Tag 2 (binary) and 4 (string) configure can be used to configure the full camera orientation.

Read will always return both variant of one configuration (string + binary representation)

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

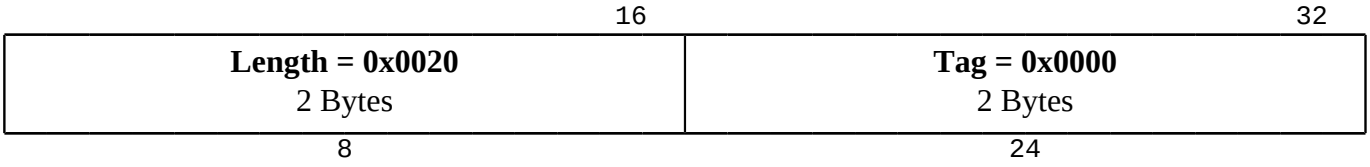
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

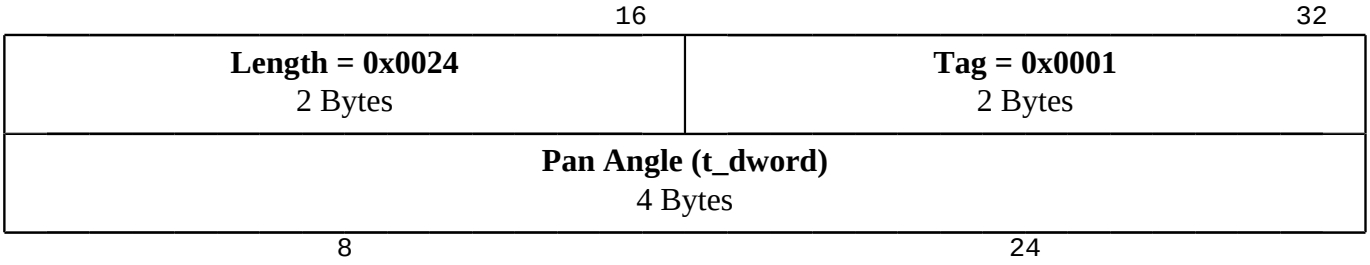
Tag 0: Undefined orientation

Default value when no camera orientation is configured, can also be used to remove the camera orientation configuration



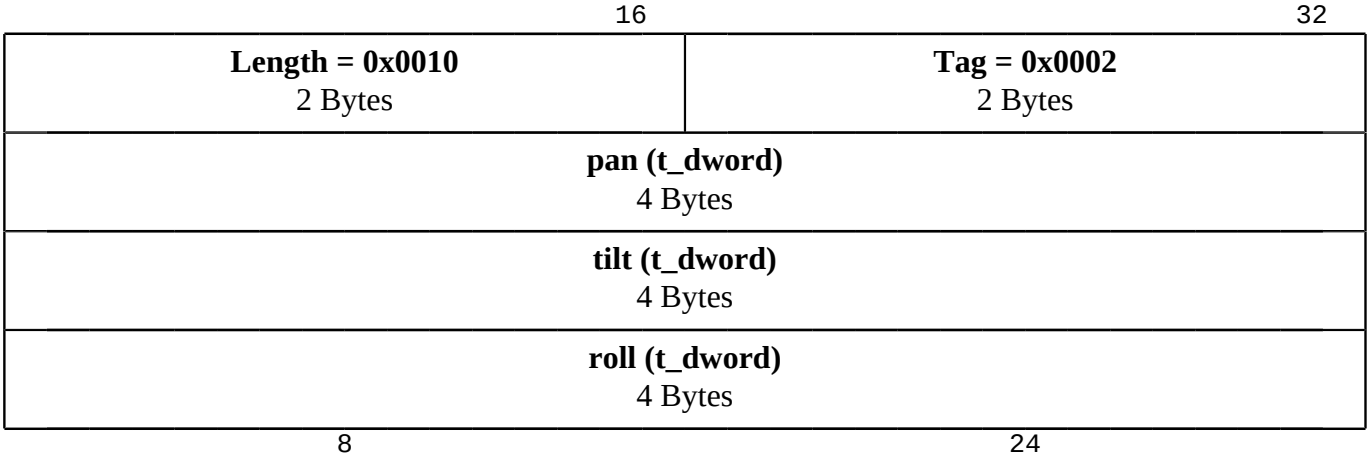
Tag 1: Pan Angle

Configure camera orientation only pan angle can be configured (legacy mode: Plane is configured via VCA config). pan u(32): Specifies the pan angle in units of (2*pi/(2^32)).



Tag 2: Full Orientation

Specify the full camera orientation at once



pan

Specifies the pan angle in units of (2*pi/(2^32))

© Copyright 2001-2021 Bosch Security Systems

tilt

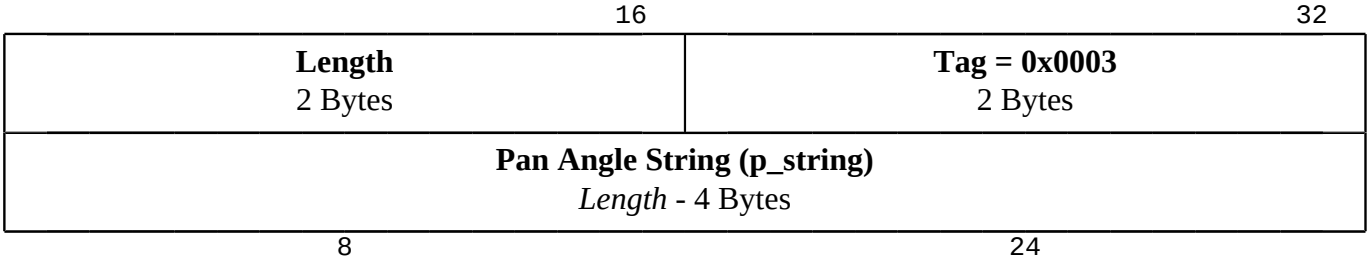
Specifies the tilt angle in units of (2*pi/(2^32))

roll

Specifies the roll angle in units of (2*pi/(2^32))

Tag 3: Pan Angle String

Specifies the pan angle (in degrees) as a string.



Format of Payload String

Pan

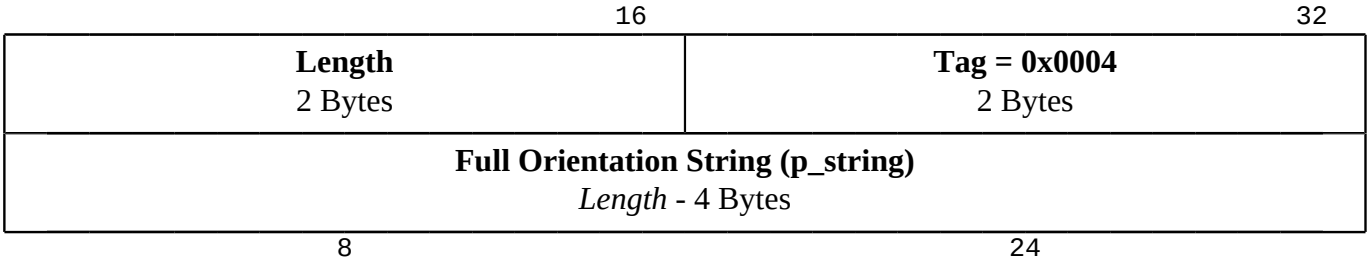
	Max. Length	Limits	Description
Pan	3.6	[-360.0,360.0]	Pan angle in degrees

Example

```
pan="3.455"
-> pan = 3.455°
```

Tag 4: Full Orientation String

Specify the full camera orientation at once



Format of Payload String

Pan;Tilt;Roll

© Copyright 2001-2021 Bosch Security Systems

	Max. Length	Limits	Description
Pan	3.6	[-360.0,360.0]	Specifies the pan angle in degrees
Tilt	3.6	[-360.0,360.0]	Specifies the tilt angle in degrees
Roll	3.6	[-360.0,360.0]	Specifies the roll angle in degrees

© Copyright 2001-2021 Bosch Security Systems

2.88 CONF_CAMERA_POSITION

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bdf	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The format has an internal tagged structure, the allowed tags can be queried via CONF_CAMERA_POSITION_OPTIONS

(Byte[0] support local cartesian coordinate system [0 no, 1 yes];) 1 -> tag 1 and 2 are supported

(Byte[1] support wgs 1984 coordinate system [0 no, 1 yes]) 1 -> tag 3 and 4 are supported

Only a single set of coordinates can be used e.g. local coordinates or global WGS coordinates it is possible to define the coordinates via a string or a binary command

Tag 1 (binary) and 3 (string) configure a local cartesian coordinate system.

Tag 2 (binary) and 4 (string) configure a wgs 1984 coordinate system.

Read will always return both variant of one configuration (string + binary representation)

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0: Undefined Position

Default value when no camera position is configured, can also be used to remove the camera position configuration

16	32
Length = 0x0020 2 Bytes	**Tag = 0x0000** 2 Bytes
8	24

Tag 1: Cartesian Position

Local cartesian camera position

16	32
Length = 0x0010 2 Bytes	**Tag = 0x0001** 2 Bytes
X (t_dword) 4 Bytes	
Y (t_dword) 4 Bytes	
Z (t_dword) 4 Bytes	
8	24

X

Specifies the x coordinate in units of 1/(2^16)

Y

Specifies the y coordinate in units of 1/(2^16)

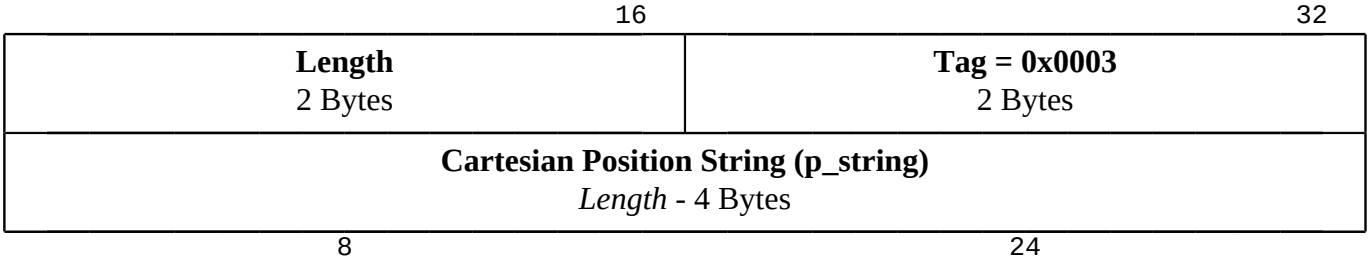
Z

Specifies the z coordinate in units of 1/(2^16)

© Copyright 2001-2021 Bosch Security Systems

Tag 3: Cartesian Position String

Local cartesian camera position



Format of Payload String

X;Y;Z

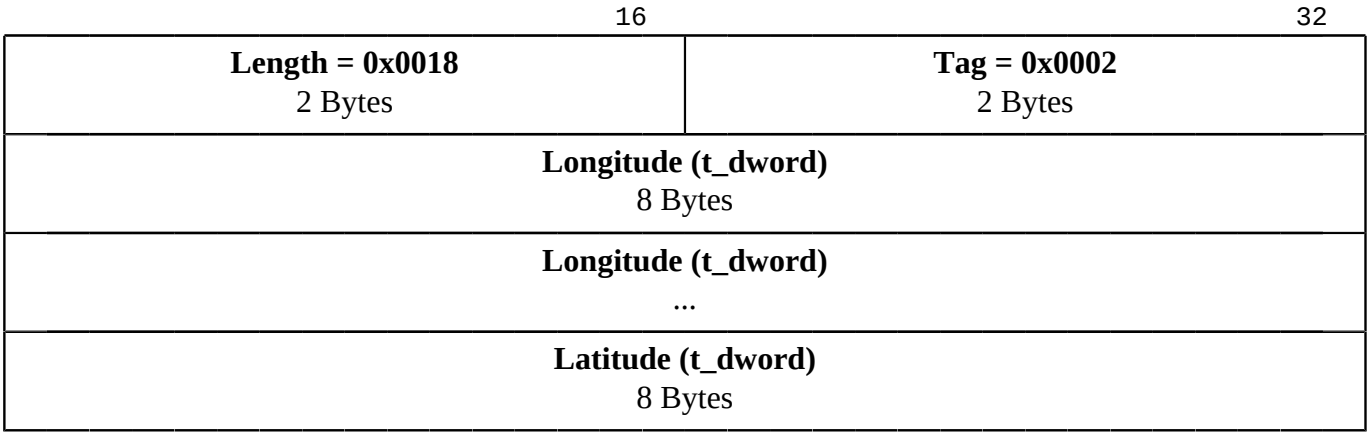
	Max. Length	Limits	Description
X	5.4	[-32767,32767]	X position in local cartesian coordinate system
Y	5.4	[-32767,32767]	Y position in local cartesian coordinate system
Z	5.4	[-32767,32767]	Z position in local cartesian coordinate system

Example

```
XYZ = "1000.124;145.3434;123.2355"
-> x= 1000.124 m, y = 145.3434 m, z = 123.2355
```

Tag 2: WGS 1984 Camera Position

Define the camera position based on wgs 1984 (World Geodetic System 1984)



© Copyright 2001-2021 Bosch Security Systems

Latitude (t_dword) ...
Height (t_dword) 4 Bytes

8 24

Longitude

Specifies the longitude in units of (2*pi/(2^64))

Latitude

Specifies the latitude in units of (2*pi/(2^64))

Height

Signed value height above sea level in units of 1/(2^16) m

Tag 4: WGS 1984 Camera Position String

Define the camera position based on wgs 1984 (World Geodetic System 1984)

16 32

Length 2 Bytes	**Tag = 0x0004** 2 Bytes
WGS 1984 Camera Position String (p_string) *Length* - 4 Bytes	

8 24

Format of Payload String

Longitude;Latitude;Height

	Max. Length	Limits	Description
Longitude	3.12	[-360.0,360.0]	Specifies the longitude in degrees
Latitude	5.4	[-90.0,90.0]	Specifies the latitude in degrees
Height	3.12	[-32767,32767]	Height above sea level in m.

Example

```
longitude;latitude;height="12.45674964;45.456464;125.1234"
-> longitude = 12.45674964°, latitude = 45.456464°, 125.1234 m
```

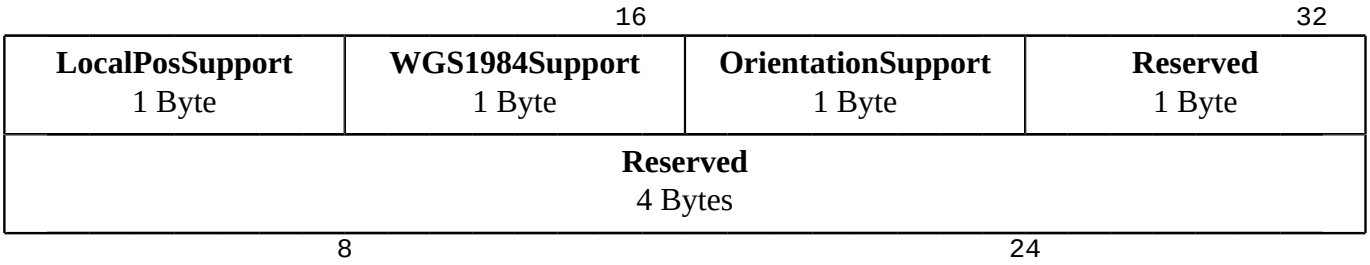
© Copyright 2001-2021 Bosch Security Systems

2.89 CONF_CAMERA_POSITION_OPTIONS

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



LocalPosSupport

Support local cartesian coordinate system

Values:

No	0
Yes	1

WGS1984Support

Support wgs 1984 coordinate system

Values:

No	0
Yes	1

OrientationSupport

Support pan, tilt, and/or roll orientation

Values:

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

2.90 CONF_CAMERA_SURROUNDING

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be1	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The format has an internal tagged structure, the allowed tags can be queried via CONF_CAMERA_SURROUNDING_OPTIONS ([0: no, 1: flat plane only elevation can be configured (plane is assumed to be parallel to the (world, local) coordinate system])

1 -> Tag 1 (binary) and 3 (string) configure a offset to the ground plane (orientation is derived from (CONF_CAMERA_ORIENTATION))

Only a single surrounding configuration it is possible to define the coordinates via a string or a binary command

Tag 1 (binary) and 3 (string) configure a elevation to the ground plane under the assumption that the ground plane is parallel to the camera surrounding defined by (CONF_CAMERA_POSITION and CONF_CAMERA_ORIENTATION) surrounding is defined by CONF_CAMERA_ORIENTATION [tilt, roll] + CONF_CAMERA_SURROUNDING [elevation]

Read will always return both variant of one configuration (string + binary representation)

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

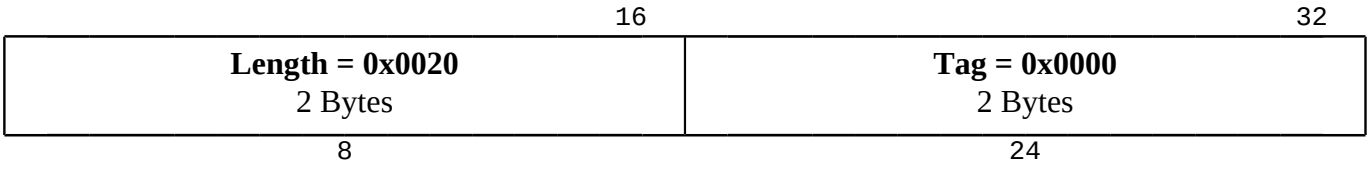
© Copyright 2001-2021 Bosch Security Systems

Tag

Tag specifying the encoding and meaning of the value

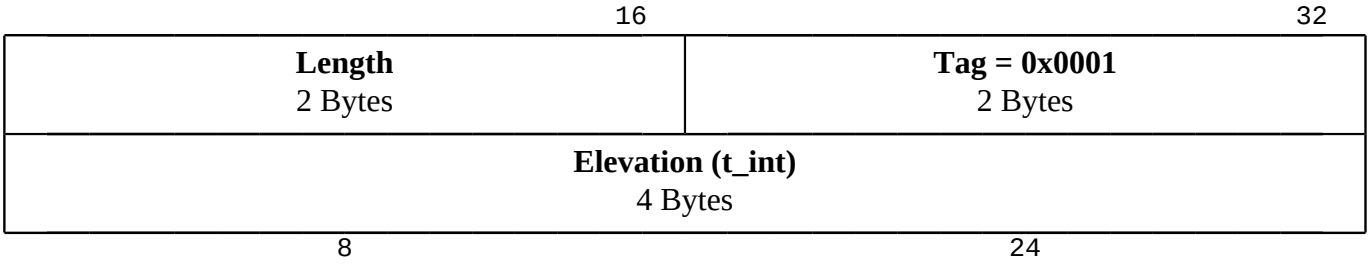
Tag 0: Undefined Surrounding

Default value when no camera surrounding is configured, can also be used to remove the camera surrounding configuration



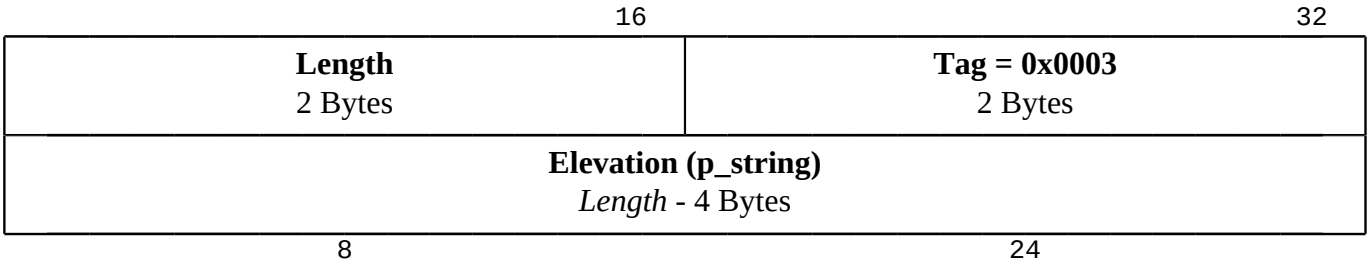
Tag 1: Elevation

Specify the elevation relativ to the ground plane in units of 1/(2^16) m. Camera ground model plane is parallel to coordinate system defined by (CONF_CAMERA_POSITION and CONF_CAMERA_ORIENTATION).



Tag 3: Elevation

Specify the elevation relativ to the ground plane. Camera ground model plane is parallel to coordinate system defined by (CONF_CAMERA_POSITION and CONF_CAMERA_ORIENTATION).



Format of Payload String

Elevation

© Copyright 2001-2021 Bosch Security Systems

	Max. Length	Limits	Description
Elevation	5.4	[-32767,32767]	Specify the position of the camera in a local cartesian coordinate system

Example

```
elevation = "1000.124"
-> elevation= 1000.124 m
```

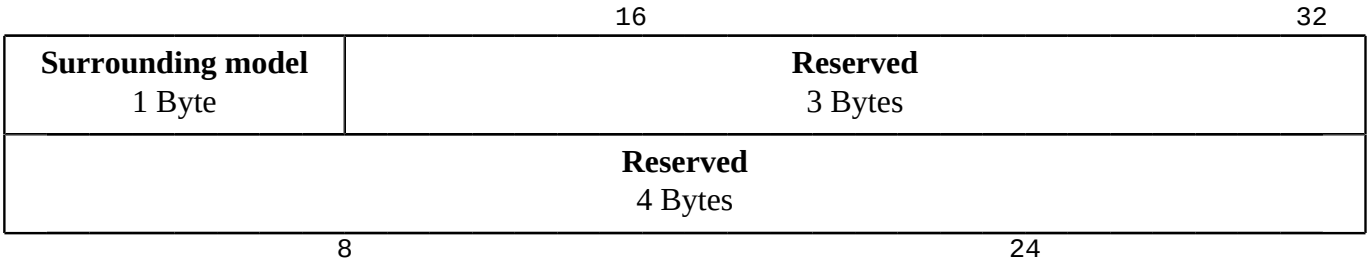
© Copyright 2001-2021 Bosch Security Systems

2.91 CONF_CAMERA_SURROUNDING_OPTIONS

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Surrounding model

Values:

No	0	
Flat plane	1	flat plane only elevation can be configured (plane is assumed to be parallel to the (world, local) coordinate system

© Copyright 2001-2021 Bosch Security Systems

2.92 CONF_CAMNAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0019	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Deprecated! Use CONF_CAMNAME_LINES instead.	
Write	p_unicode	service	Deprecated! Use CONF_CAMNAME_LINES instead. (max 31 unicode character + null delimiter)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.93 CONF_CAMNAME_LINES

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb1	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Read out camera name, use CONF_ENC_STAMPING_PROPERTIES to determine line length l (a line consists of l UTF-16 chars) and the number of supported lines n (command length = l*n | UTF-16 chars), all non used chars must be set to zero	
Write	p_unicode	service	Set camera name, use CONF_ENC_STAMPING_PROPERTIES to determine line length l (a line consists of l UTF-16 chars) and the number of supported lines n (command length = l*n | UTF-16 chars), all non used chars must be set to zero	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.94 CONF_CAMNAME2

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a7e	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Deprecated! Use CONF_CAMNAME_LINES instead.	
Write	p_unicode	service	Deprecated! Use CONF_CAMNAME_LINES instead. (max 31 unicode character + null delimiter)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.95 CONF_CAPABILITY_LIST

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff10	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16	32
0xBABA 2 Bytes	**Version** 2 Bytes
Num Sections 2 Bytes	**Section [0]** (see description)
...	**Section [N]** (see description)

8 24

Version

Current version of the capabilities

Num Sections

Number of following sections

Section

16	32
Type 2 Bytes	**Size** 2 Bytes
Num Elements 2 Bytes	**Element [0]** (see description)
...	**Element [N]** (see description)

8 24

© Copyright 2001-2021 Bosch Security Systems

Type

Type of Element

Values:

Video	0x0001
Audio	0x0002
Serial	0x0003
IO	0x0004
Camera Data	0x0005

Size

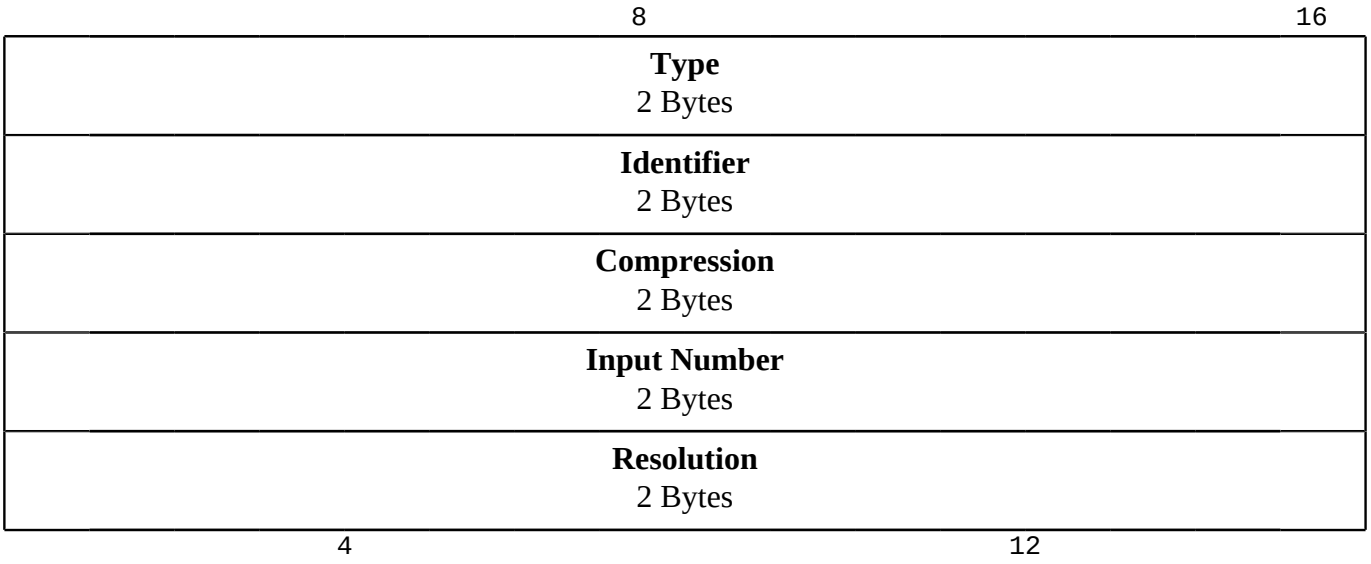
Size of the section including SectionType, Size and NbElement. If the section is unknown, you can skip to the next using the size.

Num Elements

Determines how many Elements are following. The definition of each Element depends on the type of the section.

Element

'Element' payload for 'Type' = 'Video Element' (0x0001)



Type

Values:

VIDEO_ENCODER	0x0001
VIDEO_DECODER	0x0002
VIDEO_TRANSCODER	0x0003

© Copyright 2001-2021 Bosch Security Systems

Identifier

Identifier is the RCP numeric descriptor to use to address the entity. It should be unique when associated with the type and the compression. In case of transcoders this numeric descriptor is informative only. It is not necessary to address the entity directly (use the relevant rcp commands instead).

Compression

One or multiple of the following

Values:

VIDEO_COMP_MPEG2	0x0001
VIDEO_COMP_MPEG4	0x0002
VIDEO_COMP_H264	0x0004
VIDEO_COMP_JPEG	0x0008
VIDEO_COMP_H265	0x0010

Note: Please use CONF_CODER_VIDEO_OPERATION_MODE to configure the video standard. The dependencys are explained in detail at CONF_CODER_VIDEO_OPERATION_MODE.

Input Number

Input Number is the number of the physical input from which the entity gets or puts its video. In case of transcoder this value is unused and set to 0.

Resolution

One or multiple of the following

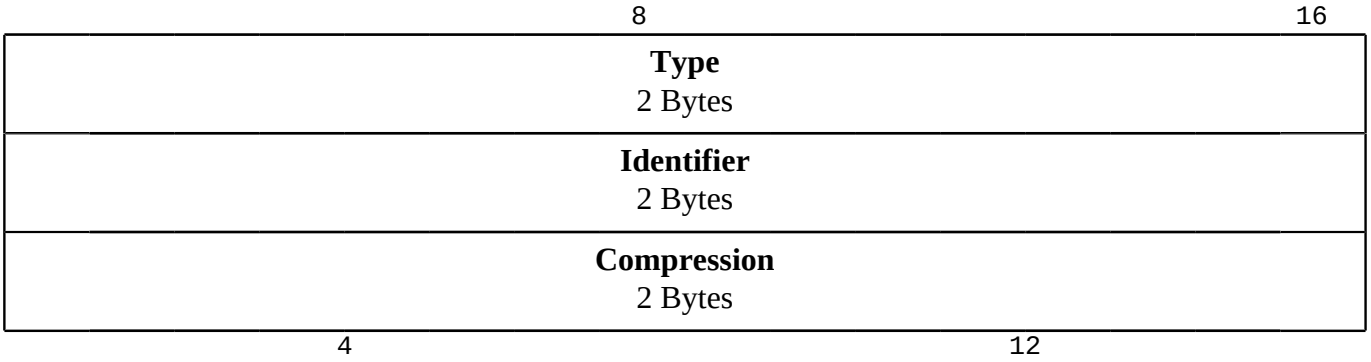
Values:

VIDEO_RESO_QCIF	0x0001
VIDEO_RESO_CIF	0x0002
VIDEO_RESO_2CIF	0x0004
VIDEO_RESO_4CIF	0x0008
VIDEO_RESO_CUSTOM	0x0010
VIDEO_RESO_QVGA	0x0020
VIDEO_RESO_VGA	0x0040
VIDEO_RESO_HD720	0x0080
VIDEO_RESO_HD1080	0x0100
VIDEO_RESO_WD144	0x0200
VIDEO_RESO_WD288	0x0400
VIDEO_RESO_WD432	0x0800
VIDEO_RESO_HD2592x1944	0x1000

Note: This list is not further maintained/extended. If additional resolutions are supported, the VIDEO_RESO_CUSTOM value is set and the available resolutions can be obtained via the command CONF_VID_H264_ENC_BASE_OPERATION_MODE_CAPS. For JPEG encoders the available resolutions can be obtained via the command CONF_JPEG_STREAM_SETUP_OPTIONS_VERBOSE.

© Copyright 2001-2021 Bosch Security Systems

'Element' payload for 'Type' = 'Audio Element' (0x0002)



Type

Values:

AUDIO_ENCODER	0x0001
AUDIO_DECODER	0x0002

Identifier

Identifier is the RCP numeric descriptor to use to address the entity. It should be unique when associated with the type and the compression.

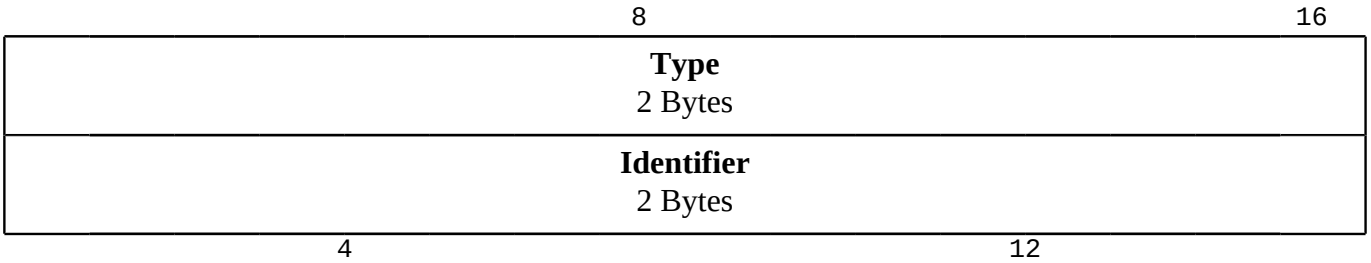
Compression

One or multiple of the following

Values:

AUDIO_COMP_MPEG2	0x0001
AUDIO_COMP_G711	0x0002
AUDIO_COMP_AAC	0x0004

'Element' payload for 'Type' = 'Serial Element' (0x0003)



© Copyright 2001-2021 Bosch Security Systems

Type

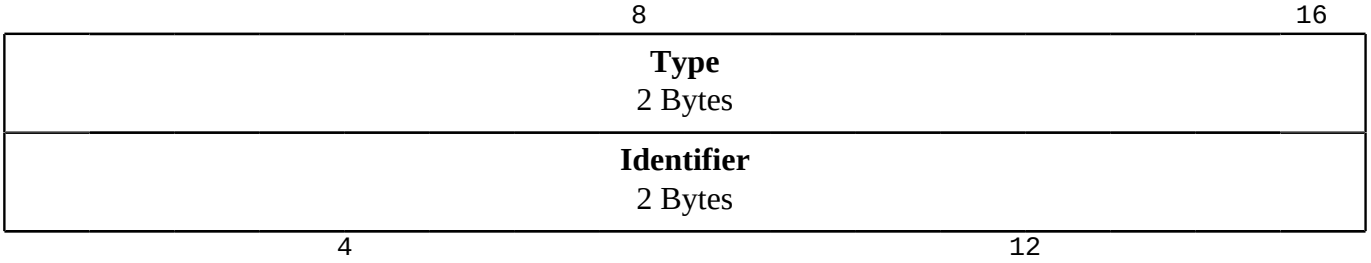
Values:

SERIAL_RS232	0x0001
SERIAL_RS485	0x0002
SERIAL_RS422	0x0004

Identifier

RCP numeric descriptor to use to address the entity.

'Element' payload for 'Type' = 'IO Element' (0x0004)



Type

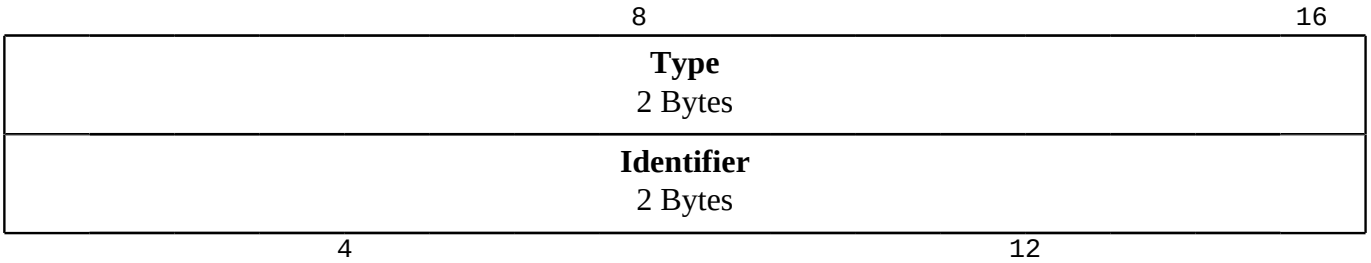
Values:

IO_INPUT	0x0001
IO_OUTPUT	0x0002
IO_VIRTUAL_IN	0x0004

Identifier

RCP numeric descriptor to use to address the entity.

'Element' payload for 'Type' = 'Camera Data Element' (0x0005)



Type

Values:

CAMDATA_BILINX	0x0001

© Copyright 2001-2021 Bosch Security Systems

Identifier

RCP numeric descriptor to use to address the entity.

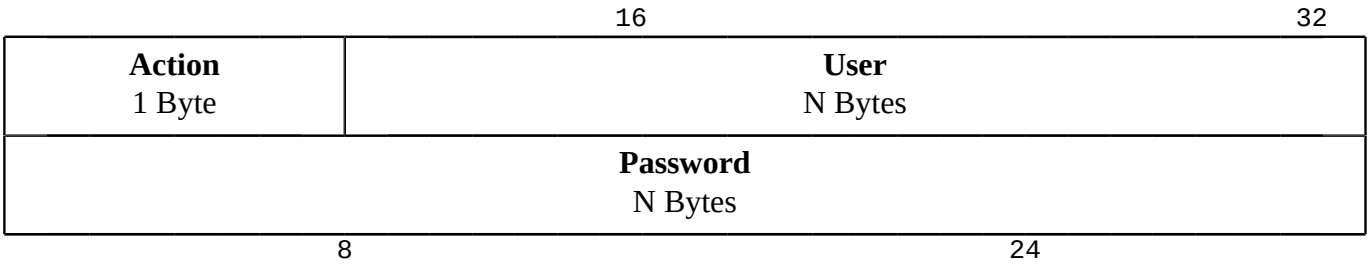
© Copyright 2001-2021 Bosch Security Systems

2.96 CONF_CBS_COMMISSION

[API: Cbs]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c72	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	minimal	Commission to CBS-Account. Action, User, Password. See detailed description for payload structure	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Action

Cancel Commission	0
Start Commission	1

User

User name (0-terminated, max-len 128)

Password

User password (0-terminated, max-len 128)

© Copyright 2001-2021 Bosch Security Systems

2.97 CONF_CBS_DESTINATION

[API: Cbs]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c75	destination number (starting with 1)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get CBS destination and connection configuration. See detailed description for payload structure	
Write	p_octet	service	Set CBS destination and connection configuration. See detailed description for payload structure	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16		32
Port 2 Bytes	**SSL** 1 Byte	**reserved...** 5 Bytes
reserved ...		
URL N Bytes		
8		24

Port

Destination port

SSL

SSL Socket not provided	0
SSL Socket provided	1

reserved

Reserved for future use. Set to zero.

© Copyright 2001-2021 Bosch Security Systems

URL

Destination URL (0-terminated, max-len 128)

Example

Payload for port 443, ssl 1, url "api.remote.boschsecurity.com"

```
0x01bb0100000000006170692e72656d6f74652e626f736368736563757269747 92e636f6d00
```

Payload for port 443, ssl 1, url "api.remotest.cbs.boschsecurity.com"

```
0x01bb0100000000006170692e72656d6f746573742e6362732e626f7363687365637572697 47
```

© Copyright 2001-2021 Bosch Security Systems

2.98 CONF_CBS_STATUS

[API: Cbs]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c73	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Status of CBS Commission. see detailed Description :	
Write	p_octet	minimal	Update commission status, no parameters needed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

8	16
cbs status 1 Byte	**cbs action status** 1 Byte
4	12

cbs status

Is a status byte for the Remote Portal connection

Registered	0
OnGoing	1
UnRegistered	2
Unknown	255

cbs action status

Is a detailed status byte for 'Remote Portal' dialog

Ok	0
OnGoing	1
UnRegistered	2
Canceled	3
Reconfiguration Forbidden	0xe0
Reconfiguration Conflict	0xe1
Reconfiguration UnprocessibleEntity	0xe2
Reconfiguration Locked	0xe3
Reconfiguration ServiceUnavailable	0xe4
Reconfiguration AuthError	0xe5
CloudConnector not running	0xf5
CloudDestinationError	0xf6

© Copyright 2001-2021 Bosch Security Systems

CloudUnreachable no DNS Server configured	0xf7
CloudUnreachable DNS could not be resolved	0xf8
DeviceSecretError	0xf9
ReceiveError	0xfa
MessageError	0xfb
InvalidMacAddressFormat	0xfc
FriendlyNameEmpty	0xfd
CloudUnreachable	0xfe
UnexpectedError	0xff

© Copyright 2001-2021 Bosch Security Systems

2.99 CONF_CERTIFICATE

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command CONF_CERTIFICATE supports the following operations:

- Upload a certificate optionally including its private key.
- Download a certificate or signing request from the device.
- Delete a certificate or signing request. A delete implicitly deletes an associated private key.

Each certificate/signing request/key is identified by a unique label. If a certificate has a matching private key both have the same label. When the command fails it returns a command specific error code in the payload.

On upload the command supports the following file formats:

- PEM: PEM file format. With this format you can upload a certificate and a key at once. For this you must set Certificate and Private Key bits in type tag.
- DER: Binary DER file format. For this file format you can only upload one type of content and the type must be exactly specified in type tag.
- PKCS#12 or PFX: PKCS#12 (sometimes also called PFX) file format. This file format carried information what is included, so you can upload certificate and key in one file. The type tag does not matter when using this file format and can be set to certificate and key or even be not provided.

You can also use this command to upload encrypted private keys as PEM, DER or PKCS12 file. To do this you upload the file containing the encrypted key. The device stores the encrypted key internally and waits, that the password is provided later with help of the CONF_CERTIFICATE_REQUEST command.

To delete a certificate/CSR or key you have to provide the following tags: Tag 0 with the label, tag 1 with 0, tag 2 with length 4 and no data and tag 3 with entry type 0xFF (delete).

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

© Copyright 2001-2021 Bosch Security Systems

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

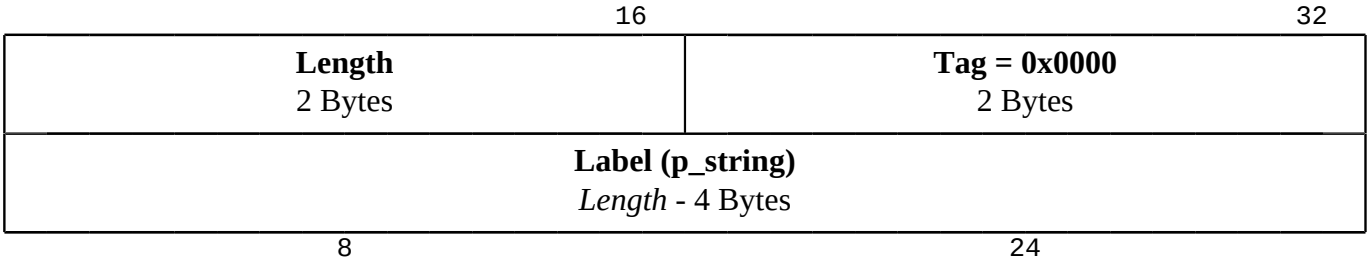
Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

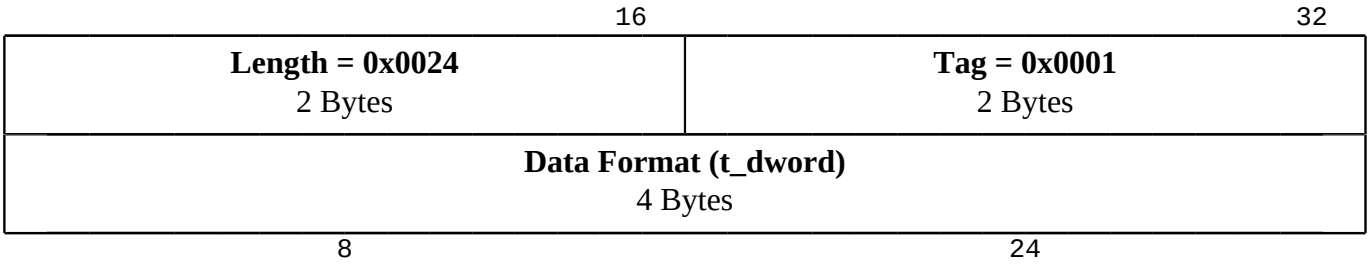
Tag 0: Label

Contains the label string, no zero termination. For the maximum supported length see CONF_CERTIFICATE_OPTIONS.



Tag 1: Data Format

Specifies the certificate/signing request/key data format (PEM / DER / PKCS#12(PFX) is supported).



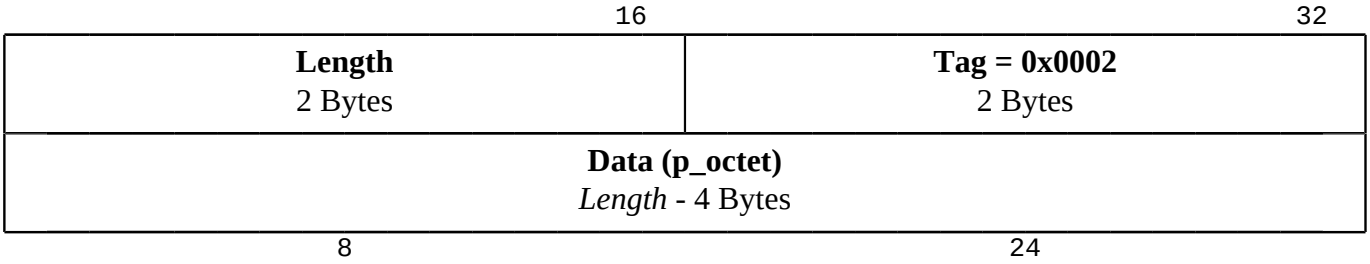
Values:

PEM	0
DER	1
PKCS#2 / PFX	2

© Copyright 2001-2021 Bosch Security Systems

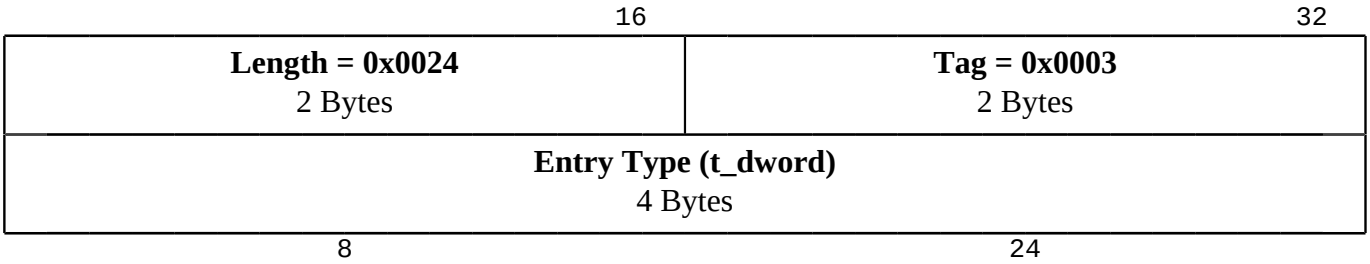
Tag 2: Data

Contains the certificate/signing request/key data in the specified format. If this tag contains no data (zero payload length) the corresponding entry is deleted from the device.



Tag 3: Entry Type

Specifies the entry type for read and write. This tag is mandatory for PEM and DER format and should appear only once.

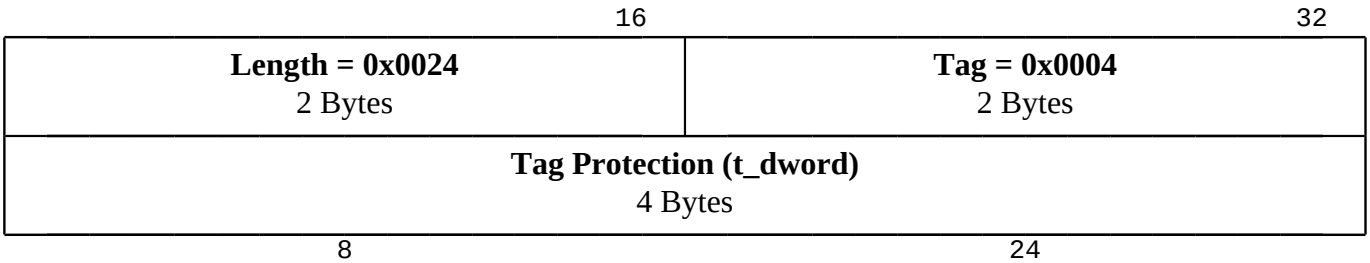


Values:

Certificate	1
Signing Request (read or delete)	2
Private key (write only)	4
Encryption key (write only)	8
Any type (delete only)	255

Tag 4: Tag Protection

Contains a bit mask showing the protection bits of the certificate. Each bit of the 32 bit value shows a specific protection.



© Copyright 2001-2021 Bosch Security Systems

Values:

	Mask	Name	Description
Bit 0	0x00000001	Factory Reset	Protect at factory reset

Example

Delete cert with label 'test':

```
0x0008000074657374 0008000100000000 00040003000000FF
```

Command Specific Errors

CERTIFICATE_CMD_ERROR_FAIL	0x00
CERTIFICATE_CMD_ERROR_INV_ARGS	0x01
CERTIFICATE_CMD_ERROR_ALREADY_EXISTS	0x02
CERTIFICATE_CMD_ERROR_FORMAT	0x03
CERTIFICATE_CMD_ERROR_NO_CERT_FOR_KEY	0x04
CERTIFICATE_CMD_ERROR_NO_KEY_ENTRY	0x05
CERTIFICATE_CMD_ERROR_NO_STORAGE_SPACE	0x06

© Copyright 2001-2021 Bosch Security Systems

2.100 CONF_CERTIFICATE_LIST

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0beb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_CERTIFICATE_LIST allows to query the labels of all certificates, signing requests and keys installed on a device. The command CONF_CERTIFICATE can then be used to retrieve a certificate or signing request using the label. The format has an internal tagged structure. Some tags are optional and might be absent if the data is not available. The tags are always in the sequence as listed below.

Payload Structure

List Entry [0]
...
List Entry [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

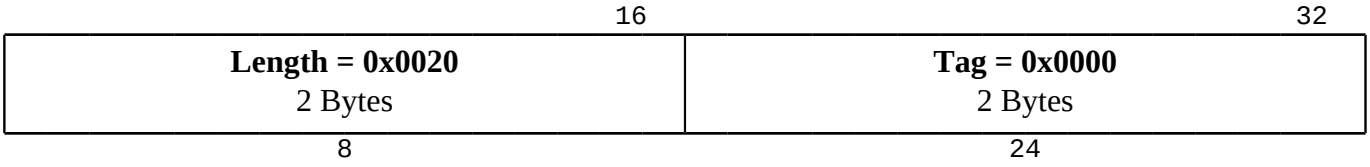
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

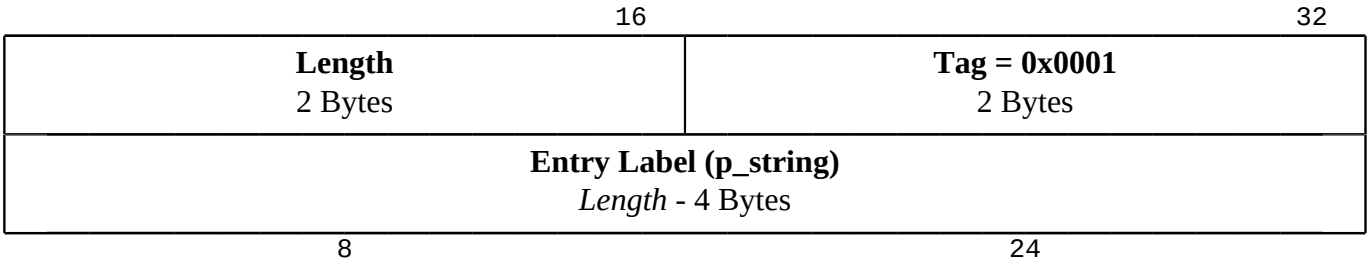
Tag 0: List Entry

The payload of tag list entry contains further tagged values described below. For each certificate, signing request or key with unique label on the device one list entry tag is contained in the command payload.



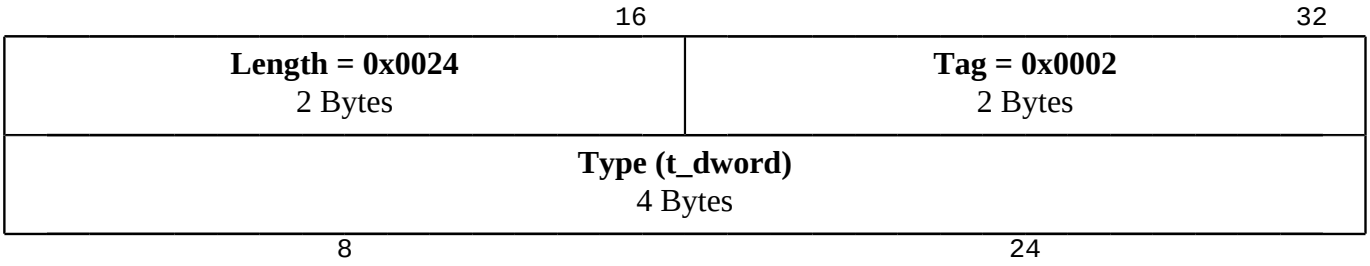
Tag 1: Entry Label (Required)

Contains the entry label string, no zero termination.



Tag 2: Type

Contains the entry type bitmask. All types with the same label are listed together as one entry.

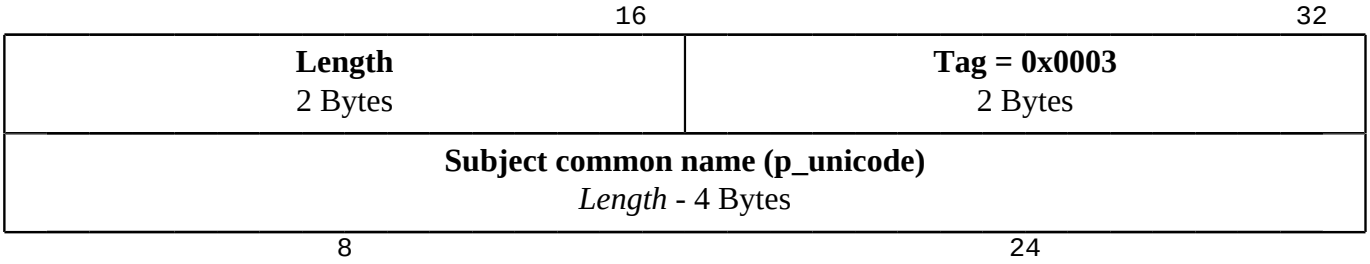


	Mask	Name
Bit 7	0x00000080	Encrypted PKCS#12
Bit 6	0x00000040	Key decryption in progress
Bit 5	0x00000020	Signing request generation in progress
Bit 4	0x00000010	Certificate generation in progress
Bit 3	0x00000008	Encrypted private key
Bit 2	0x00000004	Private key
Bit 1	0x00000002	Signing request
Bit 0	0x00000001	Certificate

© Copyright 2001-2021 Bosch Security Systems

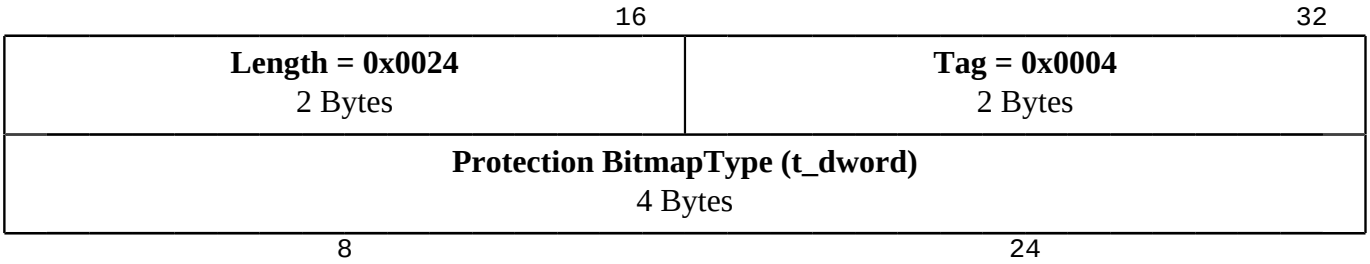
Tag 3: Subject common name

Contains the subject common name string (UTF-16) of the certificate or signing request. This entry is optional and only used for certificates and signing requests.



Tag 4: Protection BitmapType

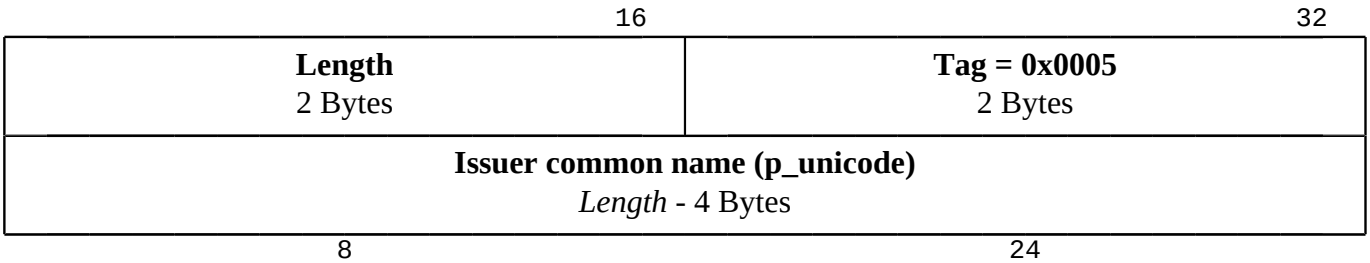
Show the protection bits of this entry. This tag is optional if the type supports protection bits and if any protection is set. If this tag is missing, the item is unprotected. The following values are possible:



	Mask	Name
Bit 1	0x00000002	Protect from manual deletion
Bit 0	0x00000001	Protect at factory reset

Tag 5: Issuer common name

Contains the issuer common name string (UTF-16) of the certificate. This entry is optional and only used for certificates.

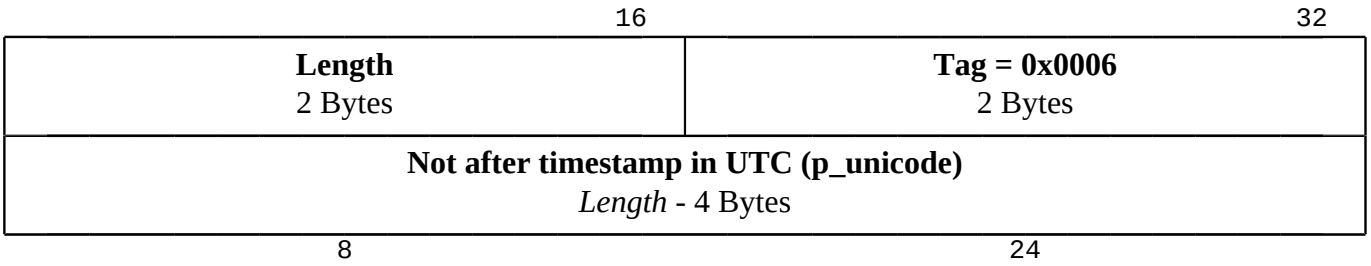


Tag 6: Not after timestamp in UTC

Contains the UTC string (UTF-16, format YYMMDDhhmmssZ) of the not after timestamp of the certificate . This entry is optional and only used for certificates. If tag = 7 (Certificate's not after timestamp in

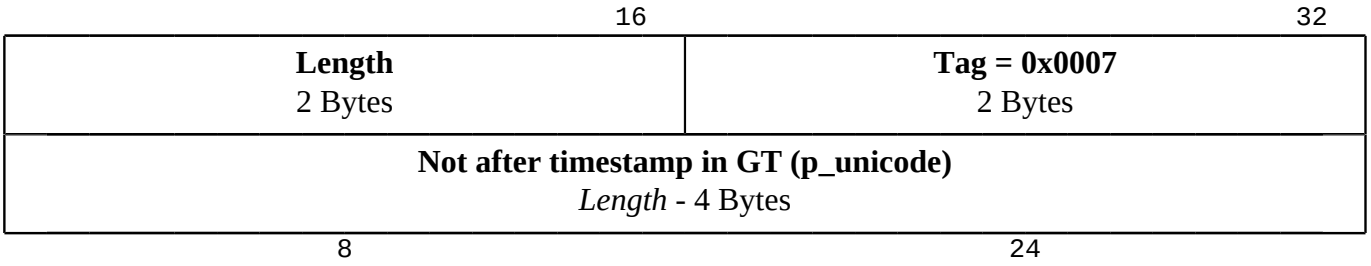
© Copyright 2001-2021 Bosch Security Systems

GT) is present you should always use the date from there as UTC is limited to 1950-01-01 up to 2049-12-31. For compatibility reason both tags will be present when the date can be presented in both formats.



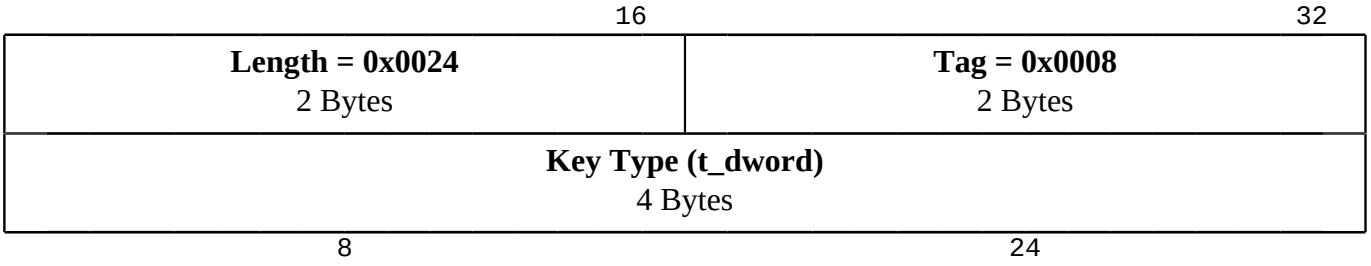
Tag 7: Not after timestamp in GT

Contains the General time (GT) string (UTF-16, format YYYYMMDDhhmmssZ) of the not after timestamp of the certificate. This entry is optional and only used for certificates. Thisa entry should always be used instead of tag 6 (Certificate's not after timestamp in UTC) because of it's wider time range.



Tag 8: Key Type

Indicating the key algorithm and the key size this certificate or signing request uses. The following values are possible:



Below enumeration lists example values for the key type association:

RSA 1024 Bit	0
RSA 2048 Bit	1
Elliptic Curve P256	2
RSA 4096	3

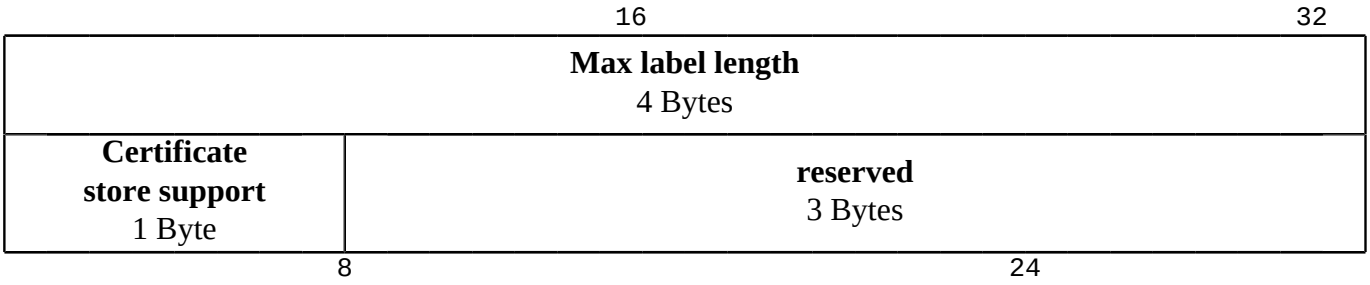
© Copyright 2001-2021 Bosch Security Systems

2.101 CONF_CERTIFICATE_OPTIONS

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bea	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Certificate store support

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

2.102 CONF_CERTIFICATE_REQUEST

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bec	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command CONF_CERTIFICATE_REQUEST allows to start a certificate signing request or a key decryption operation. Since both operations may take some time to complete, this command may return before the actual completion of the operation. The progress of the operation can be tracked by registering for the CONF_CERTIFICATE_REQUEST_PROGRESS message.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

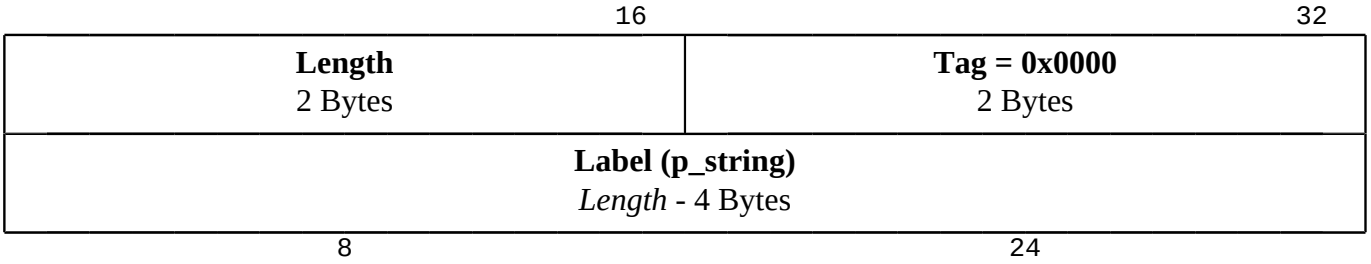
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

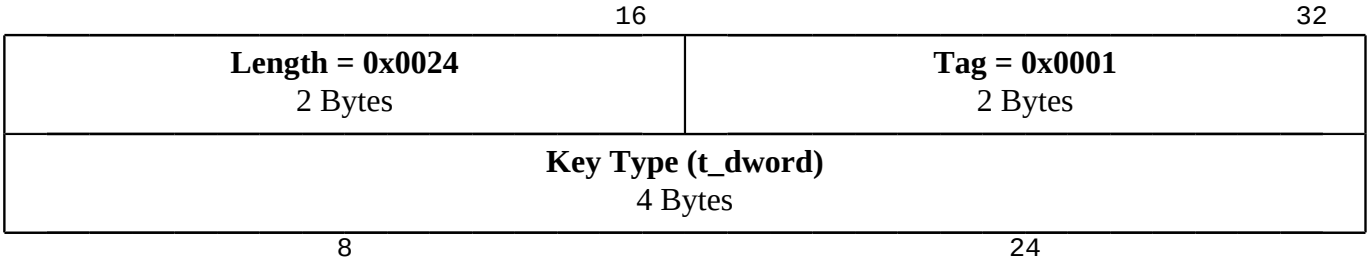
Tag 0: Label

Contains the label string, no zero termination. Access the generated private key and the signed certificate using the provided label after the process has finished successfully. For key decryption this label is used to identify the key to be decrypted.



Tag 1: Key Type

Specifies the key type for the signing request/certificate to be generated. The list of supported types for a specific device can be requested using command CONF_CERTIFICATE_REQUEST_OPTIONS.

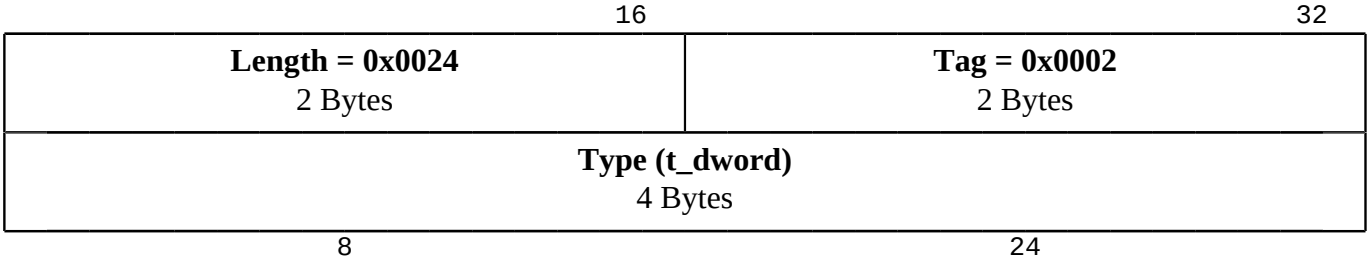


Below enumeration lists example values for the key type association:

RSA 1024 Bit	0
RSA 2048 Bit	1
Elliptic Curve P256	2
RSA 4096	3

Tag 2: Type

Below enumeration lists example values for the key type association:



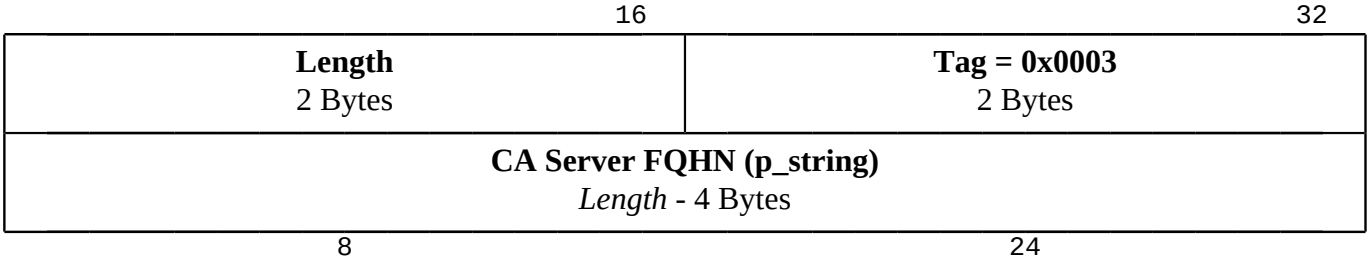
© Copyright 2001-2021 Bosch Security Systems

Values:

PKCS10	0	Create PKCS#10 CSR
SelfSigned	3	Create self-signed certificate
Descrypt	4	Decrypt private key

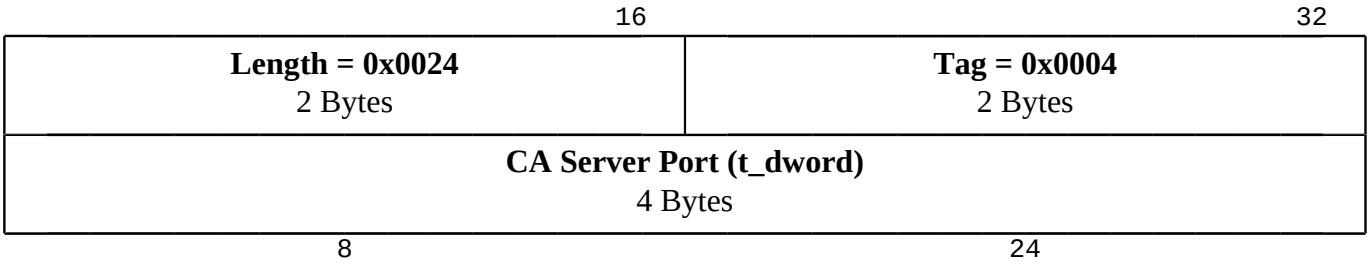
Tag 3: CA Server FQHN

Specifies the CA server address as string in Fully-Qualified Host Name format (fully domain name or IPV4 or IPV6 as string), no zero termination. This tag is optional and only used when a signing request is sent to a server.



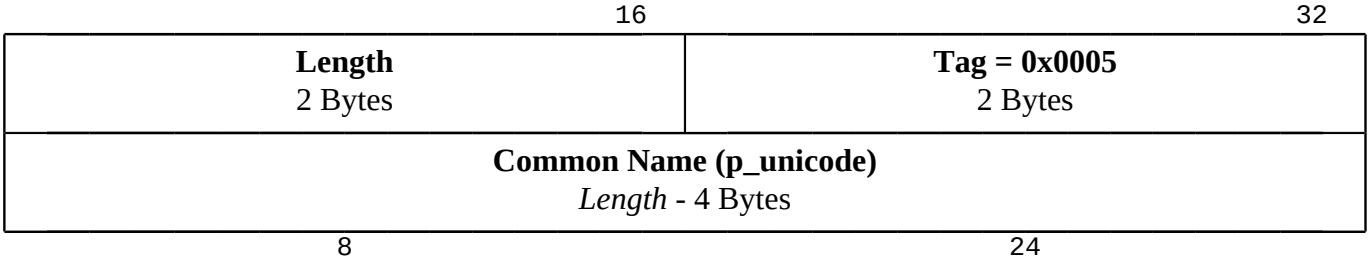
Tag 4: CA Server Port

Specifies the CA server port. This tag is optional and only used when a signing request is sent to a server.



Tag 5: Common Name

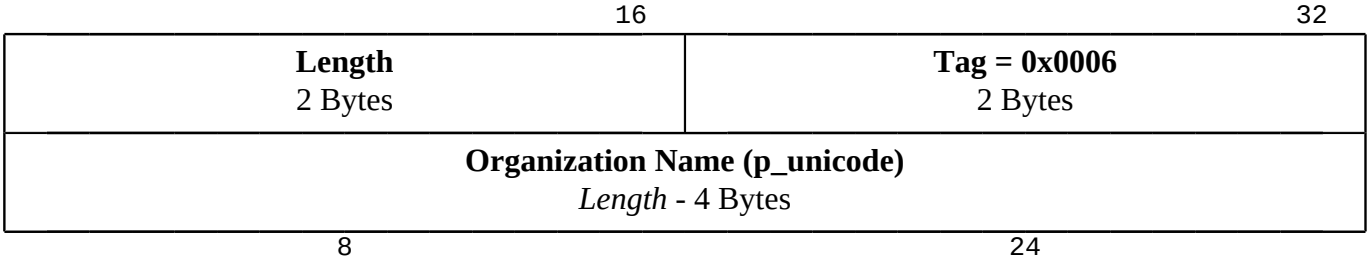
Contains the common name string (UTF-16) for the certificate/signing request, no zero termination. This tag is optional.



© Copyright 2001-2021 Bosch Security Systems

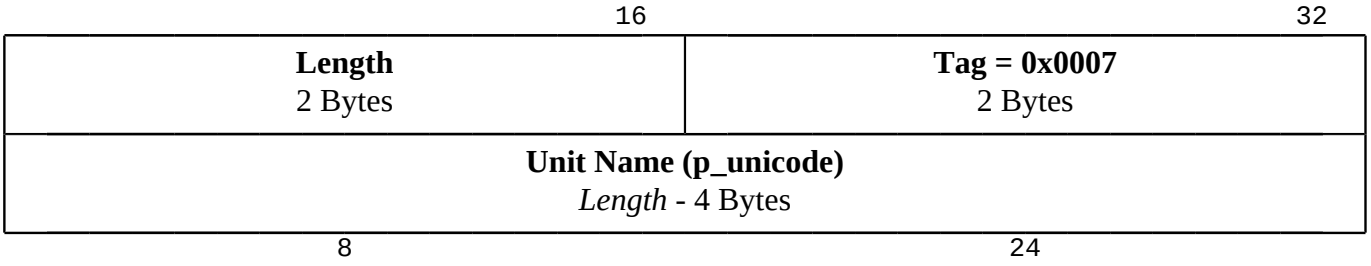
Tag 6: Organization Name

Contains the organization name string (UTF-16) for the certificate/signing request, no zero termination.



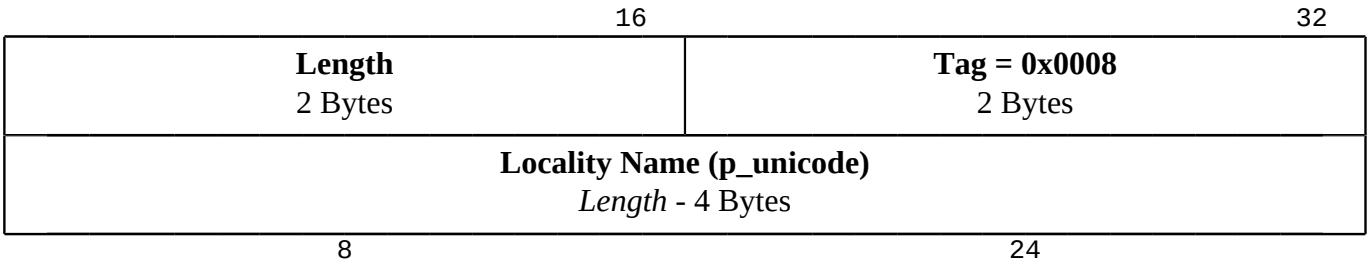
Tag 7: Unit Name

Contains the organizational unit name string (UTF-16) for the certificate/signing request, no zero termination. This tag is optional.



Tag 8: Locality Name

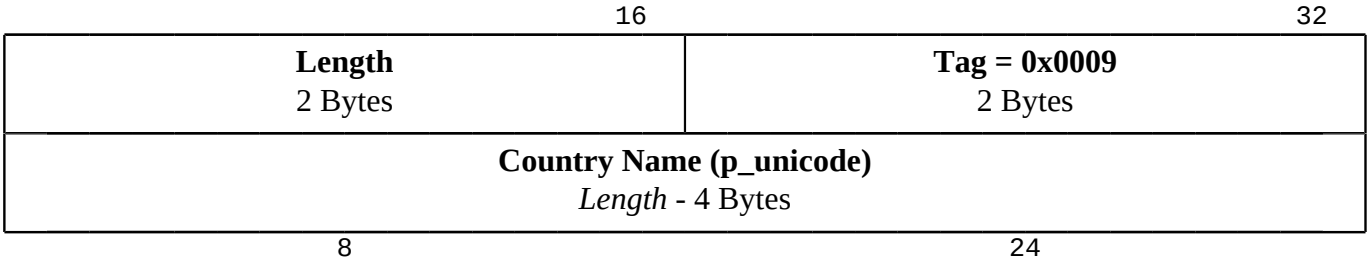
Contains the locality name string (UTF-16) for the certificate/signing request, no zero termination. This tag is optional.



© Copyright 2001-2021 Bosch Security Systems

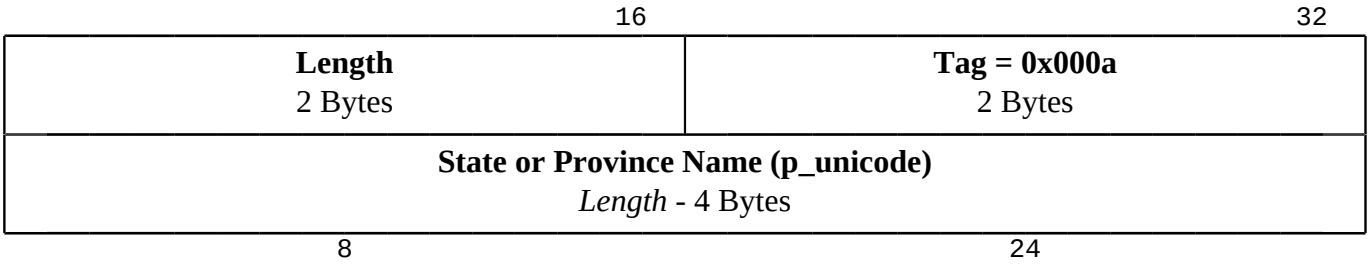
Tag 9: Country Name

Contains the country name string (UTF-16) for the certificate/signing request, no zero termination. This tag is optional.



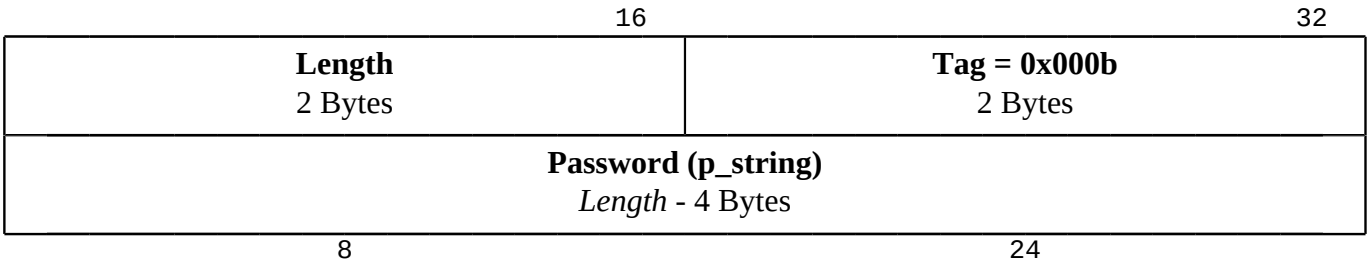
Tag 10: State or Province Name

Contains the state or province name string (UTF-16) for the certificate/signing request, no zero termination. This tag is optional.



Tag 11: Password

Contains the password used for private key decryption, no zero termination. This tag is optional.



Tag 12: Certificate Extension

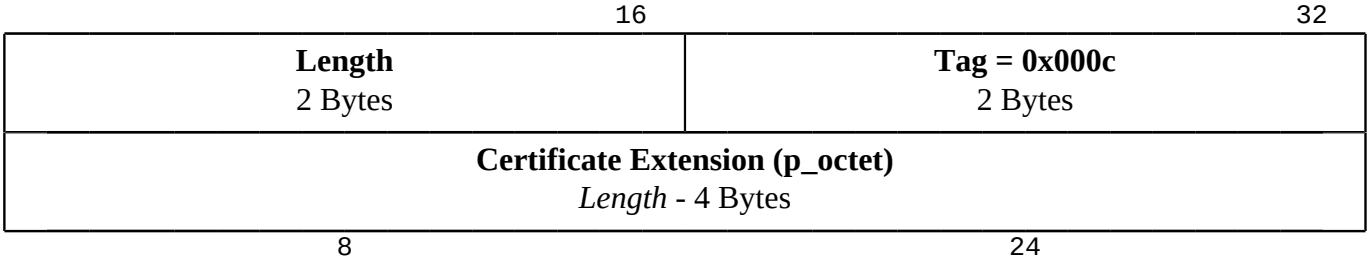
Using this tag you can provide one X.509 certificate extension which should be added to a self signed certificate or a certificate request (there the extension will be added as ExtensionRequest).

The extension must already be encoded in it's ASN.1 representation as defined for 'Extension' in RFC5280 chapter 4.1. This means a sequence containing an object identifier followed by a bitstring which contains the extension specific payload.

© Copyright 2001-2021 Bosch Security Systems

The ASN.1 structure is directly added to the certificate/certificate request, so the caller is responsible for proper content of the extension which will be provided by this tag.

If this tag is missing no extension will be added. It can occur a maximum of 5 times.



© Copyright 2001-2021 Bosch Security Systems

2.103 CONF_CERTIFICATE_REQUEST_OPTIONS

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bed	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	4 bytes max label length, 4 bytes max IP string length, 4 bytes number of supported key types, 4 bytes number of supported cmd types, numKeyTypes * keyType (4 bytes type ID, 64 bytes type label), numCmdTypes * Type (4 bytes cmd type ID, 64 bytes cmd type label), 4 bytes max password length	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.104 CONF_CERTIFICATE_REQUEST_PROGRESS

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf0	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message informs about the current status of certificate/signing request/key operations started by a CONF_CERTIFICATE_REQUEST. The format has an internal tagged structure.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

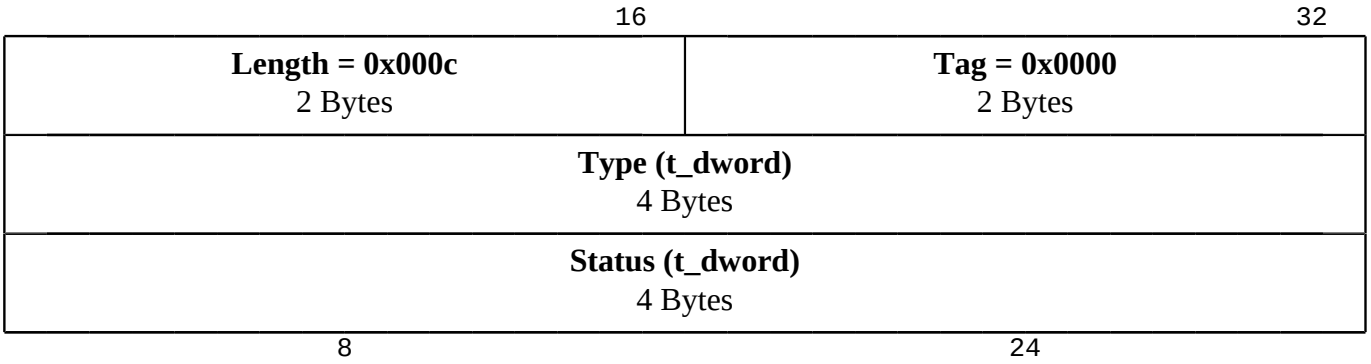
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0: Contains the command type and its status.

Contains the command type and its status.



Type

PKCS10	0	Create PKCS#10 CSR
SelfSigned	3	Create self-signed certificate
Descrypt	4	Decrypt private key

Status

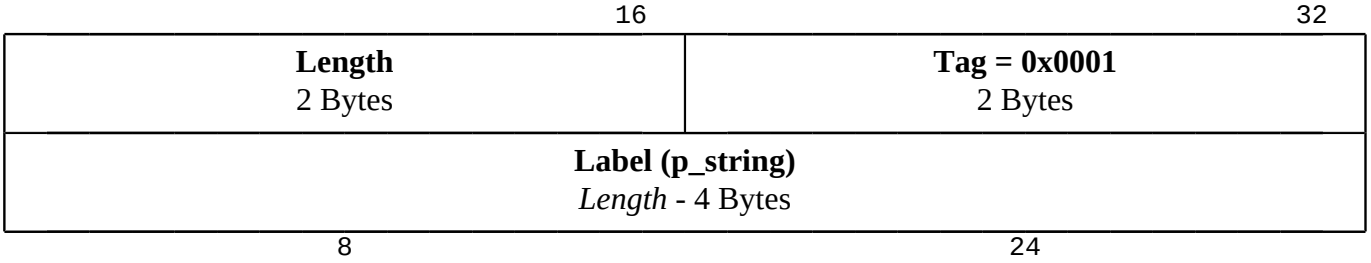
Values in the range of 0 to 100 represent the current status, other values need to be handled as error.

Status values:	
Generating request	25
Sending request	50
Request completed successfully	100
Error values:	
General error	129
File already exists error	130
No free key entry error	131
No storage space error	132
Server error	133
Decrypt error	134
Entry not found error	135
No certificate found for key error	136

© Copyright 2001-2021 Bosch Security Systems

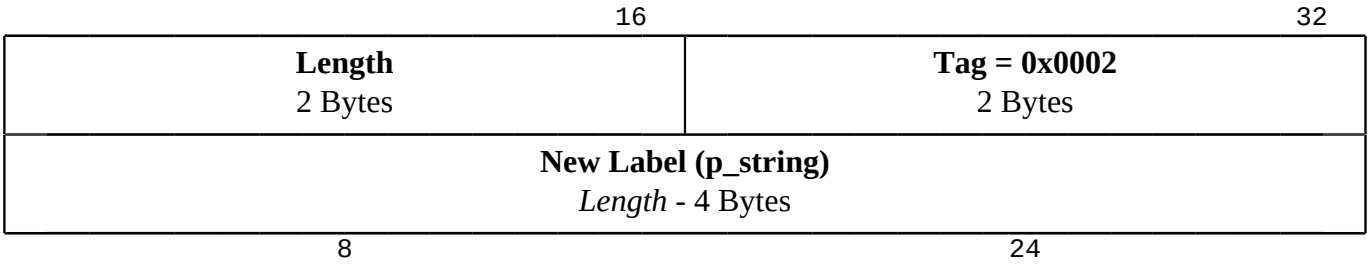
Tag 1: Label

Contains the certificate/signing request/key label, no zero termination.



Tag 2: New Label

Contains the certificate/signing request/key label, no zero termination. This tag is optional and is used to transmit the new label that is used to store the entry after the operation is complete (e.g. a key is decrypted and matched against a certificate and stored using the certificate label).



© Copyright 2001-2021 Bosch Security Systems

2.105 CONF_CERTIFICATE_USAGE

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_CERTIFICATE_USAGE allows to configure for which purpose the certificates and keys on the device are used. The format has an internal tagged structure, the allowed tags and values can be queried via CONF_CERTIFICATE_USAGE_OPTIONS.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

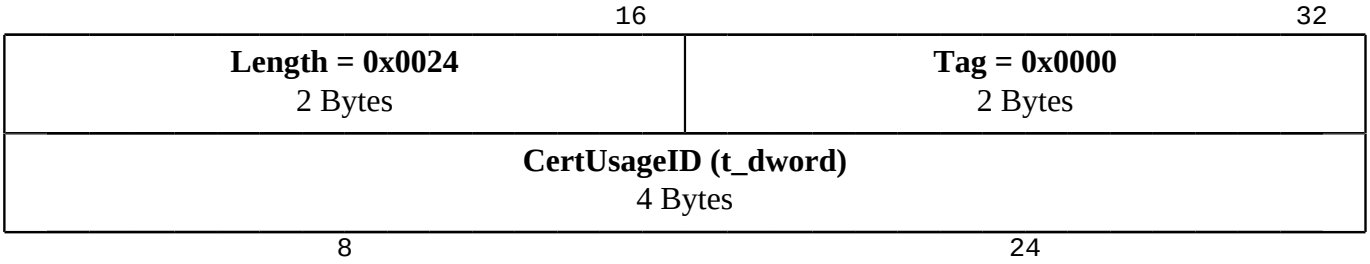
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0: CertUsageID

Contains the certificate usage ID. This tag is mandatory for the read and write commands and should appear only once.

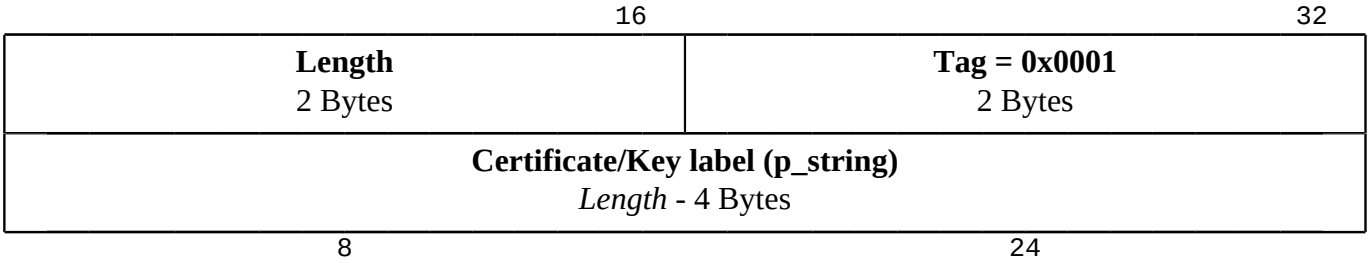


There are two types of certificate usages that can be distinguished by the highest bit (bit 31) of the CertUsageID:

	Mask	Name	Description
Bit 31	0x80000000	Type	0: Usage is only applicable for certificates having the private key stored on the device also (e.g. certificates used for signing), 1: Usage can be used for certificates for which the private key is not stored on the device (e.g. certificates only used for verify other certificates or signatures).

Tag 1: Certificate/Key label

Contains the label string used to identify the certificate/key, no zero termination. Multiple label tags can be contained in the payload.



© Copyright 2001-2021 Bosch Security Systems

2.106 CONF_CERTIFICATE_USAGE_OPTIONS

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command returns the list of certificate usages available in the camera. Each usage has a unique ID and a label which can be presented to the end user. In the CONF_CERTIFICATE_USAGE command the usageID is referenced.

Payload Structure

16	32
Write support 4 Bytes	
Max Label Length 4 Bytes	
Max Number Of Labels 4 Bytes	
Number Of Usages 4 Bytes	
Usage Entry [0] (see description)	
...	
Usage Entry [N] (see description)	
8	24

Write support

Write Not Supported	0
Write Supported	1

© Copyright 2001-2021 Bosch Security Systems

Max Label Length

Maximum length of one label string for the usages.

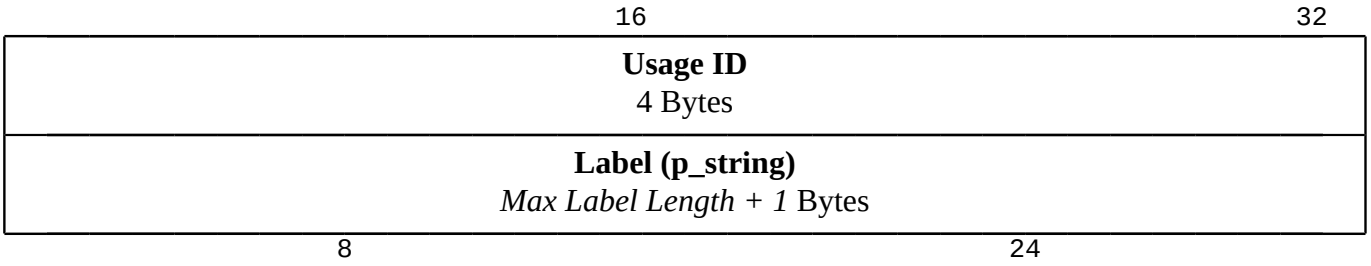
Max Number Of Labels

Defines the maximum number of certificates which can be associated with one usage.

Number Of Usages

Amount of usage entries following after this entry.

Usage Entry



Usage ID

This bit mask contains a unique numerical ID for this usage. Bits 31 and 30 are not part of this numerical ID and have a special meaning. They indicate properties of this usage regarding the certificates which can be associated with this usage.

	Mask	Name	Description
Bit 31	0x80000000	Certificate Type	0: Certificate is an end-user certificate. Certificates in these usages are presented to a peer for client authentication for example. These usages can only contain one certificate. The certificate must be associated with a private key stored on the device. 1: Certificate is a trusted certificate. Certificates in these usages are used to verify certificates presented from the external peer. These usage can contain multiple certificates. The certificate needs no private key on the device.
Bit 30	0x40000000	Multi Associate	This bit is only valid, if bit 31 is 0. 0: This end-user certificate usage can only be associated with 1 certificate. 1: This end-user certificate usage can be associated with multiple certificate (not used or supported at the moment).

© Copyright 2001-2021 Bosch Security Systems

Note: The functionality of this bit is only reserved for future usage with the described meaning and is not implemented yet.

Combined Values

Mask	Name	Description
0x3FFFFFFF	ID Number	

© Copyright 2001-2021 Bosch Security Systems

2.107 CONF_CHECK_DEFAULT_BUTTON_STATE

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c3b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Check the current status of the (factory) default button. 0=inactive; 1=active	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.108 CONF_CHECK_POS_FB_STATE

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c3c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Check the current status of the position feedback sensors. (Only applicable on certain devices).	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

	Mask	Name
Bit 1	0x02	hall sensor
Bit 0	0x01	end position fb switch

© Copyright 2001-2021 Bosch Security Systems

2.109
CONF_CHECK_WS_USERNAMETOKEN_AUTH_TIMESTAMP

[API: onvif]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d2d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	service	#ValueList: 0=WSUsernametoken timestamp check disabled; 1=WSUsernametoken timestamp check disabled(default value);	
Write	t_dword	service	Requires reboot, #ValueList: 0=WSUsernametoken timestamp check disabled; 1=WSUsernametoken timestamp check disabled(default value);	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.110 CONF_CLOUD_COMMISSIONING_STATUS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c44	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the status of cloud commissioning. After successful commissioning, the cloud sets this status so the camera knows about it (and can e.g. change LED accordingly). 0=uncommissioned; 100=commissioning successfully finished	
Write	t_word	service	Store the status of cloud commissioning. After successful commissioning, the cloud sets this status so the camera knows about it (and can e.g. change LED accordingly). 0=uncommissioned; 100=commissioning successfully finished	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.111 CONF_CLOUD_WATCH_SETTINGS

[API: CloudWatch]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns cloud watch settings of the device in a tagged format	
Write	p_octet	service	Writes cloud watch settings of the device in a tagged format	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

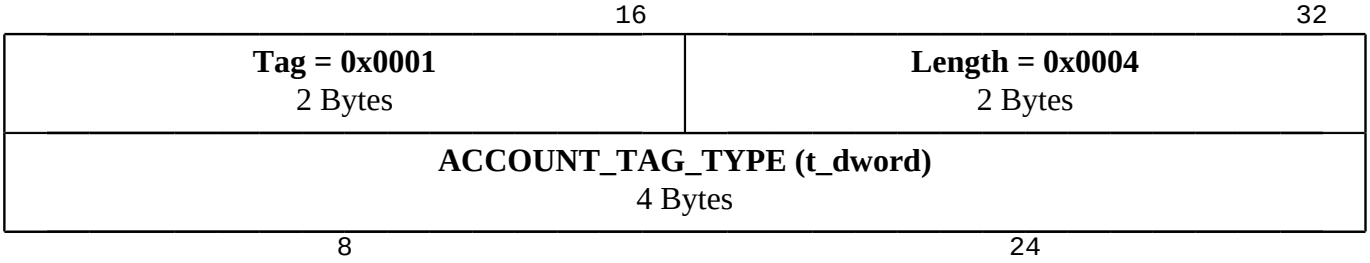
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

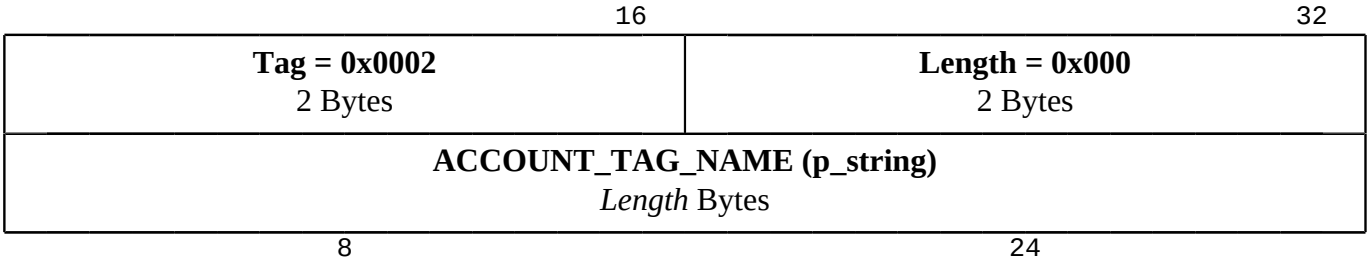
Tag 1: ACCOUNT_TAG_TYPE

0: not configured, 1: ftp, 2: dropbox, 5: amazon s3, 6: kinesis



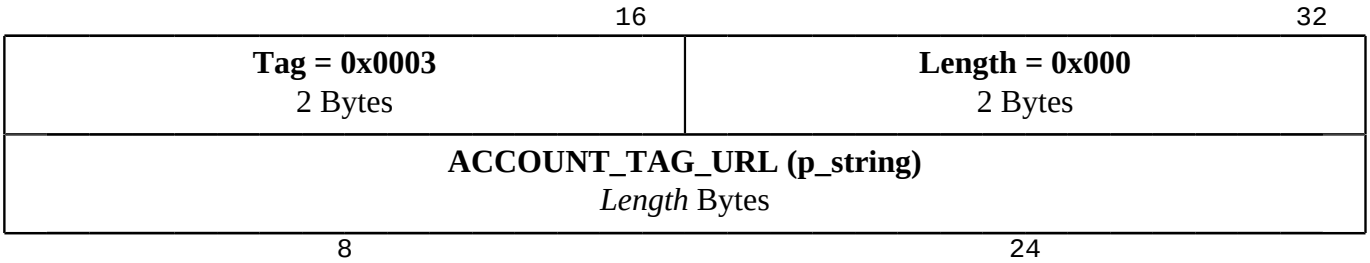
Tag 2: ACCOUNT_TAG_NAME

account name



Tag 3: ACCOUNT_TAG_URL

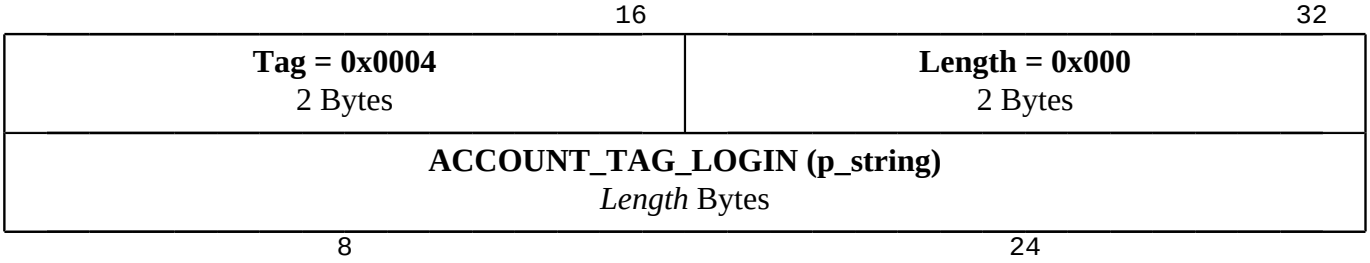
url, e.g url to the ftp server



© Copyright 2001-2021 Bosch Security Systems

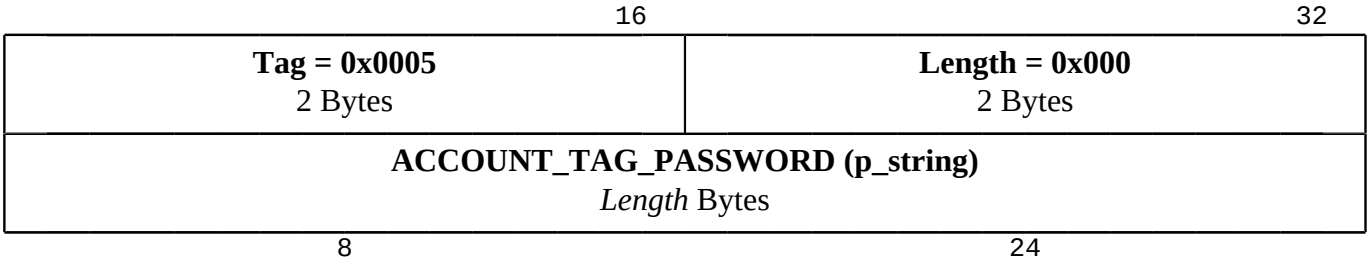
Tag 4: ACCOUNT_TAG_LOGIN

username



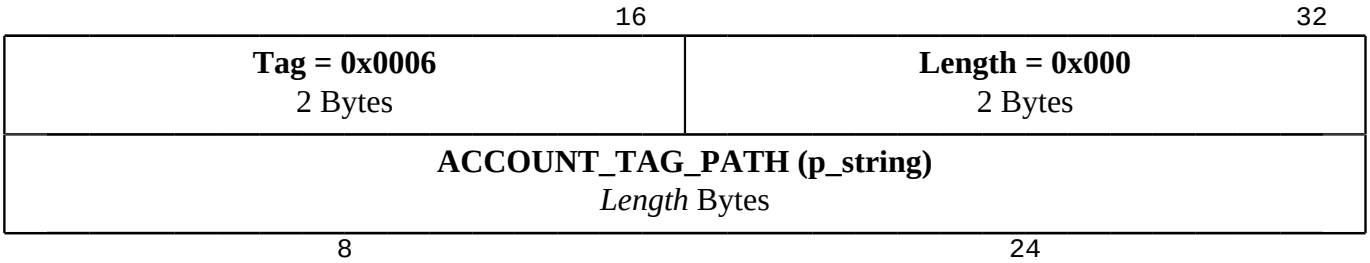
Tag 5: ACCOUNT_TAG_PASSWORD

password



Tag 6: ACCOUNT_TAG_PATH

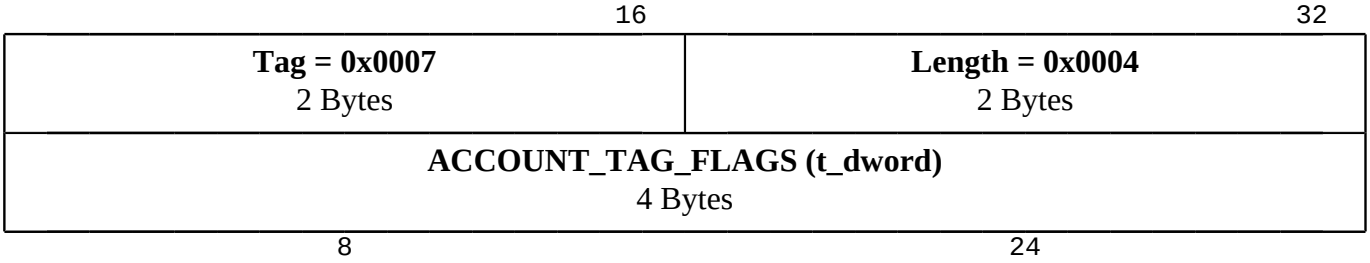
path



© Copyright 2001-2021 Bosch Security Systems

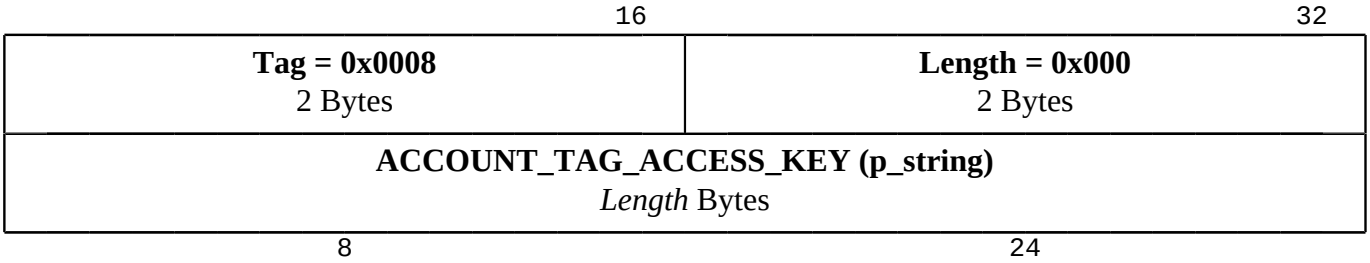
Tag 7: ACCOUNT_TAG_FLAGS

optional flags (e.g. FTP Encryption mode 0=off,1=TLS)



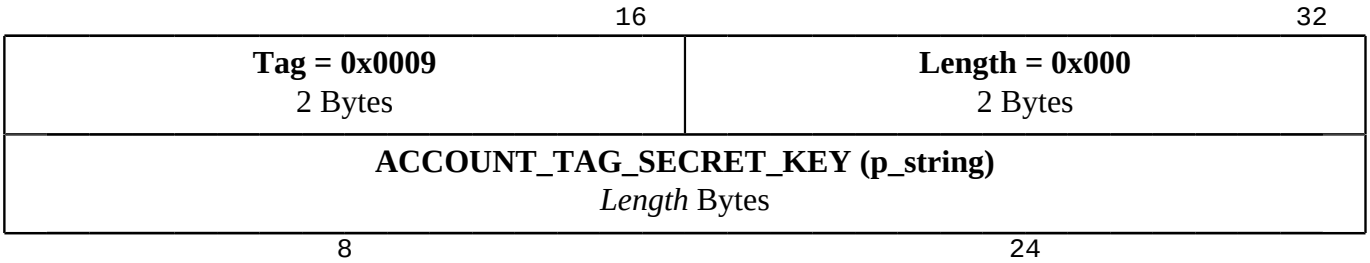
Tag 8: ACCOUNT_TAG_ACCESS_KEY

aws access key (max 128 bytes)



Tag 9: ACCOUNT_TAG_SECRET_KEY

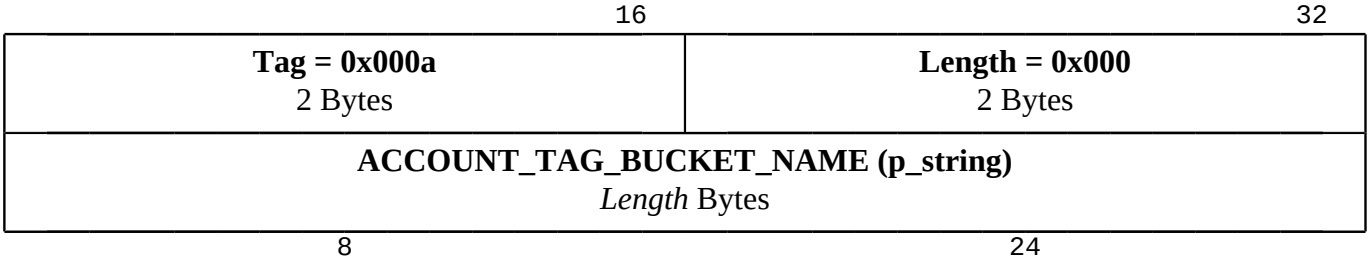
aws secret access key (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

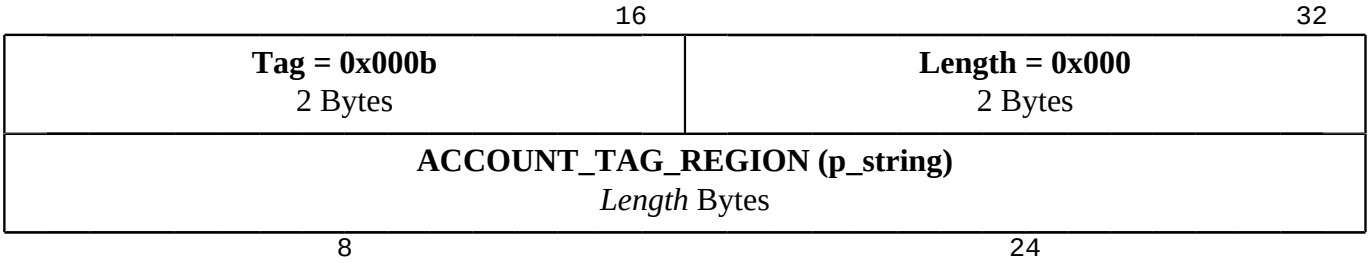
Tag 10: ACCOUNT_TAG_BUCKET_NAME

bucket name (amazon s3), max 128 bytes



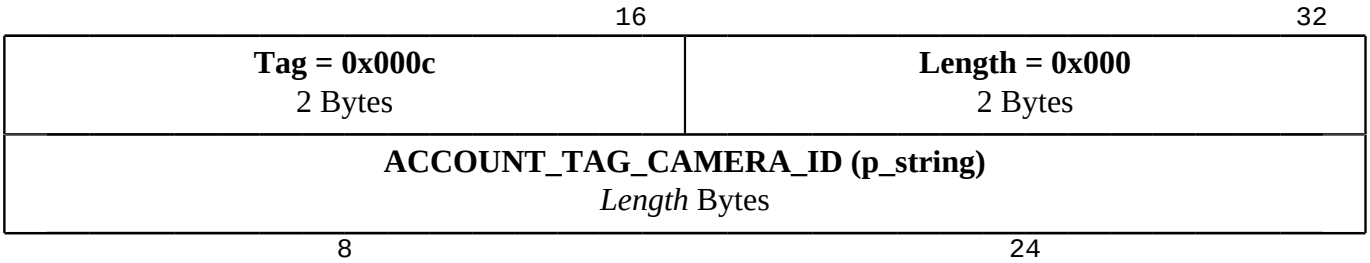
Tag 11: ACCOUNT_TAG_REGION

aws region



Tag 12: ACCOUNT_TAG_CAMERA_ID

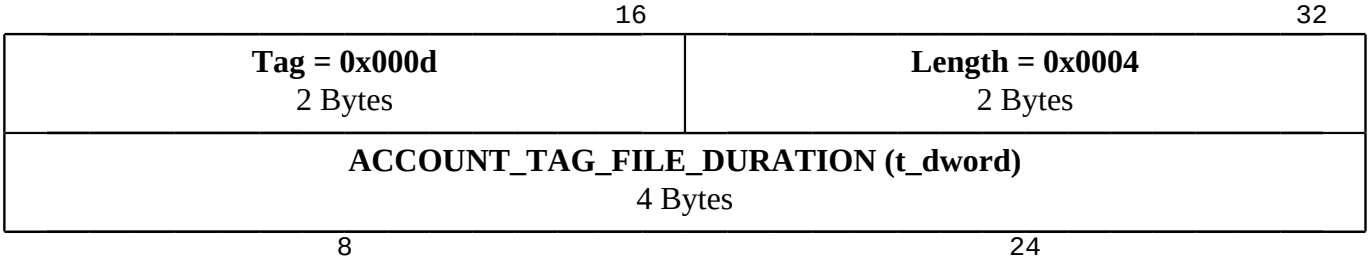
s3 camera id, string will be added to the uploaded filename (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

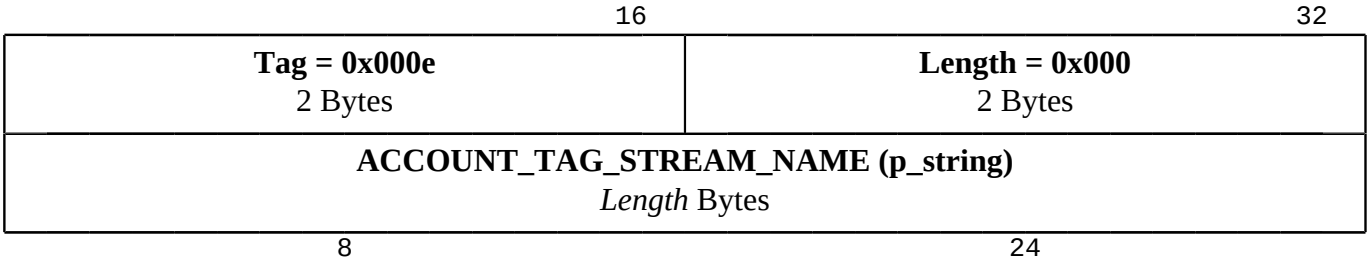
Tag 13: ACCOUNT_TAG_FILE_DURATION

s3 file duration in seconds



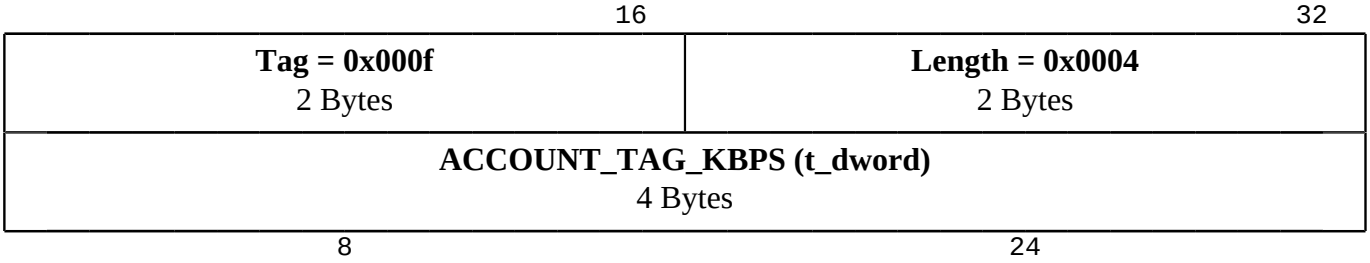
Tag 14: ACCOUNT_TAG_STREAM_NAME

stream name



Tag 15: ACCOUNT_TAG_KBPS

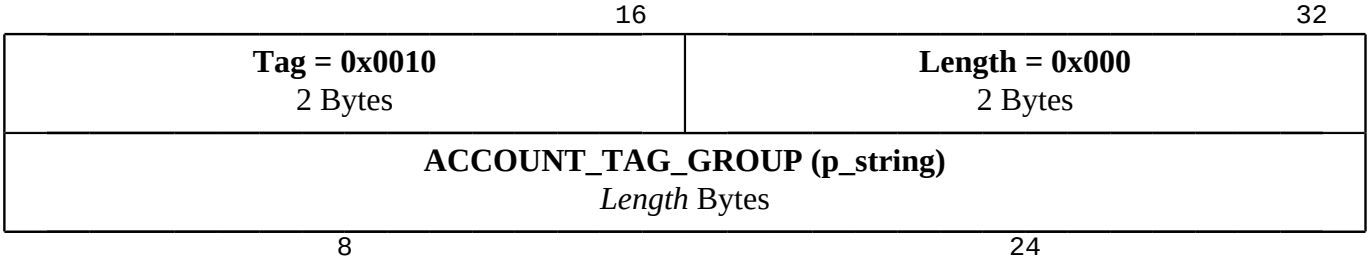
maximum datarate for the backup, see CONF_BACKUP_MAX_KBPS



© Copyright 2001-2021 Bosch Security Systems

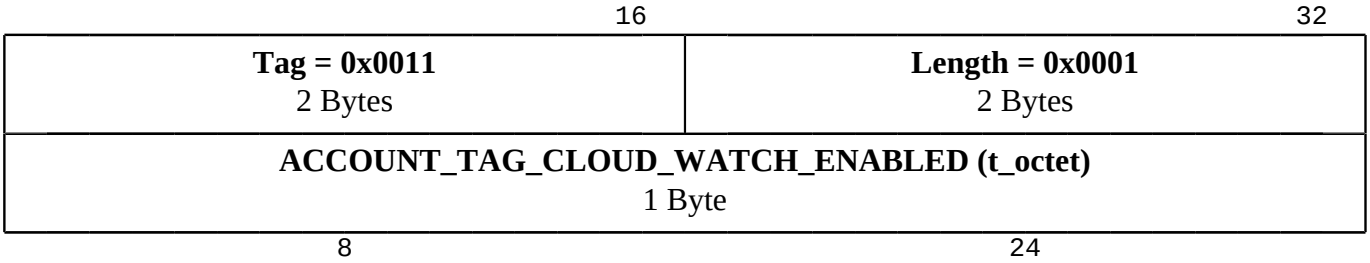
Tag 16: ACCOUNT_TAG_GROUP

cloud watch log group



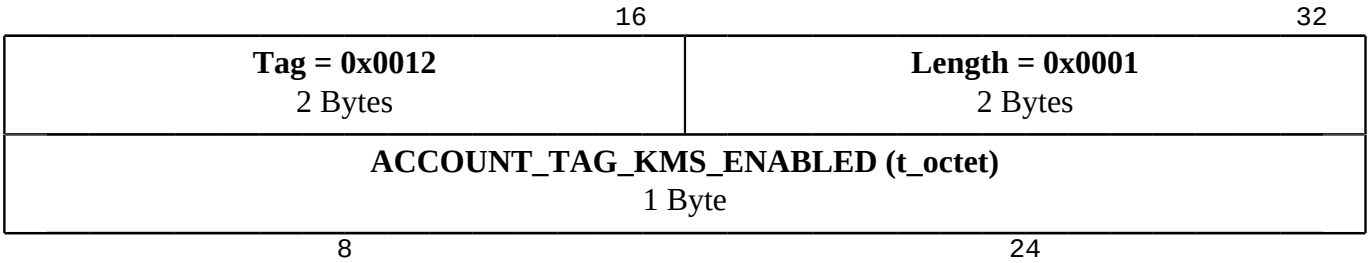
Tag 17: ACCOUNT_TAG_CLOUD_WATCH_ENABLED

cloud watch enabled/disabled



Tag 18: ACCOUNT_TAG_KMS_ENABLED

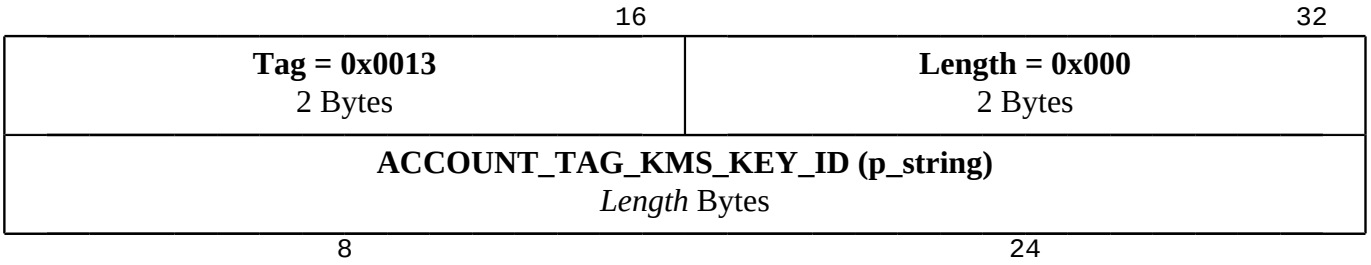
kms key encryption enabled/disabled



© Copyright 2001-2021 Bosch Security Systems

Tag 19: ACCOUNT_TAG_KMS_KEY_ID

aws kms key id (max 128 bytes)



© Copyright 2001-2021 Bosch Security Systems

2.112 CONF_CLUSTER_GROUP_SETTING

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ca	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy	List of (DWORD ip, DWORD flags, OCTET[6] mac, WORD reserved) cluster members including this unit	
Write	p_octet	minimal	List of (DWORD ip, DWORD flags, OCTET[6] mac, WORD reserved) cluster members including this unit	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.113 CONF_CLUSTER_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09cb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	always_legacy	Returns the logical position inside a cluster (0=stand alone, >0=slot nbr.)	
Write	t_dword	service	Only in generic: set vrm instance (0 <= ClusterId < 32)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.114 CONF_CODER_SPECIFIC_ENC_PROFILES

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c9b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	0 means not supported on this platform; if supported all commands that address a encoder profile via the 'num' parameter have to send the abs video tx coder in the upper 8bits of the 'num';	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.115 CONF_CODER_VIDEO_OPERATION_MODE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a9c	absolute coder instance, if zero - all coders which can handle the option are addressed internally	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal		
Write	t_dword	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

There are two different modes signaled via INDIVIDUAL_ENCODER_OPERATION_MODE_CONFIG @CONF_DEVICE_CAPABILITIES

1: invidual encoder configuration mode:

CONF_CODER_VIDEO_OPERATION_MODE, num parameter for a specific absolute coder has to be provided; global configration (numDesc = 0) is not supported

0: global encoder configuration mode :

CONF_CODER_VIDEO_OPERATION_MODE, num parameter supports only 0 as global option for all possible encoders

1: invidual encoder configuration mode:

use CONF_CODER_VIDEO_OPERATION_MODE_OPTIONS with NumDesc = absolute coder number to get the supported options at CONF_CODER_VIDEO_OPERATION_MODE

0: global encoder configuration mode :

use CONF_CODER_VIDEO_OPERATION_MODE_OPTIONS with NumDesc = 0 to get the supported global options at CONF_CODER_VIDEO_OPERATION_MODE

Set one video coder (absolute coder number given by NumDesc) or all video coders, which can handle the option, depending on the supported configuration mode, to a video operation mode.

Allowed options can be queried by CONF_CODER_VIDEO_OPERATION_MODE_OPTIONS: 0 = jpeg, 1 = h263, 2 = h264, 3 = h265, 0x103 = h_265 without B-Frames.

Live connections will be dropped; Only applicable if recording isn't active and configured to off.

© Copyright 2001-2021 Bosch Security Systems

2.116 CONF_CODER_VIDEO_OPERATION_MODE_OPTIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca3	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List of options (DWORDS) which can be written at CONF_CODER_VIDEO_OPERATION_MODE for a specific coder	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Global options are provided.	0
Absolute coder instance.	Any

Payload Structure

Option [0] 4 Bytes
...
Option [N] 4 Bytes

Option

Values:

jpeg	0	
h263	1	
h264	2	
h265	3	
h_265 without B-Frames	0x0103	h_265 without B-Frames delivers higher framerates

© Copyright 2001-2021 Bosch Security Systems

2.117 CONF_COMMERCIAL_TYPE_NUMBER

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the commercial type number (CTN) of the device. The string may be shortened or generalized as the CTN also contains parts the firmware does not know.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.118 CONF_COMMISSION_SETUP

[API: commissioning]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Returns status of commissioning	
Write	p_octet	service	Moves camera to coordinates commissioning	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Read Payload Structure

16	32
Flags 4 Bits	**Reserved** 28 Bits
8	24

Flags

	Mask	Name	Description
Bit 3	0x8	Commissioning Available	This camera has commisioning
Bit 2	0x4	Is Initialized	Commissioning is ready to use
Bit 1	0x2	Last Command Failed	set to 1 on fail, will be set to 0 with each write command.
Bit 0	0x1	Running	Last write command is still processing

Write Payload Structure

16	32
Reserved 4 Bytes	
New View Left 4 Bytes	
New View Top 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

New View Right 4 Bytes
New View Bottom 4 Bytes

8 24

Coordinates are relative in 1/0x8000. Corner top left is (0,0), bottom right is (0x8000,0x8000). If new view top left coordinate is same as new view bottom right coordinate, zoom will not be changed, just view is repositioned.

New View Left

X coordinate of top left corner.

New View Top

Y coordinate of top left corner.

New View Right

X coordinate of lower right corner.

New View Bottom

Y coordinate of lower right corner.

© Copyright 2001-2021 Bosch Security Systems

2.119 CONF_COMPLETE_CALIBRATION_ELEMENT

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c35	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	4 bytes reserved, calibration element	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.120 CONF_CONFIG_SEALING_ENABLED

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c8a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get enabled state of config sealing 0=sealing disabled; 1=sealing enabled; 2=seal broken	
Write	t_octet	service	Enable config sealing 0=sealing disabled; 1=sealing enabled; 2=seal broken (2 for internal write)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.121 CONF_CONFIG_SEALING_STATUS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c8c	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Return status of config seal,	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

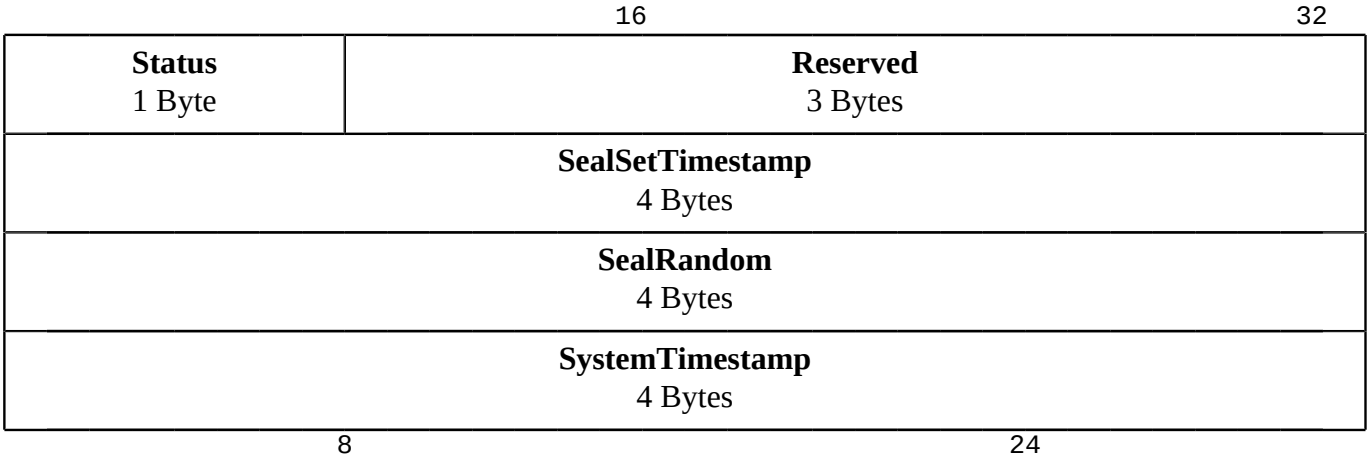
Description

The system can be set up in a way that unexpected configuration changes on the device cause a alert message, even when the user uses a valid login and password for this action.

After the whole system configuration is completed CONFIG_SEALING_ENABLED needs to be set to 1 (= Seal enabled). In this state each critical configuration change causes the device to send a CONFIG_SEALING_STATUS message.

If a client wants to activly verify if a seal is valid then it should read CONFIG_SEALING_STATUS.

Payload Structure



Status

Sealing off	0
Sealing on and seal valid	1
Sealing on and seal broken	2

SealSetTimestamp

Timestamp when seal was activated in seconds since year 2000.

© Copyright 2001-2021 Bosch Security Systems

SealRandom

Random number generated once when seal was activated.

SystemTimestamp

Current system time in seconds since year 2000. Can be used to protect against time manipulation on SealSetTimestamp

© Copyright 2001-2021 Bosch Security Systems

2.122 CONF_CONNECT_PRIMITIVE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff0c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	live		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Each RCP client may have different media sources and media receivers. To connect these media channels, a low level connect/disconnect primitive is implemented. These low level primitives can be grouped together to more complex commands, which can connect more than one media channel at the same time. The primitives are designed to be asserted by media sources or receivers itself. Manager software should use the high level commands (see next chapter).

By sending a CONNECT GET to the remote endpoint, this endpoint will start sending media to the requested address. For example, a VideoJet receiver is registered at a VideoJet transmitter. On sending a ‘CONNECT GET video to port 1234’ to the transmitter, the transmitter starts to send video to the appropriate port at the receiver. The endpoint requesting the media stream (the VideoJet receiver) must be prepared to receive the stream from the network. The coding bits for video must be set according to the receivers capability to decode the data stream. E.g. a Decoder capable of decoding MPEG2 and MPEG4 should set all these bits.

By sending a CONNECT PUT to the remote endpoint, the endpoint will prepare itself for receiving data from the one initiating the request. For example, a VideoJet transmitter is registered at a VideoJet receiver. On sending a ‘CONNECT PUT video to port 2222’ to the receiver, the receiver will prepare its network stack to receive data from the appropriate port. The endpoint initiating the media stream (the VideoJet transmitter) must start sending the media stream to the network. The coding bits for video must be set to a value suitable to the current encoding parameters inside the VideoJet. E.g. if a video encoder is set to encode MPEG4, only a PUT request having these bits set is accepted. The coder parameter is used to address a specific encoder/decoder inside the VideoJet. As the VideoJet may have a Dual-stream feature, one physical line input might be connected to more than one encoder. By setting a coder to another value than 0, the VideoJet only tries this specific coder to establish a connection. When the coder value is set to 0, all coders inside the system will be checked. Each connect primitive will result in a Session ID for maintaining and controlling the media stream with other RCP commands. The Session ID can be treated as unique on the entire system.

By sending a DISCONNECT supplied with a valid Session ID, the receiver of the command will stop sending/receiving to/from the media stream. The endpoint initiating the DISCONNECT command must stop receiving/sending from/to the media stream.

Payload Structure

© Copyright 2001-2021 Bosch Security Systems

Item [0] (see description)
...
Item [N] (see description)

Item

16			32
Method 1 Byte	**Media** 1 Byte	**Reserved** 1 Byte	**Header Flags** 1 Byte
Reserved 4 Bytes			
Media Descriptor [0] (see description)			
...			
Media Descriptor [N] (see description)			
8		24	

Method

Values:

Get	0x00
Put	0x01
Key Transport	0xE0

Media

Values:

Video	1
Audio	2
Data	3

Header Flags

	Mask	Name	Description
Bit 7	0x80	Multi ROI capable	This Bit signals that a client is able to provide a session ID to steer multiple ROIs (region of interrest) with E-PTZ.

© Copyright 2001-2021 Bosch Security Systems

Bit 6	0x40	Media over ssl	This Bit is set in the response to signal that the device supports 'media over ssl'.
Bit 5	0x20	Transcoder	Set this bit to instanciate and establish a connection to the transcoder.
Bit 4	0x10	Dry Connect	If this bit is set a replay connection is established (and a session id to control the replay connection is returned). The replay source is not connected. This can be done using the CONF_RCP_CONNECT_SALVO command.
Bit 3	0x08	Media Key	Media key request (when using GET) / Media key is appended at the end of this section (when using PUT)
Bit 2	0x04	Signalling	Optional signalling for streaming (for correct connection list display)
Bit 1	0x02	Free Running Connection	When this bit is set, the connection does not need to be re-triggered by the command CONF_RCP_CONNECTIONS_ALIVE.
Bit 0	0x01	Do reverse Login	This will control whether the called server will perform a reverse RCP Login.

Media Descriptor

'Media Descriptor' payload for 'Media' = 'Video' (1)

16 32

MEP 1 Byte	**Flags** 1 Byte	**MTA Port** 2 Bytes
MTA IP Address 4 Bytes		
Coder 1 Byte	**Line** 1 Byte	**MCTA Port** 2 Bytes
MCTA IP Address 4 Bytes		
Coding 2 Bytes		**Resolution** 2 Bytes
Linked Coder 1 Byte	**Linked Line** 1 Byte	**Flags2** 2 Bytes
Key (conditional) (see description)		

8 24

MEP

Media Encapsulation Protocol.

© Copyright 2001-2021 Bosch Security Systems

Values:

RTP over UDP	0x01
RTP over TCP	0x04

Note: When using RTP over TCP, media channels must be established as described in 'RCP Protocol Procedure / Receiving media data'.

Flags

	Mask	Name	Description
Bit 6	0x40	ExclusiveROI	Request an exclusive (ROI) stream. (Only applicaple if device is in multi ROI mode and an exclusive coder is still available).
Bit 2	0x04	UseFlags2	Flags2 are used
Bit 1	0x02	RelativeNumber	Use relative to line coder addressing.
Bit 0	0x01	Substitute	Substitute connection (if an RX connection to the requested interface is already established, the old connection will be disconnected and substituted by a new connection coming with this request).

MTA Port

Media Transport Address - Network port.

MTA IP Address

Media Transport Address IP Address

Values:

0.0.0.0	VideoJet will take address from the incoming TCP header
224.0.0.0	VideoJet will take its configured multicast group IP

Coder

Specifies the internal coder number; when set to 0, the appropriate coder corresponding to the given line number and/or coding parameter is chosen by the VideoJet itself.

Line

Identifies the Line Input/Output channel for the specified coding engine.

MCTA Port

Media Control Transport Address Network Port. Currently unused.

MCTA IP Address

Media Control Transport Address IP Address. Currently unused.

© Copyright 2001-2021 Bosch Security Systems

Coding

	Mask	Name	Description
Bit 15	0x8000	Mpeg2Prog	MPEG-2 (program stream)
Bit 14	0x4000	RecordedMedia	Recorded Media
Bit 9	0x0200	H265	H265
Bit 7	0x0080	Jpeg	JPEG
Bit 6	0x0040	H264	H264
Bit 4	0x0010		
Bit 3	0x0008	Mpeg2	MPEG-2 (only video)
Bit 2	0x0004	Mpeg4	MPEG-4 (elementary stream)

Resolution [Obsolete]

	Mask	Name
Bit 11	0x0800	WD432
Bit 10	0x0400	WD288
Bit 9	0x0200	WD144
Bit 8	0x0100	HD1080
Bit 7	0x0080	HD720
Bit 6	0x0040	VGA
Bit 5	0x0020	QVGA
Bit 3	0x0008	4CIF
Bit 2	0x0004	2CIF
Bit 1	0x0002	CIF
Bit 0	0x0001	QCIF

Linked Coder

Use this field for signalling the callers local coder in the connection list of the called device.

Linked Line

Use this field for signalling the callers local line in the connection list of the called device.

Flags2

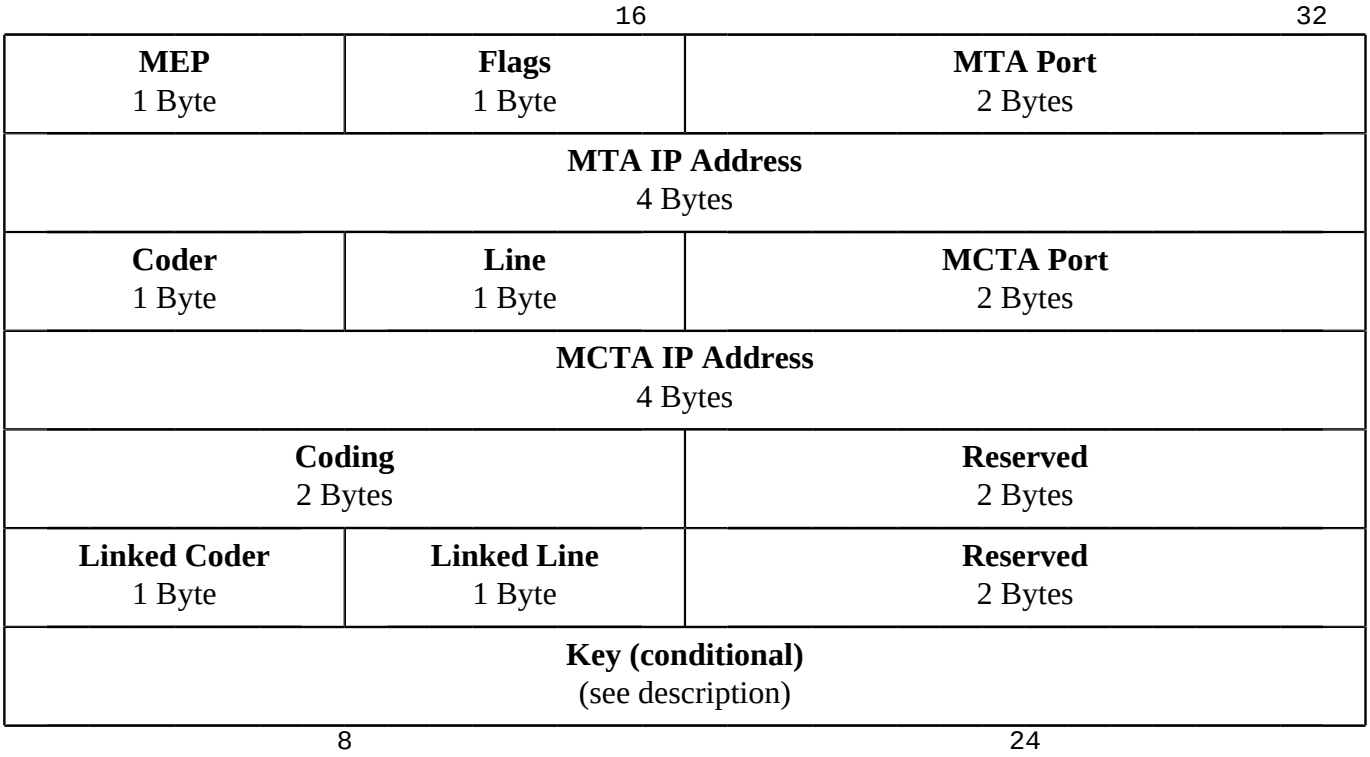
Only evaluated if Bit3 of Flags is set. For internal usage

Key (conditional)

See description of 'Key transport'

© Copyright 2001-2021 Bosch Security Systems

'Media Descriptor' payload for 'Media' = 'Audio' (2)



MEP

Media Encapsulation Protocol.

Values:

RTP over UDP	0x01
RTP over TCP	0x04

Flags

	Mask	Name	Description
Bit 1	0x02	RelativeNumber	Use relative to line coder addressing.
Bit 0	0x01	Substitute	Substitute connection (if an RX connection to the requested interface is already established, the old connection will be disconnected and substituted by a new connection coming with this request).

MTA Port

Media Transport Address - Network port.

MTA IP Address

Media Transport Address IP Address

© Copyright 2001-2021 Bosch Security Systems

Values:

0.0.0.0	VideoJet will take address from the incoming TCP header
224.0.0.0	VideoJet will take its configured multicast group IP

Coder

Specifies the internal coder number; when set to 0, the appropriate coder corresponding to the given line number and/or coding parameter is chosen by the VideoJet itself.

Line

Identifies the Line Input/Output channel for the specified coding engine.

MCTA Port

Media Control Transport Address Network Port. Currently unused.

MCTA IP Address

Media Control Transport Address IP Address. Currently unused.

Coding

	Mask	Name	Description
Bit 15	0x8000	Mpeg2Prog	MPEG-2 (program stream)
Bit 14	0x4000	RecordedMedia	Recorded Media
Bit 4	0x0010	L16	L16 (sampling rate: 16 kHz, rtp clock rate: 90 kHz)
Bit 3	0x0008	L16 16kHz	L16_16kHz (sampling rate: 16 kHz, rtp clock rate: 16 kHz)
Bit 2	0x0004	G711 8kHz	G.711_8kHz (sampling rate: 8 kHz, rtp clock rate: 8 kHz)
Bit 1	0x0002	AAC	
Bit 0	0x0001	G711	G.711 (sampling rate: 8 kHz, rtp clock rate: 90 kHz)

Linked Coder

Use this field for signalling the callers local coder in the connection list of the called device.

Linked Line

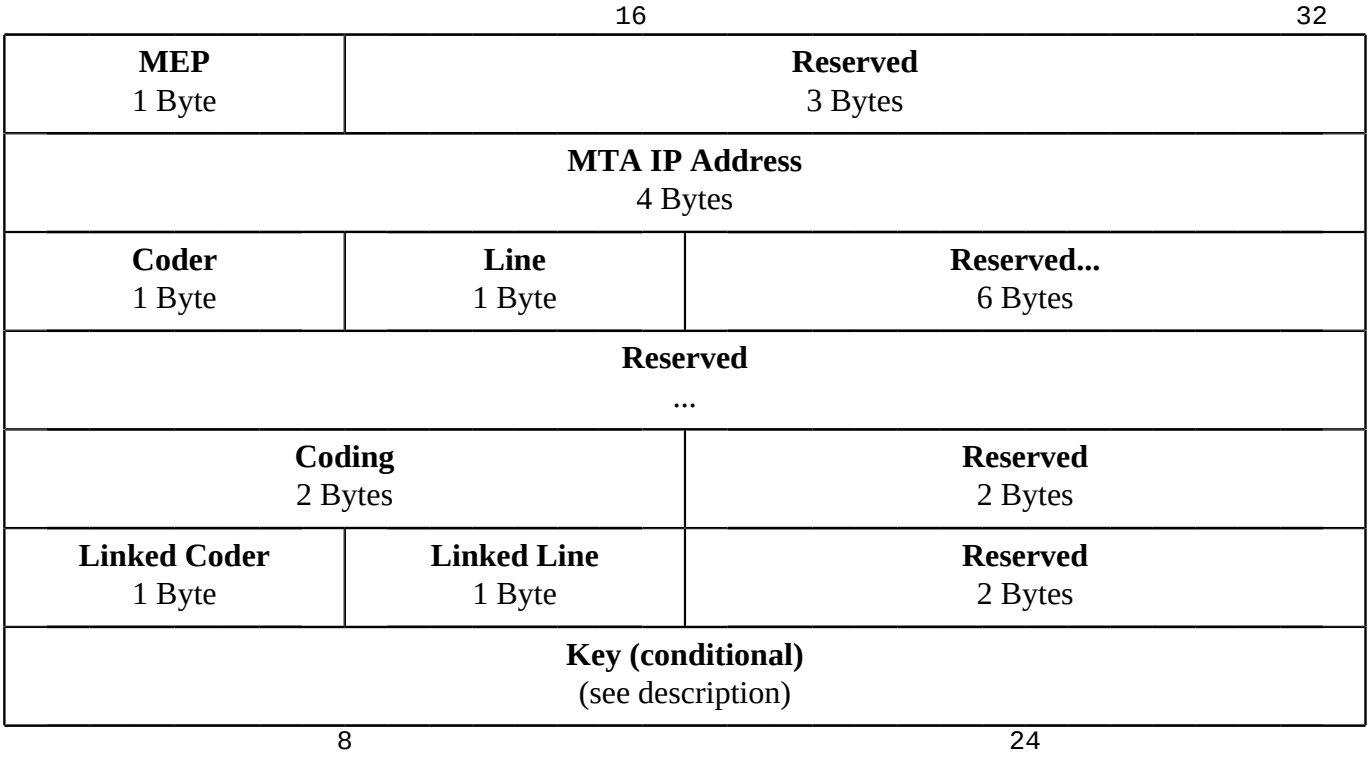
Use this field for signalling the callers local line in the connection list of the called device.

Key (conditional)

See description of 'Key transport'

© Copyright 2001-2021 Bosch Security Systems

'Media Descriptor' payload for 'Media' = 'Data' (3)



MEP

Media Encapsulation Protocol.

Values:
RCP Intrinsic 0x02

MTA IP Address

Media Transport Address IP Address

Values:
0.0.0.0 VideoJet will take address from the incoming TCP header

Coder

Informational

Line

Identifies the Line Input/Output channel for the specified coding engine.

© Copyright 2001-2021 Bosch Security Systems

Coding

	Mask	Name	Description
Bit 14	0x4000	RecordedMedia	Recorded Media
Bit 0	0x0001	RCP	

Linked Coder

Use this field for signalling the callers local coder in the connection list of the called device.

Linked Line

Use this field for signalling the callers local line in the connection list of the called device.

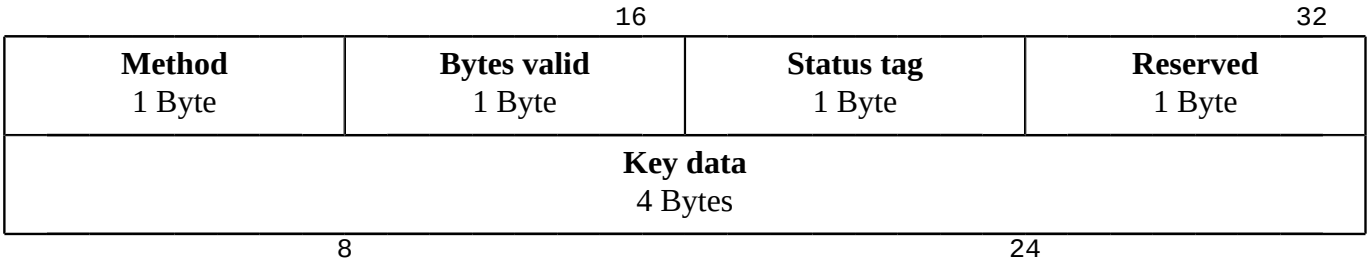
Key (conditional)

See description of 'Key transport'

'Key transport' structure

Key [0] (see description)
...
Key [N] (see description)

Key



Method

Must be set to 'Key transport' 0xE0

Bytes valid

The number of valid key bytes in this header

© Copyright 2001-2021 Bosch Security Systems

Status tag

Must be set to 'Key transport tag' 0xE0

Key data

The key payload

The media key is transferred using a sequence of key headers all with the method field set to 0xE0. The end of the key section is marked by a header with less than 4 bytes (or zero) indicated in the bytes valid field. The typical key length is 16 bytes (128 bit AES). In this case, four full key headers followed by one empty header must be transferred (4 x 4 bytes valid + 1 x 0 bytes valid = 16 bytes). For backward compatibility, the header method 0xE0 will be skipped by units with former firmwares versions. The key is located and stored on each media source (using the CONF_CRYPT_KEY* commands); When the system is configured to publish its keys (see CONF_PUBLISH_MEDIA_KEYS), then the media sources must fill in the secret by sending (PUT method) or replying to (GET method) the connect primitive. For example, when sending a GET connect primitive to a device and the key transport is selected, then the device will fill in its key in the reply to the connect primitive.

Note: Be sure to use a secure RCP connection, the keys in the connect primitive are tranferred in plain

Response Payload Structure

Item [0] (see description)
...
Item [N] (see description)

Item

16			32
Method 1 Byte	**Media** 1 Byte	**Status** 1 Byte	**Flags** 1 Byte
Reserved 4 Bytes			
Media Descriptor [0] 4 Bytes			
...			
Media Descriptor [N] 4 Bytes			
8		24	

© Copyright 2001-2021 Bosch Security Systems

Method

Values:

Get	0
Put	1

Media

Values:

Video	1
Audio	2
Data	3

Status

Values:

InterfaceUnavailable	0	Interface not available (Remark: former version may return this error in case of unknown method/media type)
AccessGranted	1	Access to this interface granted
AccessDenied	2	Access to this interface rejected
InvalidIdOrConnectionLoss	3	Session ID invalid or connection no longer active
CodingIncompatible	4	Coding parameters incompatible
ExternalData	5	Interface data will be supplied by another media descriptor
UnknownMethod	8	The method field in the header does not carry a known method
UnknownMedia	9	The media field in the header does not carry a known media type

Note: Any wildcards used in the transport addresses will be replaced by the real used network addresses. This command is NOT readable; to gain information on active connection refer to the ACTIVE_CONNECTION_LIST command. The Media Control sockets have currently no function

© Copyright 2001-2021 Bosch Security Systems

2.123 CONF_CONNECT_TO

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xffcc	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	live		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

A message will be generated if all requested channels are established or respectivly failed; if a channel fails, the appropriate bit will be cleared in the channel section.

Payload Structure

16		32
Destination IP Address 4 Bytes		
Reserved 1 Byte	**Line** 1 Byte	**Flags** 2 Bytes
Local Coder 1 Byte	**Local Line** 1 Byte	**Put Channel** 2 Bytes
Remote Coder 1 Byte	**Remote Line** 1 Byte	**Get Channels** 2 Bytes
Appendix (optional) (see description)		
8		24

Destination IP Address

The reception of this command will force the host to connect to the mentioned destination IP address.

Line [Obsolete]

This parameter carries the number of the remote input (for get channels) or the local input (for put channels); For connection to the HDD, the Line byte must carry the partition number of the HD. This parameter is updated by the local line/coder and remote line/coder settings.

© Copyright 2001-2021 Bosch Security Systems

Flags

	Mask	Name	Description
Bit 13	0x2000	H265	Request video mode H.265
Bit 12	0x1000	AllowAAC	Allow audio mode AAC
Bit 11	0x0800	AllowL16	Allow audio mode L16 (setting this bit to 0 uses G711 (default, previous behaviour), setting to 1 allows G.711 and L16 - both with 90kHz timestamp clock)
Bit 10	0x0400	Jpeg	Request video mode JPEG
Bit 9	0x0200	H264	Request video mode H.264
Bit 8	0x0100	UseSSL	Use SSL for the RCP control connection; if no destination port is specified, the remote port defaults to 443 (HTTPS) in case of SSL is requested otherwise 80(HTTP)
Bit 7	0x0080	NoKeyExchange	Supress automatic media key exchange
Bit 6	0x0040	ConnectVCA	Connect a VCA meta data stream
Bit 4	0x0010	ConnectHDD	Connect to the HDD to receive a recorded media stream
Bit 3	0x0008	ForceTCP	Force the use of TCP as transportation protocol
Bit 2	0x0004	Mpeg2	Request video mode MPEG-4
Bit 1	0x0002	Mpeg4	Request video mode MPEG-2
Bit 0	0x0001	Substitute	Substitute an existing connection

Note: Only one video standard (MPEG2/4) can be used; setting all bits will result in best currently available mode.

Local Coder

This parameter carries the number of the local coder number. A wildcard of ‘0’ will result in first match.

Local Line

This parameter carries the number of the local video line number. A wildcard of ‘0’ will result in first match.

Put Channel

	Mask	Name
Bit 2	0x0004	Data
Bit 1	0x0002	Audio
Bit 0	0x0001	Video

Remote Coder

This parameter carries the number of the remote coder number (at the destination IP). A wildcard of ‘0’ will result in first match.

© Copyright 2001-2021 Bosch Security Systems

Remote Line

This parameter carries the number of the remote video line number (at the destination IP). A wildcard of ‘0’ will result in first match.

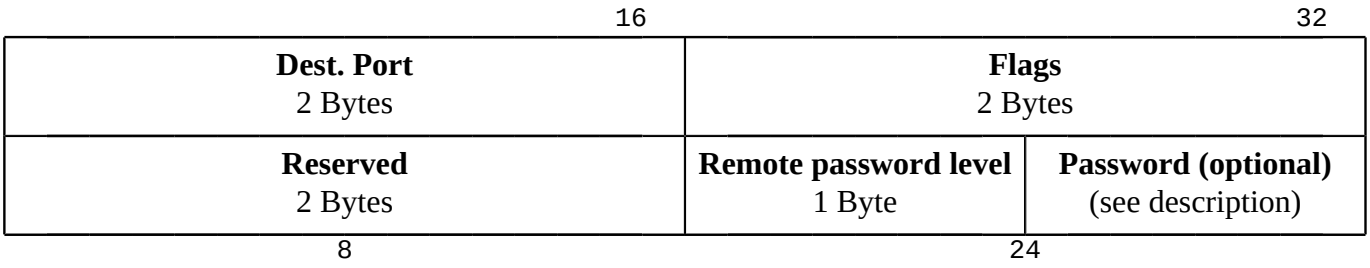
Get Channels

	Mask	Name
Bit 2	0x0004	Data
Bit 1	0x0002	Audio
Bit 0	0x0001	Video

Note: For audio connections, the local and remote line parameter are taken from the video settings.

Appendix (optional)

If this command is extended with the optional appendix, the remote port number for RCP login must be specified. This can be either 1756 for the normal RCP port or any available HTTP port at the remote host. When a port number different to 1756 is used, the login will use a HTTP tunnelling.



Dest. Port

This parameter carries the number of the remote TCP port number used for RCP login.

Flags

	Mask	Name	Description
Bit 0	0x0001	IsURL	Remote password field is url (supports urls like rtsp://user:pw@160.10.9.222:5540/rtsp_tunnel)

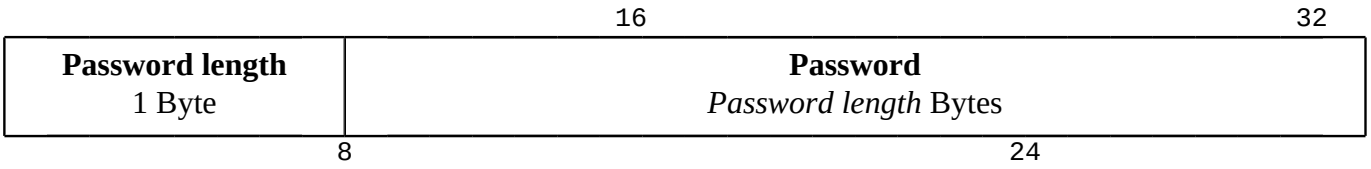
Remote password level

Values:

No protection	0
User	1
Service	2
Live	3

© Copyright 2001-2021 Bosch Security Systems

Password (optional)



© Copyright 2001-2021 Bosch Security Systems

2.124 CONF_CONNECT_URL

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a1e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the connect url	
Write	p_string	service	Set the connect url	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.125 CONF_CONT_VIPROC_CONFIG_EDITING

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3a	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	iva	Configuration of a config stops automatically after 20 seconds. This command can be used to keep configuration active for another 20 seconds. If the specified config is currently not in configuration mode, the command replys with an error.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.126 CONF_CPU_COUNT

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a09	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of cpus	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.127 CONF_CPU_LOAD

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0a	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

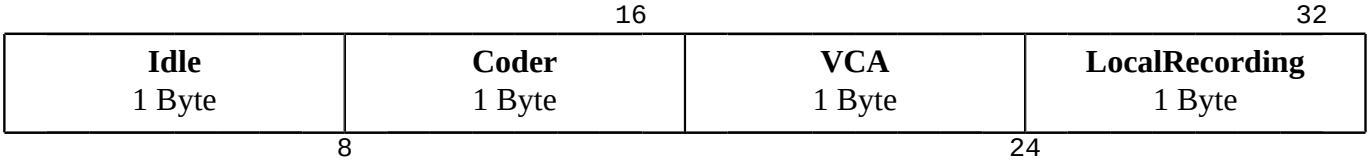
Description

Read the CPU load information from a device.

Num Descriptor Values

Host	1
CoProc	2

Payload Structure



Idle

Idle time of CPU in percent.

Coder

Coder time of CPU in percent. Only supported on CPP6/CPP7/CPP7.3, '0' otherwise

VCA

VCA time of CPU in percent. Only supported on CPP6/CPP7/CPP7.3, '0' otherwise

LocalRecording

Local recording time of CPU in percent. Only supported on CPP6/CPP7/CPP7.3, '0' otherwise

© Copyright 2001-2021 Bosch Security Systems

2.128 CONF_CPU_LOAD_CODER

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a07	host=1	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Cpu load coder in percent	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.129 CONF_CPU_LOAD_IDLE

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a06	host=1	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Cpu load idle in percent	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.130 CONF_CPU_LOAD_VCA

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a08	host=1	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Cpu load vca in percent	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.131 CONF_CRYPT_KEY_GENERATE_ALL

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a13	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Deprecated - use SRTP instead for video/audio encryption. Returns flag=0 when all key are cleared otherwise flag=1	
Write	f_flag	service	Deprecated - use SRTP instead for video/audio encryption. Generates a random key(flag=1) or clears all(flag=0) keys for AES encryption for all media encoders	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.132 CONF_CRYPT_KEYAUDIO_DEC

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ff	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Get key for AES Audio data encryption/ decryption	
Write	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Set key for AES Audio data encryption/ decryption, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

This description is valid for the the commands:

- CONF_CRYPT_KEYVIDEO_ENC
- CONF_CRYPT_KEYVIDEO_DEC
- CONF_CRYPT_KEYAUDIO_ENC
- CONF_CRYPT_KEYAUDIO_DEC

Set all Key- and all Salt- Bytes to zero, if no encryption/decryption is wished.

Num Descriptor Values

coder	Any	The number of the coder instance (absolute)

Payload Structure

16	32
Key... 16 Bytes	
Key ...	
Key ...	
Key ...	

8 24

© Copyright 2001-2021 Bosch Security Systems

Key

The 16 Byte Masterkey for AES encryption/decryption of data

© Copyright 2001-2021 Bosch Security Systems

2.133 CONF_CRYPT_KEYAUDIO_ENC

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09fe	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Get key for AES Audio data encryption/ decryption	
Write	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Set key for AES Audio data encryption/ decryption, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

This description is valid for the the commands:

- CONF_CRYPT_KEYVIDEO_ENC
- CONF_CRYPT_KEYVIDEO_DEC
- CONF_CRYPT_KEYAUDIO_ENC
- CONF_CRYPT_KEYAUDIO_DEC

Set all Key- and all Salt- Bytes to zero, if no encryption/decryption is wished.

Num Descriptor Values

coder	Any	The number of the coder instance (absolute)

Payload Structure

16	32
Key... 16 Bytes	
Key ...	
Key ...	
Key ...	

8 24

© Copyright 2001-2021 Bosch Security Systems

Key

The 16 Byte Masterkey for AES encryption/decryption of data

© Copyright 2001-2021 Bosch Security Systems

2.134 CONF_CRYPT_KEYVIDEO_DEC

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a00	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Get key for AES Video data encryption/ decryption	
Write	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Set key for AES Video data encryption/ decryption, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

This description is valid for the the commands:

- CONF_CRYPT_KEYVIDEO_ENC
- CONF_CRYPT_KEYVIDEO_DEC
- CONF_CRYPT_KEYAUDIO_ENC
- CONF_CRYPT_KEYAUDIO_DEC

Set all Key- and all Salt- Bytes to zero, if no encryption/decryption is wished.

Num Descriptor Values

coder	Any	The number of the coder instance (absolute)

Payload Structure

16	32
Key... 16 Bytes	
Key ...	
Key ...	
Key ...	

8 24

© Copyright 2001-2021 Bosch Security Systems

Key

The 16 Byte Masterkey for AES encryption/decryption of data

© Copyright 2001-2021 Bosch Security Systems

2.135 CONF_CRYPT_KEYVIDEO_ENC

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ed	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Get key for AES Video data encryption/ decryption	
Write	p_octet	service	Deprecated - use SRTP instead for video/audio encryption. Set key for AES Video data encryption/ decryption, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

This description is valid for the the commands:

- CONF_CRYPT_KEYVIDEO_ENC
- CONF_CRYPT_KEYVIDEO_DEC
- CONF_CRYPT_KEYAUDIO_ENC
- CONF_CRYPT_KEYAUDIO_DEC

Set all Key- and all Salt- Bytes to zero, if no encryption/decryption is wished.

Num Descriptor Values

coder	Any	The number of the coder instance (absolute)

Payload Structure

16	32
Key... 16 Bytes	
Key ...	
Key ...	
Key ...	

8 24

© Copyright 2001-2021 Bosch Security Systems

Key

The 16 Byte Masterkey for AES encryption/decryption of data

© Copyright 2001-2021 Bosch Security Systems

2.136 CONF_CSRF_PROTECTION_ENABLED

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d0d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Read if the CSRF (cross-site request forgery) protection is enabled. #ValueList: 1= protection active (default); 0= protection disabled	
Write	f_flag	service	Enable the CSRF (cross-site request forgery) protection #ValueList: 1= protection active (default); 0= protection disabled	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.137 CONF_DATA_COPY_JOB_START

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b32	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Start copy job,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command starts a copy job, given by first the source descriptor and second the destination descriptor. If starting the job was ok, the status will be reported by CONF_DATA_COPY_JOB_STATUS messages or can be querried by the same comand. The return payload structure is the same as the CONF_DATA_COPY_JOB_STATUS payload and contains the job id on success. This id can be used for stopping the job, query the state or to identify it in the status messages payload.

Payload Structure

16	32
Source Descriptor (see description)	
Destination Descriptor (see description)	
8	24

Source Descriptor

See description of 'Source/Destination Descriptor'

Destination Descriptor

See description of 'Source/Destination Descriptor'

'Source/Destination Descriptor' structure

16	32
Offset 4 Bytes	
Length 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Type 2 Bytes	Type spec. parameter len 2 Bytes
Type spec. parameter (see description)	

8 24

Offset

The offset in lba sized units (= 512 bytes)

Length

Amount of data to copy in lba sized units (= 512 bytes), not relevant for destination descriptor

Full Copy	0
Length	1 - n

Type

Interface Test	0
Span on ISCSI	1
File	2
Raw ISCSI lun	3

Type spec. parameter len

Length of the following additional parameters.

Type spec. parameter

'Type spec. parameter' payload for 'Type' = 'Interface Test' (0)

Fakes the access to a source/destination and can be used for example to test the interface and copy job engine. No further parameters are needed.

'Type spec. parameter' payload for 'Type' = 'Span on ISCSI' (1)

16 32

Target Id 4 Bytes		
Target index 1 Byte	Lun index 1 Byte	Span index 2 Bytes
File type 1 Byte	Reserved 1 Byte	Timeout 2 Bytes

8 24

Access to a span on an iscsi lun. This type can be used for example to copy a span. The parts of a span (unit, lock, mgr header, data) has to be copied in seperate copy jobs.

© Copyright 2001-2021 Bosch Security Systems

Target Id

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

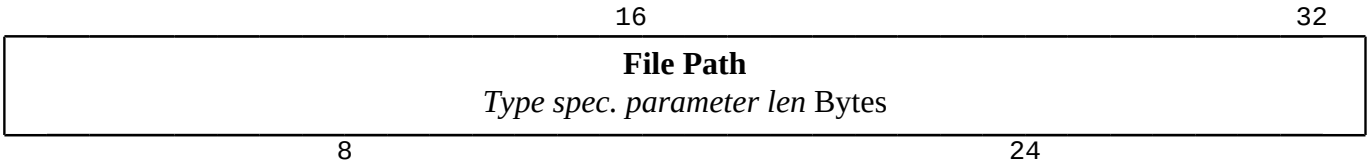
File type

Lock Header	1
Unit Header	2
Manager Header	3
Recording Data	4

Timeout

Default	0	Default, using default iscsi timeouts
Timeout	1 - 65534	Timeout in seconds
Never	65535	Never timeout

'Type spec. parameter' payload for 'Type' = 'File' (2)

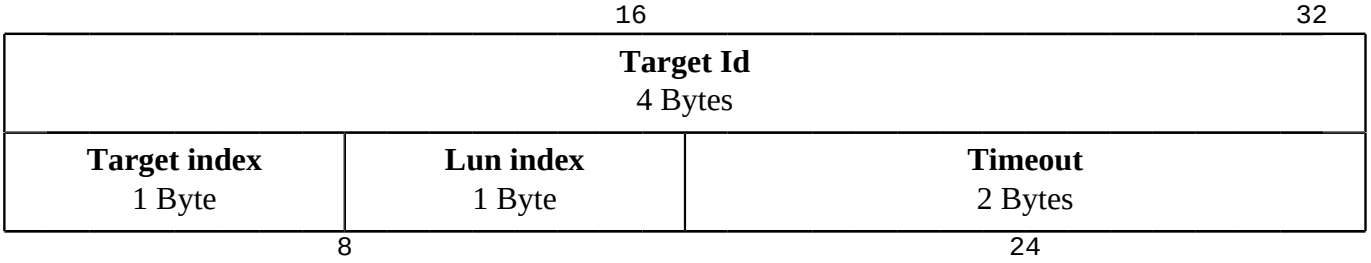


Access to a file. Can be used for example to copy a span to a file. If the file is the source, it has to exist. If the file is destination it will be created if it doesn't exist. Parameter length is the path length including zero termination and stuffing bytes in order to be 4 byte aligned. Size is at least 4 bytes.

File Path

Zero terminated ascii string specifing the path and file name of a source/destination

'Type spec. parameter' payload for 'Type' = 'Raw ISCSI lun' (3)



Access to a complete iscsi lun. This type can be used for example to make a one to one copy of a complete lun.

© Copyright 2001-2021 Bosch Security Systems

Target Id

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

Timeout

Default	0	Default, using default iscsi timeouts
Timeout	1 - 65534	Timeout in seconds
Never	65535	Never timeout

© Copyright 2001-2021 Bosch Security Systems

2.138 CONF_DATA_COPY_JOB_STATUS

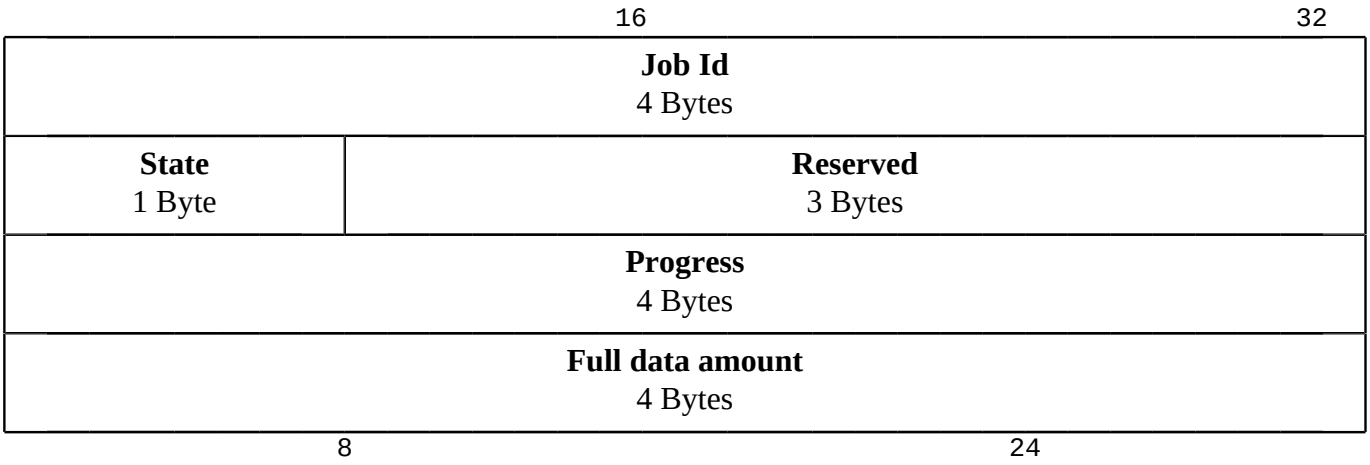
[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b34	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the status of a copy job,	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message reports the status of the copy job and will be send on each status change or several times while in state "running". The status can also be queried by sending this command in read direction.

Payload Structure



Job Id

Id of a data copy job

State

State of the copy Job

Initializing	0
Running	1
Finished	2
Failed	3
Stopped	4

© Copyright 2001-2021 Bosch Security Systems

Progress

Progress in lba units (= 512 bytes)

Full data amount

Full data amount in lba units (= 512 bytes) of the copy job. Progress has to reach this value in order to finish successfully.

© Copyright 2001-2021 Bosch Security Systems

2.139 CONF_DATA_COPY_JOB_STOP

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b33	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	service	Requests stop on a data copy job if exists, otherwise fails, payload: job id	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.140 CONF_DATE_DAY

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0028	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet p_string	minimal	Read the day of month	
Write	t_octet p_string	service	Set the day of month, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.141 CONF_DATE_MONTH

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0029	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet p_string	minimal	Read the month	
Write	t_octet p_string	service	Set the month, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.142 CONF_DATE_WDAY

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0027	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	\Sunday\" ... \"Saturday\"; read the weekday according to the systems date setting	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.143 CONF_DATE_YEAR

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word p_string	minimal	Read the year	
Write	t_word p_string	service	Set the year; century needed, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.144 CONF_DAY_LIGHT_SAVE_TIME

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0988	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Read the currently used day light save time offset in seconds	
Write	t_int	service	Write the day light save time offset in seconds; the device updates this from DAY_LIGHT_SAVE_TIME_TABLE to show current offset used	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.145 CONF_DAY_LIGHT_SAVE_TIME_TABLE

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0987	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read 20 entries for daylight save time: UTC time in sec since 2000 (4byte) and associated offset in sec (4 signed bytes), unused entries have to be filled with zero	
Write	p_octet	service	Write 20 entries for daylight save time: UTC time in sec since 2000 (4byte) and associated offset in sec (4 signed bytes), unused entries have to be filled with zero	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.146 CONF_DEC_SHOW_FREEZE

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Freeze string: 0=off, 1=on, n=on (n * 40ms delayed)	
Write	t_octet	service	Freeze string: 0=off, 1=on, n=on (n * 40ms delayed)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.147 CONF_DECODER_DELAY

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bbe	decoder line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get additional delay for decoder in milliseconds	
Write	t_dword	service	Set additional delay for decoder in milliseconds	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.148 CONF_DECODER_LAYOUT

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09a2	output line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	First word (2bytes) is the layout, followed by the relative coder list in bytes (000101 for first monitor singleview, 000201020304 for quadview with chronological order)	
Write	p_octet	user	First word (2bytes) is the layout, followed by the relative coder list in bytes (000101 for first monitor singleview, 000201020304 for quadview with chronological order)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Payload Structure

Layout 2 Bytes		**Reserved** 2 Bytes
Flags 1 Byte	**YUV** 3 Bytes	
Entry [0] (see description)		
...		
Entry [N] (see description)		

Layout

Single	1
Quad	2
Nine	3
Sixteen	4
Multiview	M

© Copyright 2001-2021 Bosch Security Systems

Flags

	Mask	Name	Description
Bit 0	0x01	useRGB	Signals RGB instead of YUV

YUV

Background YUV

Entry

16			32
len 1 Byte	**flags** 1 Byte	**xPosition** 1 Byte	**yPosition** 1 Byte
width 1 Byte	**height** 1 Byte	**inst** 1 Byte	**reserved** 1 Byte
hPos 2 Bytes		**vPos** 2 Bytes	
size 2 Bytes		**reserved** 2 Bytes	
8		24	

len

Entry len

flags

Bit 0= flip, bit 1=mirror, bit 2=allFrames

xPosition

X position in grid

yPosition

Y position in grid

width

Width in grid

height

Height in grid

© Copyright 2001-2021 Bosch Security Systems

inst

Instance

hPos

Optional ptz hpos (see CONF_ROI)

vPos

Optional ptz vpos (see CONF_ROI)

size

Optional ptz size(see CONF_ROI)

© Copyright 2001-2021 Bosch Security Systems

2.149 CONF_DECODER_LAYOUT_LIST

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09a1	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List of supported layouts; payload is a list of n WORDs. One WORD per supported layout (0001=single view, 0002=quad view, 0004=sixteen, 0077=multi view)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.150 CONF_DECODER_MODE

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0812	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Decoder mode #ValueList: 0= low delay; 1= A/V sync;	
Write	t_octet	service	Decoder mode #ValueList: 0= low delay; 1= A/V sync;	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

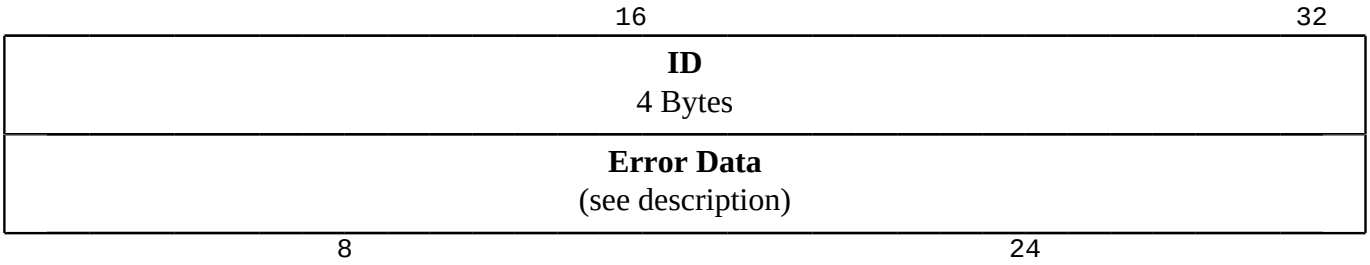
© Copyright 2001-2021 Bosch Security Systems

2.151 CONF_DECODING_ERROR

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bbf	coder	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

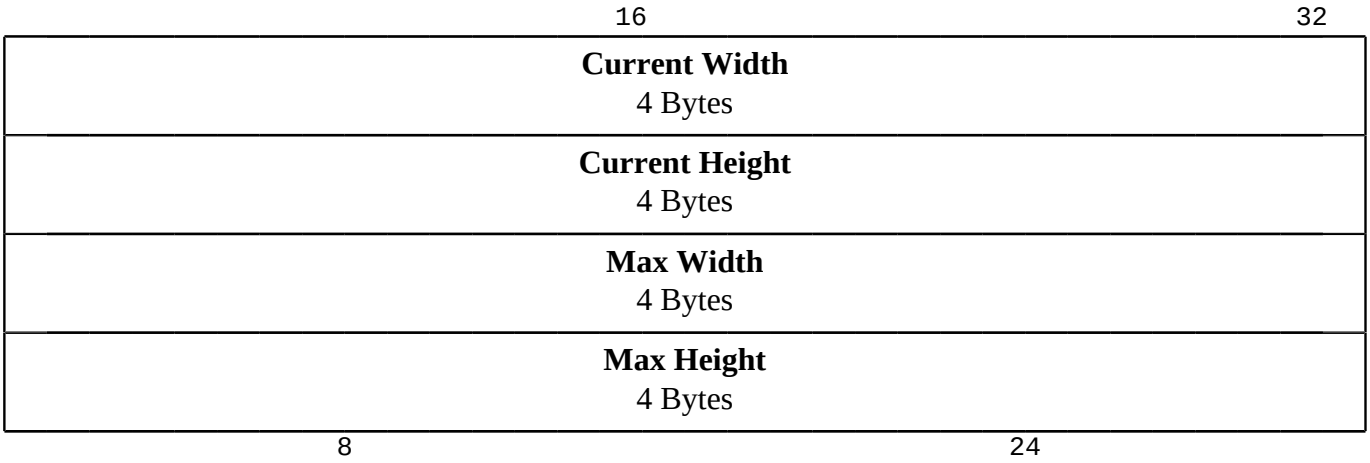
Payload Structure



Error Data

'Error Data' payload for 'ID' = '1'

H.264 video resolution exceeded decoder limit.



'Error Data' payload for 'ID' = '2'

Unknown JPEG format.

© Copyright 2001-2021 Bosch Security Systems

2.152 CONF_DEFAULT_CAM

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01af	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Reads the default camera for alarm connections	
Write	t_octet	service	Defines the default camera for alarm connections	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.153 CONF_DEFAULT_CONNECTION_MODE

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0289	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the default connection mode: 0=MPEG2, 1=MPEG4, 2=MPEG4/ AVC, 3=JPEG, 4=ANY	
Write	t_octet	service	Sets the default connection mode: 0=MPEG2, 1=MPEG4, 2=MPEG4/ AVC, 3=JPEG, 4=ANY	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.154 CONF_DEFAULTS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x093d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Sets config to default values except IP, subnet and gateway and resets the device. ATTENTION: Wait till board has fully rebooted before switching of power, to ensure that defaults are applied completly. Also propagade this to the user, that he has to wait until the reboot is complete.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.155 CONF_DELETE_CAM_REC_SPANS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b5d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Delete recording spans of a camera from a span manager on a managed lun.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

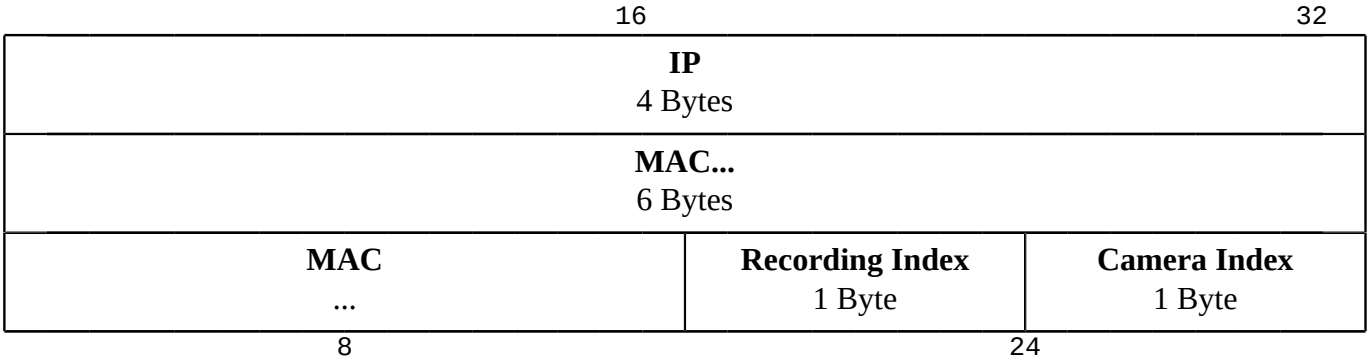
This command is used for deleting spans of a specific recording line and index containing recordings between a given time interval. The flag "delete enclodes spans only" chooses wether a span shall be deletet if it is completetly enclosed by the time interval or if it only overlaps the interval. The flag "force delete" will delete all spans of the cam independend of the state of the span. That means even if the cam is still recording on a span, the span will be deleted. So use this option very carefully, as it could lead to inconsitencies. If the command return successfully, it doesn't mean, that the deletion is completetd, it just tells the caller, that a asychonous job for deletion was invoked. The command has to be send to the storage lun managing device.

Payload Structure

16	32
Recording Camera (see description)	
Lun Address (see description)	
From 4 Bytes	
To 4 Bytes	
Flags 4 Bytes	
8	24

© Copyright 2001-2021 Bosch Security Systems

Recording Camera



IP

The IP address of the recording device.

MAC

The hardware address of the recording device. This field may be zero if you search for recordings without considering the mac.

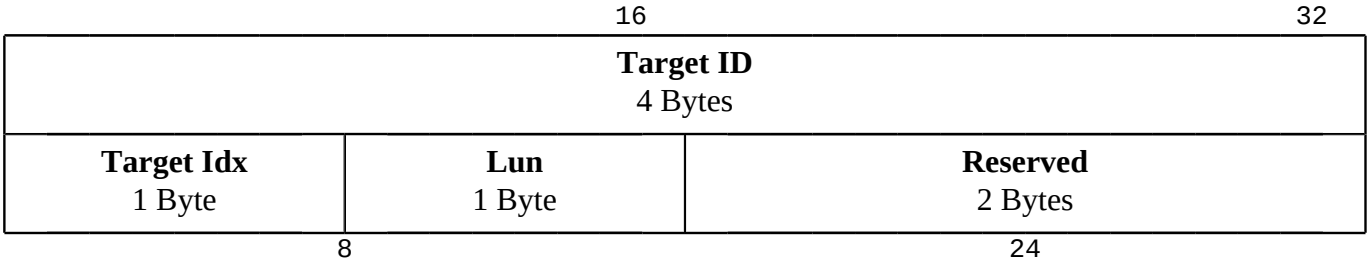
Recording Index

Primary Recording	1
Secondary Recording	2

Camera Index

The camera index

Lun Address



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

© Copyright 2001-2021 Bosch Security Systems

Target Idx

The index of the target.

Lun

The logical unit number.

From

Start time of the interval in Seconds since 2000 local time for deleting.

To

End time of the interval in Seconds since 2000 local time for deleting.

Flags

	Mask	Name	Description
Bit 1	0x00000002	Force	ForceDeletion
Bit 0	0x00000001	EnclosedOnly	Delete enclosed spans only

© Copyright 2001-2021 Bosch Security Systems

2.156 CONF_DEVELOPER_MODE_ALLOWED

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cfd	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	0=developer mode is not allowed; 1=developer mode is allowed	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.157 CONF_DEVICE_CAPABILITIES

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b60	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns capabilities of the device in a tagged format	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Number of tagged elements 4 Bytes
Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

16 32

Tag = Tag ID 2 Bytes	**Length** 2 Bytes
Tag Payload	

8 24

Tag Name	Tag ID	Payload Type	Description
DEVICE_TYPE_TAG	1	t_octet	
PTZ_CAMERA_TAG	2	t_octet	ptz capabilities; ptz capabilities are available e.g. if the device is a dome camera, if bilinx is supported and a bilinx camera is connected to the device or if a serial dome protocol is set
ROI_CAMERA_TAG	3	t_octet	region of interest available
AUTOTRACKER_TAG	4	t_octet	autotracker available

© Copyright 2001-2021 Bosch Security Systems

Tag Name	Tag ID	Payload Type	Description
NBR_VIDEO_IN_TAG	5	t_octet	number of video inputs
NBR_TRANSCODER_TAG	6	t_octet	number of transcoders
NBR_AUDIO_IN_TAG	7	t_octet	number of audio inputs
NBR_AUDIO_OUT_TAG	8	t_octet	number of audio outputs
AUDIO_OPTIONS_TAG	9	t_octet	
EPTZ_TRACKER_TAG	10	t_octet	eptz tracker supported
IMAGE_PIPE_FEATURE_TAG_OLD	11	t_dword	returns 1st 4bytes of the information that is provided by bicom command 0x0520 (ServerID 4)(for backward compatibility). Use the new tag IMAGE_PIPE_FEATURES to get all features
BEST_FACE_TAG	12	t_octet	device supports CONF_BEST_FACE
PLATFORM_TYPE_TAG	13	t_octet	
BICOM_DOME_TAG	14	t_octet	Indication if it is a dome that can be controlled using bicom (e.g via CONF_BICOM_COMMAND)
FPGA_TAG	15	t_octet	FPGA available
VOUT_TAG	16	t_octet	camera has a vout
HEATER_TAG	17	t_octet	camera has a heater
SERIAL_PORT_TAG	18	t_octet	camera has a serial port
NTCIP_TAG	19	t_octet	camera is ready for NTCIP use
REPLAY_TAG	20	t_dword	supported replay option flags
GB28181_TAG	21	t_octet	support of GB28181
FONT_SUPPORT_TAG	22	t_dword	supported utf-16 characters for Stamping; ASCII chars are always supported;
RECORDER_TAG	23	p_octet	record format version
PTZ_ON_CLIENT_TAG	24	t_octet	
VIRTUAL_LINES_TAG	25	t_octet	Returns the number of virtual lines on this device or zero if there are none. If there are virtual lines, more information is available via CONF_VIRTUAL_LINES.
DPTZ_STATIC_CAPS_TAG	26	p_octet	Signal PTZ capabilities see CONF_DPTZ_STATIC_CAPS for a detailed description of the payload
MAX_DEC_RESOLUTION_TAG	27	p_octet	maximum decoder/transcoder resolution
MAX_SUBVIEWS_TAG	28	t_octet	max number of video subviews
FIRE_DETECTION_TAG	29	t_octet	Fire detection available
FW_UPLOAD_SIGNATURE_TAG	30	t_octet	Firmware upload signature required

© Copyright 2001-2021 Bosch Security Systems

Tag Name	Tag ID	Payload Type	Description
IMAGE_PIPE_FEATURE_TAG	31	p_octet	returns the information that is provided by bicom command 0x0520 (ServerID 4)
CLUSTERED_DEVICE_TAG	32	t_octet	indication if the camera is clustered with other devices
PIR_TAG	33	t_octet	number of PIR (passive infrared) sensors
AMBIENT_LIGHT_DETECT_TAG	34	t_octet	ambient light detection supported
SUPPORT_SUDO_URL	35	t_octet	/sudo URL is available to switch users
IS_IN_FORCE_PWD_MODE	36	t_octet	Device has no password at the moment and will force the user to set one
IS_PTZ_IN_VCD	37	t_octet	PTZ information can be found in VCD stream (this capabilty is necessary for the VCA global field feature)
CODER_SPEC_ENC_PROF	38	t_octet	coder specific encoder presets
POE_BASE_CLASS_TAG	39	t_octet	base PoE class of device, without consideration of power adder values
SOD_DETECTION_TAG	40	t_octet	SOD detection is available
VIDEO_LINE_DUO	41	t_octet	Video line supports side-by-side image
ISCSI_SUPPORT	42	t_dword	iSCSI capabilities
INDIVIDUAL_ENCODER_OPERATION_MODE_CONFIG	43	t_word	
WIFI_TAG	44	t_octet	Wifi support
C_MOUNT_LENS_TYPE	45	t_octet	Device has a changeable lens C-mount
TC_ENC_H265_TAG	46	t_octet	Transcoder supports H.265
NET_DYNAMIC	47	t_octet	Device network settings can be changed dynamically without reboot
KINESIS_SUPPORT	48	t_octet	Device supports streaming to kinesis
AWS_S3_SUPPORT	49	t_octet	Device supports streaming to amazon s3
RCP_UDP_PAYLOAD_ENCRYPTION	50	t_octet	Device supports rcp udp payload encryption
IMU_TAG	51	t_octet	number of IMU sensors (Inertial Measurement Unit; acceleration, gyroscope, ..)
CLOUD_WATCH_SUPPORT	52	t_octet	device supports logging to cloud watch
CALIBRATION_LINE_DEPENDENCIES	53	p_octet	returns the information calibration dependencies between lines, if missing then all lines are independent.
ACCOUNT_TYPE_SUPPORT	54	t_dword	account types support
DYN_SCENE_CTRL_OPTIONS_TAG	55	t_octet	device supports CONF_ENC_DYN_SCENE_CTRL

© Copyright 2001-2021 Bosch Security Systems

Tag Name	Tag ID	Payload Type	Description
SUPPORT_REFERENCE_ORIENTATION	56	t_octet	device supports tamper reference orientation
IS_SAST_SYSTEM	57	t_octet	device is a S&ST camera
ENC_BASE_OPERATION_MODE_TYPE	58	t_octet	enc base operation mode type, see CONF_ENC_BASE_OPERATION_MODE_TYPE
DEVELOPER_MODE_ALLOWED	59	t_octet	developer mode is allowed
DEVELOPER_MODE_ENABLED	60	t_octet	developer mode is enabled
RETAIL_ANALYTICS_TAG	61	t_octet	Retail analytics is available
RAW_FIR_IMAGE_TAG	62	t_octet	Device provides FIR raw image via snap.fir
TRANS_DATA_OVERIP_AVAIL_TAG	63	t_octet	Transparent data over ip feature is available
UPNP_SUPPORT_TAG	64	t_octet	UPnP support
SMART_SEARCH_OPTIONS_TAG	65	t_dword	Supported forensic search options
SUPPORT_REFERENCE_IMAGE_CHECK_TAG	66	t_octet	device supports tamper reference image check
STREAM_PRIORITY_STREAMS	67	t_dword	streams that support setting priority bit
VCA_OVERLAY	68	t_dword	vca overlays supported

Tag 1: DEVICE_TYPE_TAG

Values:

encoder	1
camera	2
transcoder	3
vrm	4
decoder	5
streaming gateway	6
storage	8
other	0

Tag 2: PTZ_CAMERA_TAG

Values:

no ptz	0	
full ptz	1	
zoom only	2	no pan/tilt

© Copyright 2001-2021 Bosch Security Systems

Tag 3: ROI_CAMERA_TAG

Values:

no	0
yes	1

Tag 4: AUTOTRACKER_TAG

Values:

no	0
yes	1

Tag 9: AUDIO_OPTIONS_TAG

Values:

	Mask	Name
Bit 3	0x08	loudspeaker (see CONF_AUDIO_OPTIONS)
Bit 2	0x04	mic
Bit 1	0x02	line out
Bit 0	0x01	line in

Tag 10: EPTZ_TRACKER_TAG

Values:

no	0
yes	1

Tag 12: BEST_FACE_TAG

Values:

no	0
yes	1

Tag 13: PLATFORM_TYPE_TAG

Values:

CPP3	1
CPP4	2
CPP-ENC	3

© Copyright 2001-2021 Bosch Security Systems

CPP5	4
CPP6	5
CPP7	6
CPP7.3	7
CPP13	9
CPP14	10
OTHER	255

Tag 14: BICOM_DOME_TAG

Values:

no	0
yes	1

Tag 15: FPGA_TAG

Values:

no	0
yes	1

Tag 16: VOUT_TAG

Values:

no	0
yes	1

Tag 17: HEATER_TAG

Values:

no	0
yes	1

Tag 18: SERIAL_PORT_TAG

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 19: NTCIP_TAG

Values:

no	0	License needs to be installed
yes	1	license is installed or NTCIP support comes pre-enabled

Tag 20: REPLAY_TAG

Values:

	Mask	Name
Bit 2	0x00000004	transcoded replay
Bit 1	0x00000002	local replay
Bit 0	0x00000001	span replay

Tag 21: GB28181_TAG

Values:

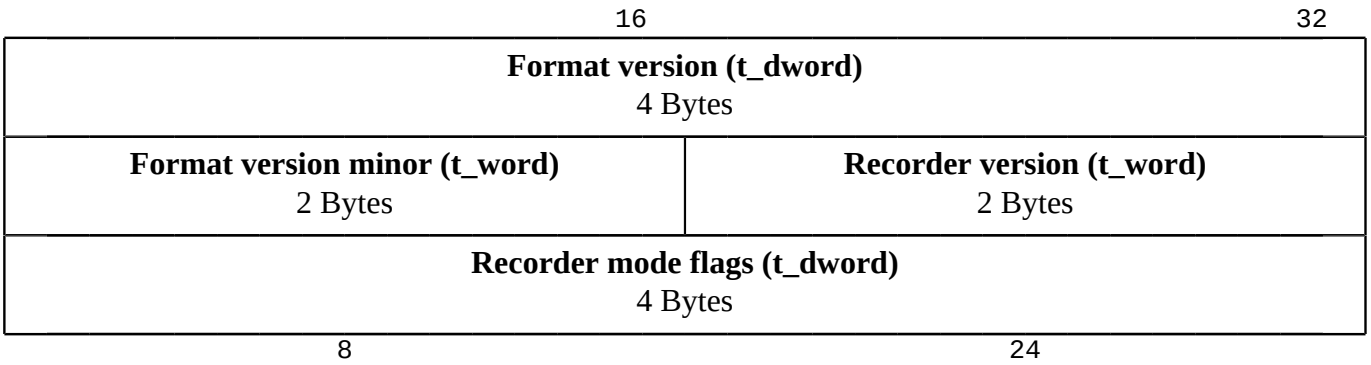
GB28181 not supported	0	
GB28181 supported	1	see CONF_GB28181 for further informations

Tag 22: FONT_SUPPORT_TAG

Values:

	Mask	Name
Bit 0	0x00000001	full BoschSansCHS-Regular.ttf support (all chinese signs included in UTF-16)

Tag 23: RECORDER_TAG



© Copyright 2001-2021 Bosch Security Systems

Recorder mode flags

Values:

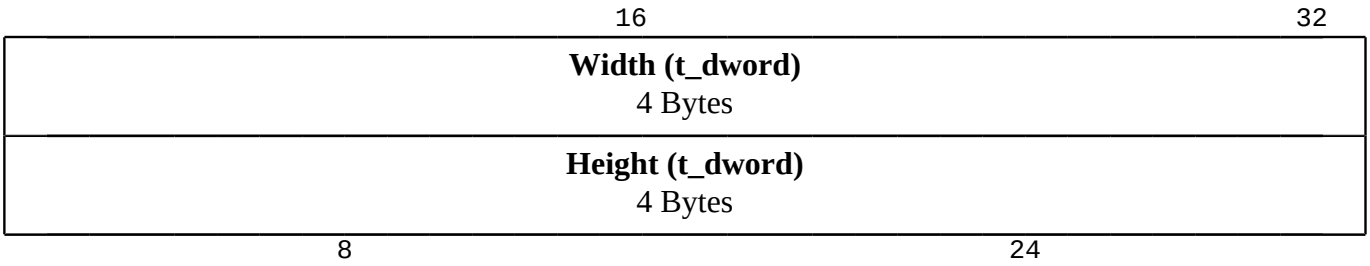
	Mask	Name
Bit 2	0x00000004	BUFFERED_RECORDING
Bit 1	0x00000002	DUAL_RECORDING
Bit 0	0x00000001	STANDARD_RECORDING

Tag 24: PTZ_ON_CLIENT_TAG

Values:

not necessary	0	
necessary	1	Dewarping on client makes sense and all required data is provided via SEI information in the h.264 stream. Normally the case on panoramic cameras with a large view angle 180ï¿½ cameras.

Tag 27: MAX_DEC_RESOLUTION_TAG



Tag 29: FIRE_DETECTION_TAG

Values:

no	0	
standard	1	standard mode
expert	2	expert mode, not VdS-certified

Tag 30: FW_UPLOAD_SIGNATURE_TAG

Values:

unsigned firmware accepted	0
only signed firmware accepted	1

© Copyright 2001-2021 Bosch Security Systems

Tag 32: CLUSTERED_DEVICE_TAG

Values:

no	0
yes	1

Tag 34: AMBIENT_LIGHT_DETECT_TAG

Values:

no	0
yes	1

Tag 35: SUPPORT_SUDO_URL

Values:

no	0
yes	1

Tag 36: IS_IN_FORCE_PWD_MODE

Values:

no	0
yes	1

Tag 37: IS_PTZ_IN_VCD

Values:

no	0
yes	1

Tag 38: CODER_SPEC_ENC_PROF

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 39: POE_BASE_CLASS_TAG

Values:

Class 0	0
Class 1	1
Class 2	2
Class 3	3
Class 4	4
no PoE	255

Tag 40: SOD_DETECTION_TAG

Values:

no	0
yes	1

Tag 41: VIDEO_LINE_DUO

Values:

	Mask	Name
Bit 1	0x02	Line2
Bit 0	0x01	Line1

Tag 42: ISCSI_SUPPORT

Values:

	Mask	Name
Bit 1	0x00000002	Support for Multipathing('NetApp specific style')
Bit 0	0x00000001	Support for NetApp E2800

Tag 43: INDIVIDUAL_ENCODER_OPERATION_MODE_CONFIG

Values:

global encoder configuration mode	0	In CONF_CODER_VIDEO_OPERATION_MODE, num parameter supports only 0 as global option for all possible encoders

© Copyright 2001-2021 Bosch Security Systems

individual encoder configuration mode	1	In CONF_CODER_VIDEO_OPERATION_MODE, num parameter for a specific absolute coder has to be provided. Global configuration (numDesc = 0) is not supported.

Tag 44: WIFI_TAG

Values:

no	0
as station	1
as access point	2

Tag 45: C_MOUNT_LENS_TYPE

Values:

Fix Attached Lens	0	Device has a fix attached lens
Changeable Lens	1	Lens is changeable

Tag 46: TC_ENC_H265_TAG

Values:

no	0
yes	1

Tag 47: NET_DYNAMIC

Values:

no	0
yes	1

Tag 48: KINESIS_SUPPORT

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 49: AWS_S3_SUPPORT

Values:

no	0
yes	1

Tag 50: RCP_UDP_PAYLOAD_ENCRYPTION

Values:

no	0
yes	1

Tag 52: CLOUD_WATCH_SUPPORT

Values:

no	0
yes	1

Tag 54: ACCOUNT_TYPE_SUPPORT

Values:

	Mask	Name
Bit 7	0x00000080	file
Bit 6	0x00000040	kinesis
Bit 5	0x00000020	aws s3
Bit 4	0x00000010	record
Bit 3	0x00000008	smb
Bit 2	0x00000004	dropbox
Bit 1	0x00000002	ftp
Bit 0	0x00000001	no

Tag 55: DYN_SCENE_CTRL_OPTIONS_TAG

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 56: SUPPORT_REFERENCE_ORIENTATION

Values:

no	0
yes	1

Tag 57: IS_SAST_SYSTEM

Values:

no	0
yes	1

Tag 58: ENC_BASE_OPERATION_MODE_TYPE

Values:

not supported	0	no base operation mode support
fixed	1	base operation modes define a specific fixed resolution (except of SD mode)
limit	2	base operation modes specify an upper limit for the resolution + frame rate

Tag 59: DEVELOPER_MODE_ALLOWED

Values:

no	0
yes	1

Tag 60: DEVELOPER_MODE_ENABLED

Values:

no	0
yes	1

Tag 61: RETAIL_ANALYTICS_TAG

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 62: RAW_FIR_IMAGE_TAG

Values:

no	0
yes	1

Tag 63: TRANS_DATA_OVERIP_AVAIL_TAG

Values:

no	0
yes	1

Tag 64: UPNP_SUPPORT_TAG

Values:

no	0
yes	1

Tag 65: SMART_SEARCH_OPTIONS_TAG

Values:

	Mask	**Name**
Bit 5	0x00000020	Text Search
Bit 4	0x00000010	Recorded Alarms
Bit 3	0x00000008	Best Face
Bit 2	0x00000004	Line Crossing
Bit 1	0x00000002	Object in Field
Bit 0	0x00000001	Any Motion

Tag 66: SUPPORT_REFERENCE_IMAGE_CHECK_TAG

Values:

no	0
yes	1

© Copyright 2001-2021 Bosch Security Systems

Tag 67: STREAM_PRIORITY_STREAMS

Values:

	Mask	Name
Bit 2	0x00000004	Stream3
Bit 1	0x00000002	Stream2
Bit 0	0x00000001	Stream1

Tag 68: VCA_OVERLAY

Values:

	Mask	Name
Bit 1	0x00000002	PrivacyMode
Bit 0	0x00000001	On

© Copyright 2001-2021 Bosch Security Systems

2.158 CONF_DEVICE_GUID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b22	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy	Read the device's GUID (32 bytes length)	
Write	p_octet	service	Write the device's GUID (32 bytes length)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.159 CONF_DEVICE_OPERATION_MODE_VERSIONS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b61	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Information about device operation_modes, see	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to get a hint for changes between firmware versions or differences between device types concerning handling or way of configuration of the devices for several aspects. In order to parse the response payload, first check the first byte of a operation mode version info structure, which is the type and depending on the type read the remaining payload of the structure. Do this again for all following operation mode version info structures until the end of the response payload.

Payload Structure

Operation Mode Version Info [0] (see description)
...
Operation Mode Version Info [N] (see description)

Operation Mode Version Info

A list of operation mode version info structures.

Type 1 Byte	**Specific Information** (see description)

Type

Type of the operation mode version info structure.

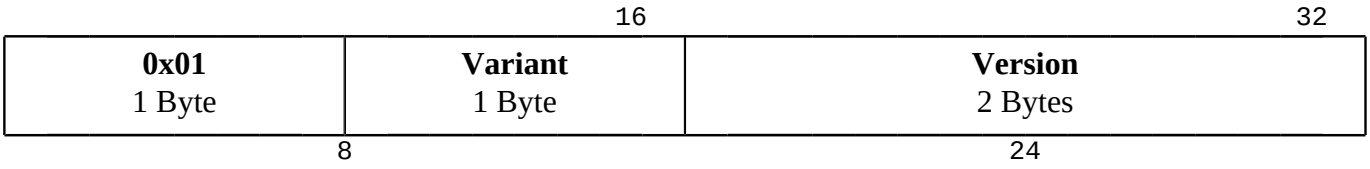
Version	1	recording configuration version

© Copyright 2001-2021 Bosch Security Systems

Specific Information

Specific information depending on the type.

'Specific Information' payload for 'Type' = 'Recording configuration version' (1)



Variant

Variant of the recording configuration.

Values:

Profile Configuration Mode	0	Configuration has to be done by command CONF_HD_RECORD_PROFILES
Stream Configuration Mode	1	Configuration has to be done by command CONF_HD_RECORD_PROFILES_V2

Version

Version of the recording configuration. Current version is 1.

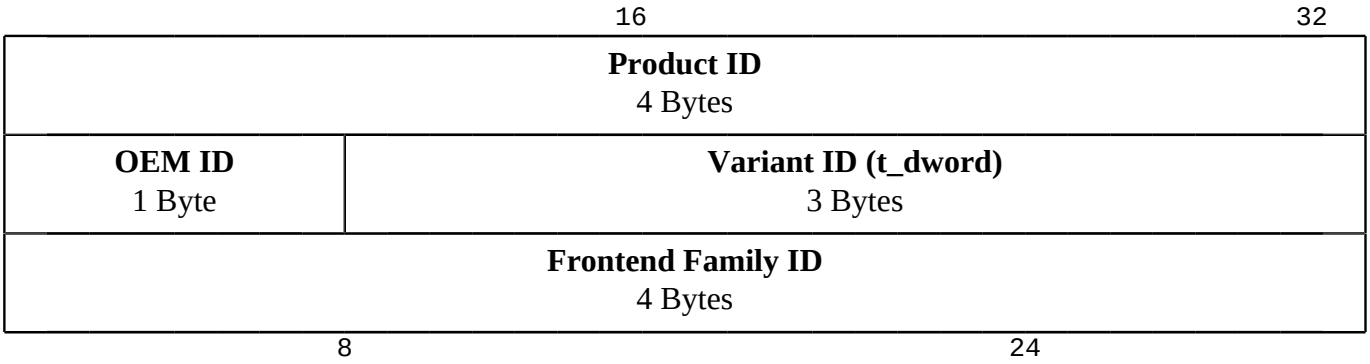
© Copyright 2001-2021 Bosch Security Systems

2.160 CONF_DEVICE_TYPE_IDS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b07	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy	Read the device type ids (hexadecimal).	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



OEM ID

BOSCH = 0

Frontend Family ID

Bicom

© Copyright 2001-2021 Bosch Security Systems

2.161 CONF_DHCP_COMPLIANCY

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ada	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read mode of DHCP standard compliancy; either it is allowed, that the last-assigned ip is used during times, where dhcp server is not available for a refresh, or this is not allowed, as demanded by rfc 2131, and device will release the assigned but expired ip and with it, enter a state of possible limited availability until requesting a new ip succeeded;	
Write	t_octet	service	Write mode of DHCP standard compliancy;	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

allow last-assigned	0
strict rfc	1

© Copyright 2001-2021 Bosch Security Systems

2.162 CONF_DHCP_OFF

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00ae	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Read, whether DHCP is in OFF mode	
Write	f_flag	service	Set DHCP mode to OFF	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.163 CONF_DHCP_ON

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00ad	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Read, whether DHCP is in ON, ON No Kick, or ON+APIPA-Fallback mode	
Write	f_flag	service	Set DHCP mode to ON	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.164 CONF_DHCP_STABLE

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ac8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read whether DHCP machine is in possession of an IP	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.165 CONF_DHCP_VAL

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00af	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read DHCP mode	
Write	t_octet	service	Set DHCP mode	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

OFF	0	
ON	1	On write: Enables DHCP and kicks lease acquisition mechanism
ON No Kick	2	Write only: enables DHCP mode in config, but does not kick lease acquisition mechanism
ON+APIPA-Fallback	3	

© Copyright 2001-2021 Bosch Security Systems

2.166 CONF_DIFF_SERV_POST_ALARM_TIME

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b3b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get post alarm time (in seconds) for DiffServ alarm values	
Write	t_word	service	Set post alarm time (in seconds) for DiffServ alarm values. (Only if DiffServ alarm values are specified and therefore DiffServ values are changed during an alarm).	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.167 CONF_DIFF_SERV_VAL

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b27	yes 1..6 Audio/Video/ Control/ Alarm-Audio/ Alarm-Video/ Alarm-Control	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the DiffServ value for media sockets	
Write	t_octet	service	Sets the DiffServ value for media sockets	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

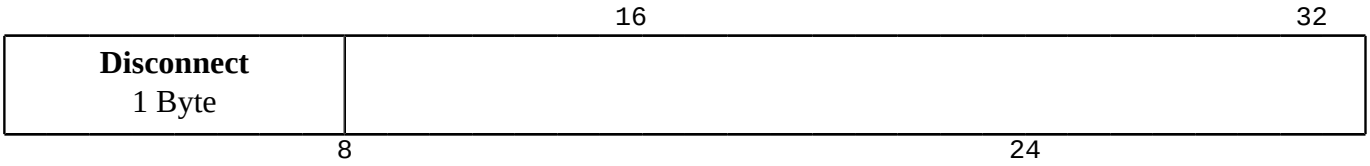
© Copyright 2001-2021 Bosch Security Systems

2.168 CONF_DISCONNECT_PRIMITIVE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff0d	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet f_flag	live		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Write Payload Structure

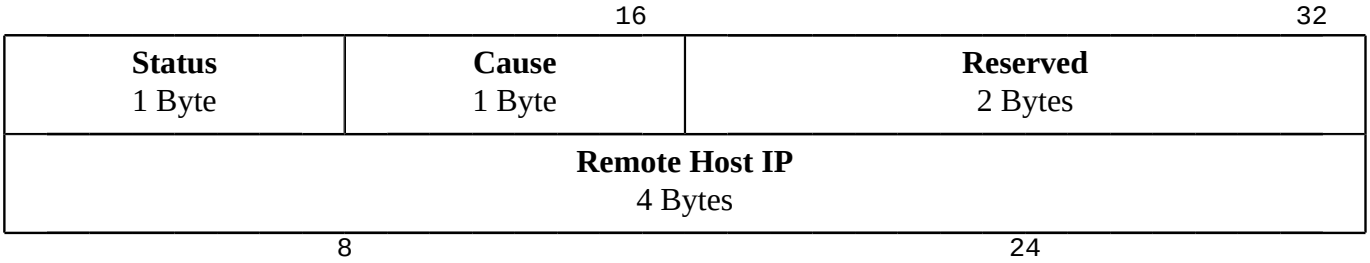


Disconnect

Values:

Disconnected	1
Keep Connection	0

Write Response Payload Structure



Status

Values:

Disconnected	1	Connection disconnected
Not Found	2	Connection identified by the given Session ID not found on this host

© Copyright 2001-2021 Bosch Security Systems

Cause

Values:

Not closed	0x00
Normal termination	0x01
Abnormal termination	0x02
No response	0x03
Remote host terminated	0x04
Timed out	0x05
Remote login rejected	0x06
No common media channels	0x07
Connection substituted	0x08
Automatic disconnect	0x09
Stop streaming	0x0a

Remote Host IP

IP address of the remote connected host.

Note: This command is not readable.

© Copyright 2001-2021 Bosch Security Systems

2.169 CONF_DISCOVER_PORT

[API: rcp.discovery]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0976	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the discover port	
Write	t_word	service	Set the discover port	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.170 CONF_DNS_SERVER_IP

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a1f	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get dns server ip/ipv6	
Write	t_dword	service	Set dns server ip/ipv6	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Primary DNS	0
Secondary DNS	2

© Copyright 2001-2021 Bosch Security Systems

2.171 CONF_DNS_SERVER_IP_STRING

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b49	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get dns server ip/ipv6	
Write	p_string	service	Set dns server ip/ipv6	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Primary DNS	0
Secondary DNS	2

© Copyright 2001-2021 Bosch Security Systems

2.172 CONF_DPTZ_DYNAMIC_CAPS

[API: roi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb7	0 - addressing via Session ID	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read current/dynamic DPTZ capabilities. 1st bit: ROI, 2nd bit: E-PTZ-Tracker	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.173 CONF_DPTZ_STATIC_CAPS

[API: roi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bfd	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read current DPTZ capabilities. They may are changed if CONF_VIDEO_H264_ENC_BASE_OPERATION_MODE or CONF_VIDEO_LINE_DEWARPING_MODE is changed. The command consists of n entries with a fixed length of 8 Bytes and the following data.

Payload Structure

	16		32
Line 1 Byte	**Stream** 1 Byte	**Bicom PTZ Steering** 1 Byte	**Auto Tracker Type** 1 Byte
Steering Type 1 Byte	**Reserved** 3 Bytes		
8	24		

Line

Line id (1 based)

Stream

Stream id (1 based)

Bicom PTZ Steering

Not Supported	0	PTZ commands are not supported
Supported	1	PTZ commands are supported

Auto Tracker Type

Not Supported	0
EPTZ supported	1

© Copyright 2001-2021 Bosch Security Systems

Steering Type

ptzSteeringTypeNoOptionPresent	0
ptzSteeringTypePhysicalDome	1
ptzSteeringTypeCutOutRegion	2
ptzSteeringTypeVirtualPtz	3

© Copyright 2001-2021 Bosch Security Systems

2.174 CONF_DROPBOX_AUTH_ADDR

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b81	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet p_string	minimal	Gets the url that should be used to login at dropbox	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.175 CONF_DROPBOX_AUTH_STATUS

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b82	account number	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets authentication status #ValueList: 0=not authenticated; 1=authenticated; 2=authentication session timed out; 3=failed;	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.176 CONF_DROPBOX_AUTH_V2

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c7b	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	The dropbox authorization code and start the process to get the dropbox access token	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.177 CONF_DROPBOX_TOKEN_V2

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c7a	account number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Gets the dropbox token (v2)	
Write	p_octet	service	Sets the dropbox token (v2), max 256 bytes	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.178 CONF_DYNAMIC_HTML_COUNT

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cef	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read number of available dynamic html pages	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.179 CONF_DYNAMIC_HTML_DATA

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0299	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the data(content) for a dynamic HTML page (num range 1..DYNAMIC_HTML_COUNT) (max. 2048 char)	
Write	p_octet	service	Set the data(content) for a dynamic HTML page (num range 1..4) (max. 2048 char)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.180 CONF_DYNAMIC_HTML_NAME

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0298	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the name for a dynamic HTML page(num range 1..DYNAMIC_HTML_COUNT)	
Write	p_string	service	Set the name for a dynamic HTML page (num range 1..4) (max. 19 char)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.181 CONF_DYNDNS_ENABLE

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a59	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Registering at dyndns.com,	
Write	t_octet	service	Registering at dyndns.com,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

enable	1
disable	0

© Copyright 2001-2021 Bosch Security Systems

2.182 CONF_DYNDNS_FORCE_REGISTER_NOW

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a5c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Writing forces registering at dyndns.com / reading checks if forcing is allowed (has not been done since bootup)	
Write	t_octet	service	Writing forces registering at dyndns.com / reading checks if forcing is allowed (has not been done since bootup)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.183 CONF_DYNDNS_HOST_NAME

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a56	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the dyndns host name that is to be registered at dyndns.com	
Write	p_string	service	Write the dyndns host name that is to be registered at dyndns.com	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.184 CONF_DYNDNS_LAST_REGISTER

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a5b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Data of last successful registration at dyndns.com, 8 bytes, first DWORD: register time in secs since 2000, second DWORD: last registered IP	
Write	p_octet	service	Data of last successful registration at dyndns.com, 8 bytes, first DWORD: register time in secs since 2000, second DWORD: last registered IP	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.185 CONF_DYNDNS_MAIL_DEST_EMAIL_ADDR

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b75	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read destination email address to be used for for Dyndns status mail transmission	
Write	p_string	service	Set destination email address to be used for for Dyndns status mail transmission	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.186 CONF_DYNDNS_MAIL_SENDER_NAME

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b76	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read senders name to be used for for Dyndns status mail transmission	
Write	p_string	service	Set senders name to be used for for Dyndns status mail transmission	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.187 CONF_DYNDNS_MAIL_SMTP_LOGIN

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b73	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read login for SMTP server to be used for for Dyndns status mail transmission	
Write	p_string	service	Set login for SMTP server to be used for for Dyndns status mail transmission	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.188 CONF_DYNDNS_MAIL_SMTP_PASS

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b74	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read password for SMTP server to be used for for Dyndns status mail transmission	
Write	p_string	service	Set password for SMTP server to be used for for Dyndns status mail transmission	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.189 CONF_DYNDNS_MAIL_SMTP_SRV_IP_STR

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b72	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read address of SMTP server to be used for for Dyndns status mail transmission	
Write	p_string	service	Set address of SMTP server to be used for for Dyndns status mail transmission	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.190 CONF_DYNDNS_MAIL_TEST_SEND

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b77	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	Send test mail with current Dyndns status to configured destination	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.191 CONF_DYNDNS_PASSWORD

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a58	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read the password of account at dyndns.com	
Write	p_string	service	Write the password of account at dyndns.com	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.192 CONF_DYNDNS_PROVIDER

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b67	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read selected dyndns provider,	
Write	t_octet	service	Write selected dyndns provider,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

dyndns.org	0
no-ip.com	1
selfhost.de	2

© Copyright 2001-2021 Bosch Security Systems

2.193 CONF_DYNDNS_STATE

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a5a	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	State of registering at dyndns.com, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

State

The current state of the process of registering the device at dyndns.com

Values:

Success	0	successfully updated
ConfigUnchanged	1	updated with unchanged cfg
InvalidCreentials	2	dyndns username or password wrong
UnqualifiedDomain	3	not fully qualified domain name (e.g. not in form host.dyndns.org)
InvalidHostname	4	wrong host name for this dyndns user account
TooManyHosts	5	too many hosts in last update
HostnameBlocked	6	host name is blocked by dyndns for update abuse
NoAgent	7	no user agent submitted or http method not permitted
ErrorDns	8	dyndns server error dns related
ErrorMaintenance	9	dyndns server error maintenance related
ErrorMisconfigured	252	update not done due to misconfiguration
ErrorFatal	253	fatal error during update process
UpdateInProgress	254	update in process
SwitchedOff	255	switched off

© Copyright 2001-2021 Bosch Security Systems

2.194 CONF_DYNDNS_STATUS_MAIL_ONOFF

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b71	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read enable or disable state for sending status update emails on Dyndns status change,	
Write	t_octet	service	Enable or disable sending status update emails on Dyndns status change,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

off	0	default
on	1	

© Copyright 2001-2021 Bosch Security Systems

2.195 CONF_DYNDNS_USER_NAME

[API: dyndns]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a57	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the user name of account at dyndns.com	
Write	p_string	service	Write the dyndns host name that is to be registered at dyndns.com	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.196 CONF_EAP_AUTO_ADJUST_INVALID_SYSTEMTIME

[API: eap]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d1f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Enable/disable if the device should auto-adjust a potentially invalid system time to a proper time for EAP/802.1x certificate based authentication. To ensure it can connect to the network even after it lost its time e.g. after a power-outage	
Write	t_dword	service	Enable/disable if the device should auto-adjust a potentially invalid system time to a proper time for EAP/802.1x certificate based authentication. To ensure it can connect to the network even after it lost its time e.g. after a power-outage	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

off	0	Do not auto-adjust system time for EAP/802.1x
on	1	auto-adjust a potentially invalid system time for EAP/802.1x certifciate based authentication

© Copyright 2001-2021 Bosch Security Systems

2.197 CONF_EAP_ENABLE

[API: eap]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09eb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read the EAP/802.1x status	
Write	t_octet	service	Write the EAP/802.1x status	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

off	0	Don't do EAP/802.1x
on	1	Use EAP/802.1x

© Copyright 2001-2021 Bosch Security Systems

2.198 CONF_EAP_GET_IDENTITY_LIST

[API: eap]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c4d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command returns the identities used for EAP authentication. If a EAP client certificate is assigned, the command returns up to 3 identity strings. All strings are UTF-16 coded. The order of the list is the order the EAP client uses the identies. If one identity fails to authenticate the next one from the list is used.

- The alternative subject name from the certificate (tag 0x0001)
- The common name from the certificate subject (tag 0x0002)
- The EAP identity from config (tag 0x0003)

Each of these entries is optional and can be missing. The complete identity list is surrounded by the tag 0x0000. If no identity is configured or found, the tag 0x0004 is returned to indicate the empty list.

Payload Structure

List Entry [0]
...
List Entry [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

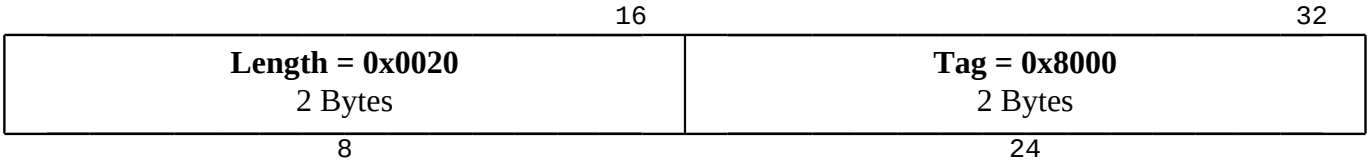
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

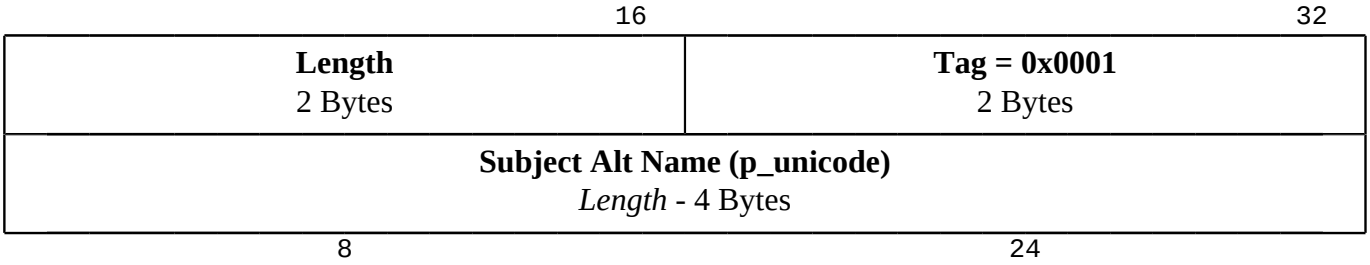
Tag 0x8000: List Entry

The payload of tag list entry contains further tagged values described below. For each identity list one list entry tag is contained in the command payload. The highest bit (bit 15) is set to indicate that this tag will contain subtags.



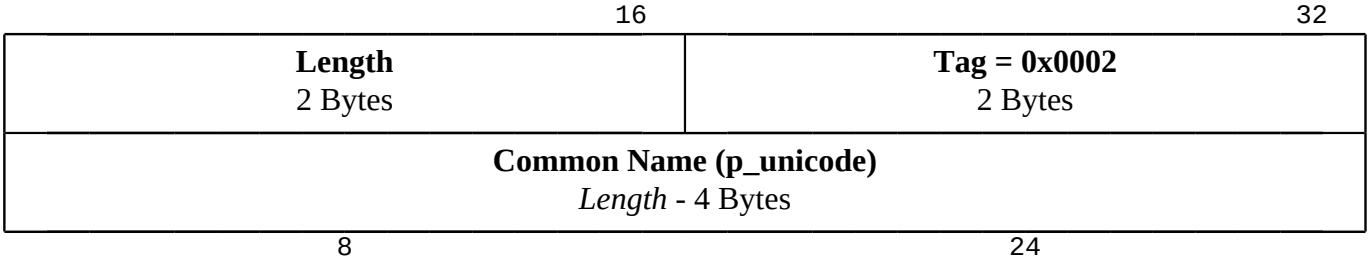
Tag 1: Subject Alt Name

Contains the Alternative Subject Name from the EAP client certificate coded as UTF-16, no zero termination.



Tag 2: Common Name

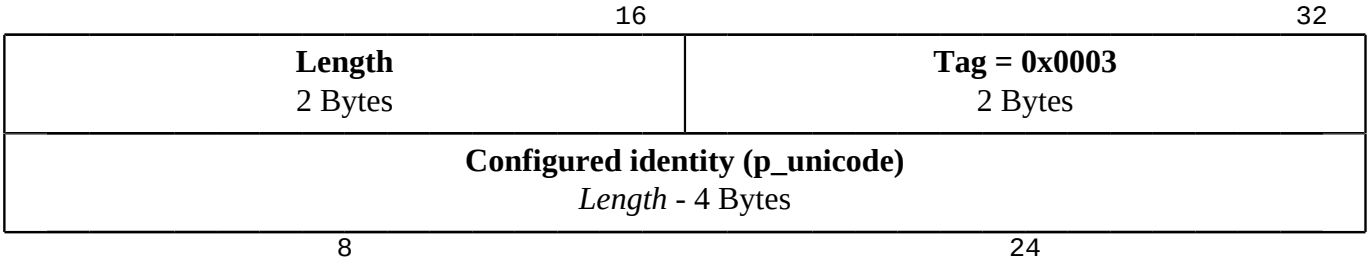
Contains the subject's Common Name from the EAP client certificate coded as UTF-16, no zero termination.



© Copyright 2001-2021 Bosch Security Systems

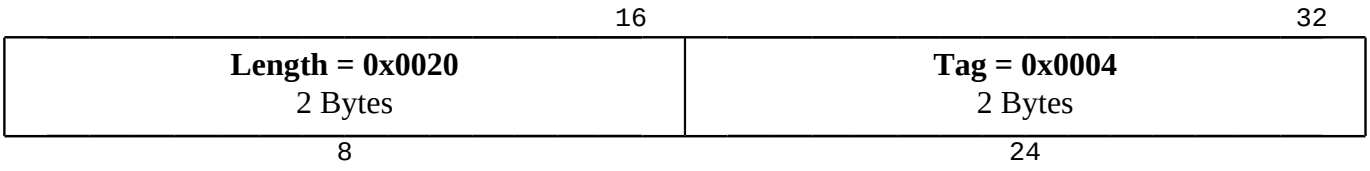
Tag 3: Configured identity

Contains the identity from the device config (given by the user) coded as UTF-16, no zero termination. Tag may appear several times.



Tag 4: Empty List

No identity strings were found.



© Copyright 2001-2021 Bosch Security Systems

2.199 CONF_EAP_IDENTITY

[API: eap]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ea	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the EAP identity	
Write	p_string	service	Write the EAP identity	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.200 CONF_EAP_PASSWORD

[API: eap]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ec	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the EAP password	
Write	p_string	service	Write the EAP password	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.201 CONF_ENABLE_DEVELOPER_MODE

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cfe	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Returns the information if the S&ST developer mode is enabled	
Write	f_flag	service	Enables the S&ST developer mode. To disable developer mode, perform a factory reset (To enable the developer mode a special license is needed)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.202 CONF_ENABLE_MASTERPWD

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Shows the password reset mechanism enable status	
Write	f_flag	service	Sets the password reset mechanism enable status, disable it, if you don't want Bosch support to be able to reset the passwords of your camera.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

disable	0
enable	1

© Copyright 2001-2021 Bosch Security Systems

2.203 CONF_ENABLE_ONVIF

[API: onvif]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bbd	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	#ValueList: 0=Onvif disabled; 1=Onvif enabled (default value)	
Write	f_flag	service	#ValueList: 0=Onvif disabled; 1=Onvif enabled (default value)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.204 CONF_ENABLE_TRAFFIC_LED

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09a6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Network activity LED (Dinion, NBC-255 camera): 0=off; 1=on; 2=off 10 minutes after startup	
Write	t_octet	service	Enables or disables the network activity LED (Dinion, NBC-255 camera)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.205 CONF_ENABLE_UPNP

[API: upnp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ade	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	#ValueList: 0= Upnp disabled; 1= Upnp enabled	
Write	f_flag	service	#ValueList: 0= Upnp disabled; 1= Upnp enabled	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.206 CONF_ENABLE_VCA

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0813	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	This command is used to configure the dome. Note that we write the information in the second bit of ptz_controller_available.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.207 CONF_ENC_BASE_OPERATION_MODE_TYPE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c95	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the type of base operation mode which is used on the device; 0: no base operation mode support, 1: base operation modes define a specific fixed resolution (except of SD mode),2: base operation modes specify an upper limit for the resolution + frame rate;	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.208 CONF_ENC_CURRENT_RESOLUTION

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b4b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the current resolution (width + height) in pixel of an encoder (relative to line). First two bytes of payload (line+coder) need to be specified as request payload. 1st Byte: video line (needs to be set when reading); 2nd Byte: encoder (needs to be set when reading); 3. +4. Byte: width; 5.+6. Byte: height; 7.+8. Byte reserved	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.209 CONF_ENC_DYN_SCENE_CTRL

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb6	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] 0: disable / 1: enable; Byte[1]: 1: Stream 1 leads, 2: Stream 2 leads, 3: Stream 3 leads ... ; default 255 smart stream selection Byte[2-15] reserved; support can be queried via CONF_ENC_DYN_SCENE_CTRL_OPTIONS	
Write	p_octet	service	Byte[0] 0: disable / 1: enable; Byte[1]: 0: Stream 0 leads, 1: Stream 1 leads, 2: Stream 2 leads ... ; default 255 smart stream selection Byte[2-15] reserved; support can be queried via CONF_ENC_DYN_SCENE_CTRL_OPTIONS	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.210 CONF_ENC_DYN_SCENE_CTRL_OPTIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] 1: CONF_ENC_DYN_SCENE_CTRL is supported 0 not supported; Byte[1] 0: one imagepipe per line 1: separated imagepipe per stream; stream count Byte[2-15] reserved	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.211 CONF_ENC_PROFILE_BASIC_PARAMS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b2e	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get basic parameters of an encoder profile preset. 1st DWORD: Resolution (parameter values see CONF_MPEG4_RESOLUTION); 2nd DWORD: Skip Ratio (values see CONF_MPEG4_FRAME_SKIP_RATIO); 3rd DWORD: Target bit rate (values see CONF_MPEG4_BANDWIDTH_KBPS); 4th DWORD: Maximum bit rate (values see CONF_MPEG4_BANDWIDTH_KBPS_SOFT_	
Write	p_octet	service	Set basic parameters of an encoder profile preset and apply them instantly. 1st DWORD: Resolution (parameter values see CONF_MPEG4_RESOLUTION); 2nd DWORD: Skip Ratio (values see CONF_MPEG4_FRAME_SKIP_RATIO); 3rd DWORD: Target bit rate (values see CONF_MPEG4_BANDWIDTH_KBPS); 4th DWORD: Maximum bit rate (values see CONF_MPEG4_BANDWIDTH_KBPS_SOFT_	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.212 CONF_ENC_PROFILE_PARAMS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb9	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the parameters of an encoder profile preset. Reply is in tagged format, see detailed description	
Write	p_octet	service	Set one or more parameters of an encoder profile preset. Payload is in tagged format, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The format has a tagged structure. Each tagged entry consists of a length field, a tag ID and a payload. Allows to read/write all relevant parameters of an encoder profile preset like resolution, skip, bandwidth, etc... within one single command. In read direction all available tags are returned. In write direction just the parameters/tags which should be changed need to be sent. So either one tag, several tags or all available tags can be provided.

Each tag ID in this command represents the according RCP command which already exists to read/write one of the encoder profile parameters individually. I.e. this command combines several individual RCP commands within one command and provides the parameters as separate tags. The payload of the tag entries corresponds to the related RCP command. For detailed information please check the documentation of the according RCP command.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

© Copyright 2001-2021 Bosch Security Systems

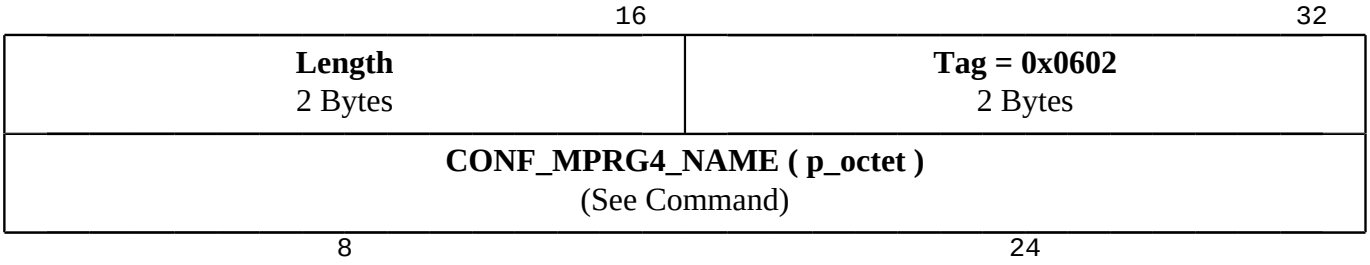
Length

Length of tagged value including length and tag field

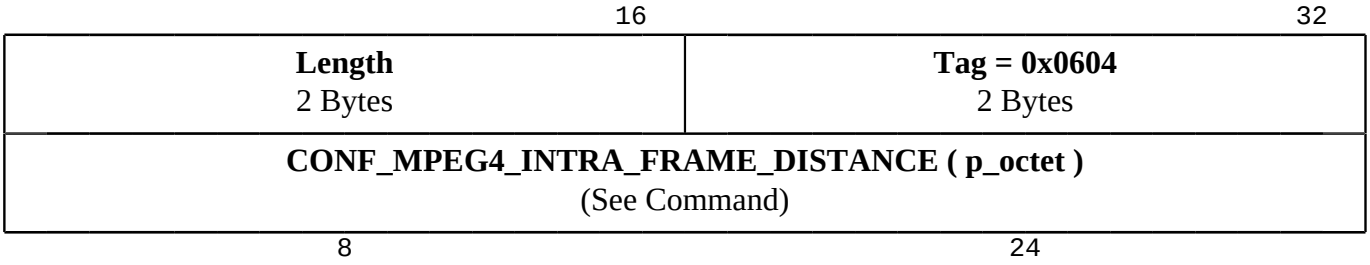
Tag

Tag specifying the encoding and meaning of the value

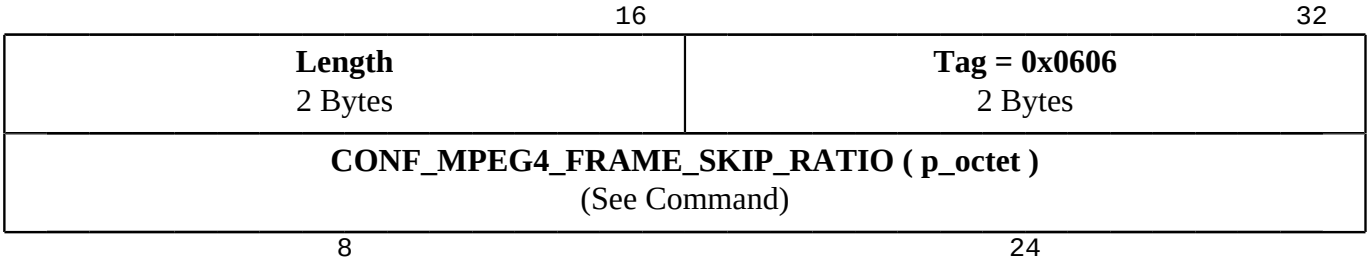
Tag 0x0602: CONF_MPRG4_NAME



Tag 0x0604: CONF_MPEG4_INTRA_FRAME_DISTANCE

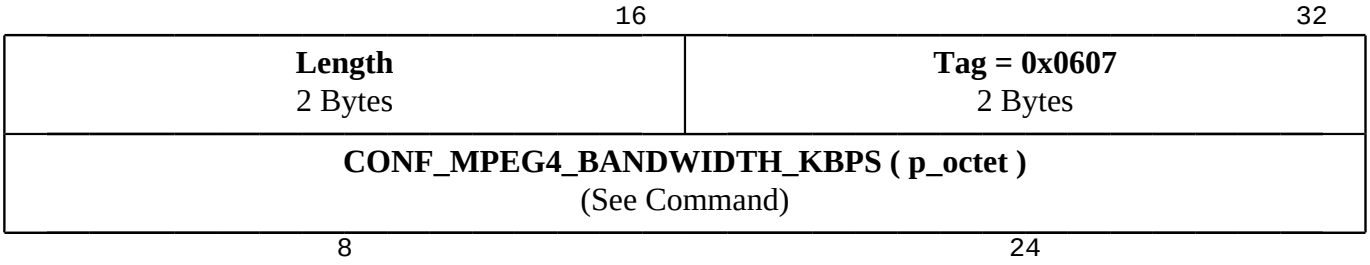


Tag 0x0606: CONF_MPEG4_FRAME_SKIP_RATIO

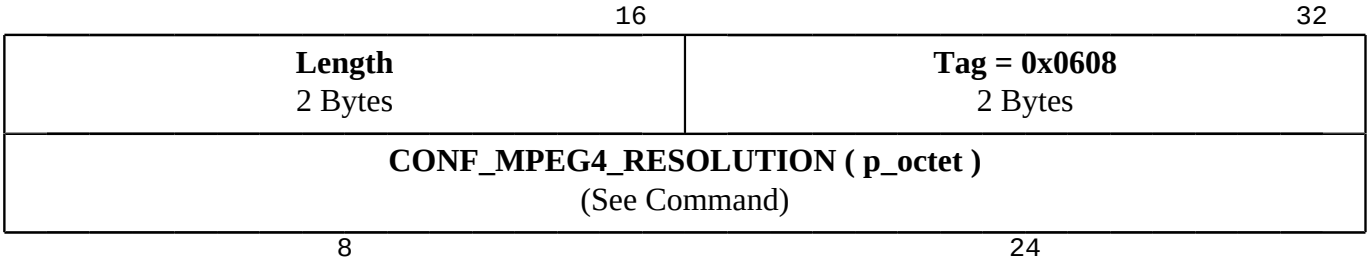


© Copyright 2001-2021 Bosch Security Systems

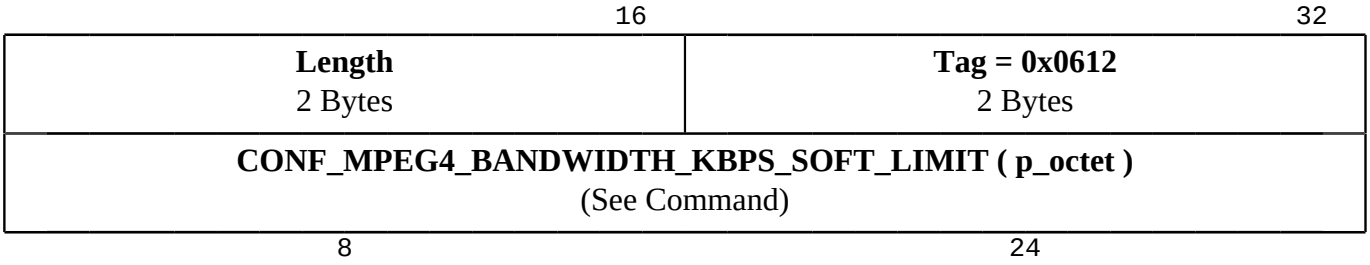
Tag 0x0607: CONF_MPEG4_BANDWIDTH_KBPS



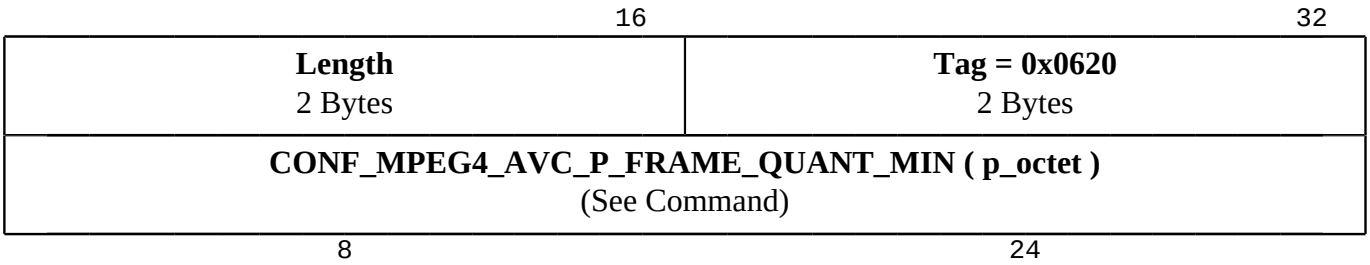
Tag 0x0608: CONF_MPEG4_RESOLUTION



Tag 0x0612: CONF_MPEG4_BANDWIDTH_KBPS_SOFT_LIMIT

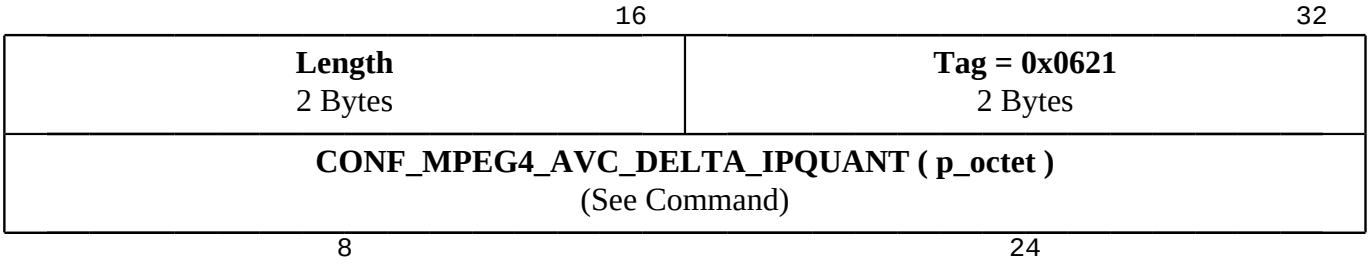


Tag 0x0620: CONF_MPEG4_AVC_P_FRAME_QUANT_MIN

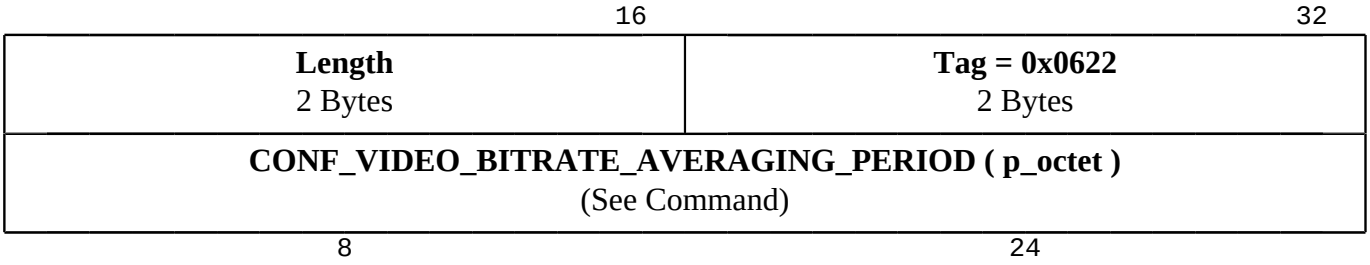


© Copyright 2001-2021 Bosch Security Systems

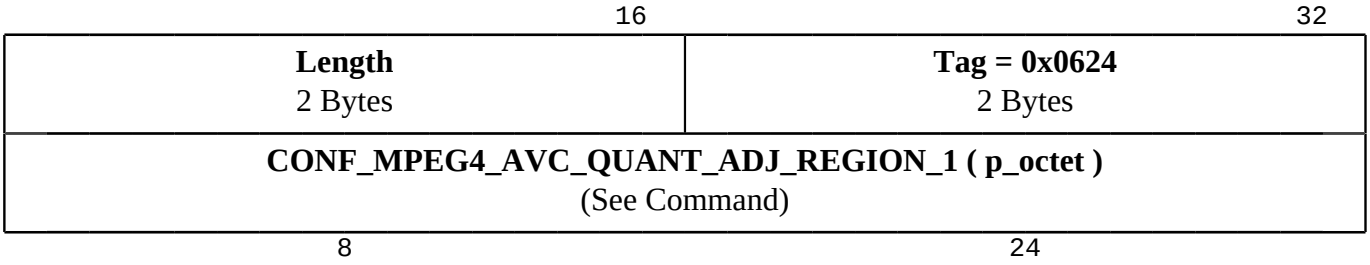
Tag 0x0621: CONF_MPEG4_AVC_DELTA_IPQUANT



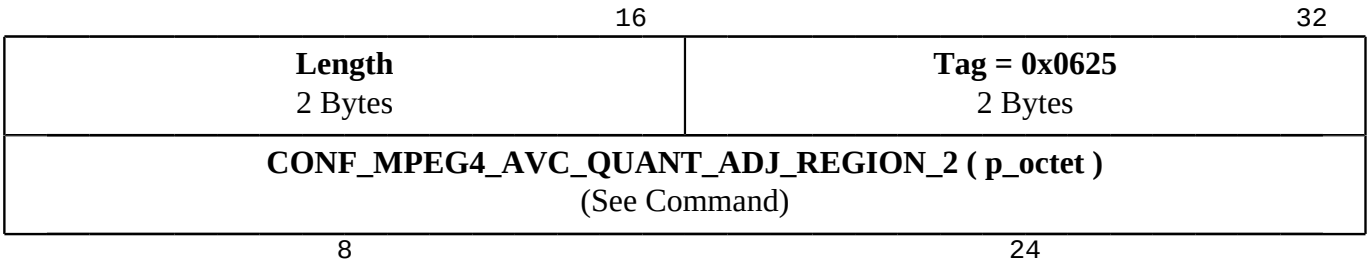
Tag 0x0622: CONF_VIDEO_BITRATE_AVERAGING_PERIOD



Tag 0x0624: CONF_MPEG4_AVC_QUANT_ADJ_REGION_1



Tag 0x0625: CONF_MPEG4_AVC_QUANT_ADJ_REGION_2



© Copyright 2001-2021 Bosch Security Systems

Tag 0x0627: CONF_VIDEO_ENC_P_REF_LIST_SIZE

16	32
Length 2 Bytes	**Tag = 0x0627** 2 Bytes
CONF_VIDEO_ENC_P_REF_LIST_SIZE (p_octet) (See Command)	
8	24

Tag 0x0a94: CONF_MPEG4_AVC_GOP_STRUCTURE

16	32
Length 2 Bytes	**Tag = 0x0a94** 2 Bytes
CONF_MPEG4_AVC_GOP_STRUCTURE (p_octet) (See Command)	
8	24

Tag 0x0c37: CONF_MPEG4_AVC_BITRATE_OPTIMIZATION

16	32
Length 2 Bytes	**Tag = 0x0c37** 2 Bytes
CONF_MPEG4_AVC_BITRATE_OPTIMIZATION (p_octet) (See Command)	
8	24

© Copyright 2001-2021 Bosch Security Systems

2.213 CONF_ENC_PROFILE_RESOLUTION_OPTIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c99	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Obtain all allowed CONF_MPEG4_RESOLUTION IDs together with their according width and height.

The format has a tagged structure. Each tagged command consits of a length field a tag id + payload. Only read is supported. Allows to query information about the resolution IDs which can be used in an encoder profile, set via CONF_MPEG4_RESOLUTION.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: Profile resolution options - for non-encoder-specific profiles

To check if the device supports encoder-specific profiles see CONF_CODER_SPECIFIC_ENC_PROFILES; For non-encoder specific profiles the resolutions are provided per line

16	32
Length 2 Bytes	**Tag = 0x0001** 2 Bytes
Profile resolution options - for non-encoder-specific profiles (p_octet) *Length* - 4 Bytes	

8 24

Profile resolution options - for non-encoder-specific profiles

16	32
id 4 Bytes	
width 4 Bytes	
height 4 Bytes	
reserved 4 Bytes	

8 24

id

Id supported by the CONF_MPEG4_RESOLUTION command

width

According width, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

height

According height, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

© Copyright 2001-2021 Bosch Security Systems

Tag 2: Profile resolution options - for encoder-specific profiles

To check if the device supports encoder-specific profiles see CONF_CODER_SPECIFIC_ENC_PROFILES; For encoder specific profiles the resolutions are provided per stream

16	32
Length 2 Bytes	**Tag = 0x0002** 2 Bytes
Profile resolution options - for encoder-specific profiles (p_octet) *Length* - 4 Bytes	
8	24

Profile resolution options - for encoder-specific profiles

16		32
stream 1 Byte	**abs coder** 1 Byte	**reserved** 2 Bytes
id 4 Bytes		
width 4 Bytes		
height 4 Bytes		
reserved 4 Bytes		
8	24	

stream

Relative stream number for which the resolution is supported; starting with '1'

abs coder

The according absolute coder number for 'stream'

id

Id supported by the CONF_MPEG4_RESOLUTION command

width

According width, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

© Copyright 2001-2021 Bosch Security Systems

height

According height, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

© Copyright 2001-2021 Bosch Security Systems

2.214 CONF_ENC_STAMPING_PROPERTIES

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16			32
UTF16CharsPerLine 2 Bytes		**CamNameLines** 2 Bytes	
AlarmStampingLines 1 Byte	**TimeStampSupport** 1 Byte	**TimeStampRes** 1 Byte	**TranspStampSupport** 1 Byte
BannerModeSupport 1 Byte	**ExtendedSysconAreas** 1 Byte	**SpinnerStampSupport** 1 Byte	**BigFontSupport** 1 Byte
SeperateLogoAreaSupport 1 Byte	**InfoStampSupport** 1 Byte	**maxLogoSize** 1 Byte	**CustomStampingLines** 1 Byte
ColorSupport 1 Byte	**customFontSupport** 1 Byte	**reserved** 2 Bytes	
8		24	

UTF16CharsPerLine

Number of UTF16 chars per line

CamNameLines

Number of stamping lines available at CONF_CAMNAME_LINES

AlarmStampingLines

Number of alarm stamping lines

TimeStampSupport

TimestampingUnsupported	0	Time stamping is not supported
TimestampingSupported	1	Time stamping is supported

© Copyright 2001-2021 Bosch Security Systems

TimeStampRes

Time stamp resolution, 1 also ms stamping is supported

TranspStampSupport

TransparentUnsupported	0	Transparent stamping is not supported
TransparentSupported	1	Transparent stamping is supported

BannerModeSupport

BannerModeUnsupported	0x00	Transparent stamping is not supported
BannerModeNameSupported	0x01	Name stamping supports banner mode
BannerModeAllSupported	0xFF	All stamping areas supports banner mode

ExtendedSysconAreas

Number of extended syscon areas. In CPP7.3 this is usually 3, in earlier platforms 0.

SpinnerStampSupport

SpinnerUnsupported	0	Spinner stamping is not supported
SpinnerSupported	1	Spinner stamping is supported

BigFontSupport

BigFontUnsupported	0	Big font is not supported
BigFontSupported	1	Big font is supported

SeperateLogoAreaSupport

TransparentUnsupported	0	Seperate logo area is not supported
TransparentSupported	1	Seperate logo area is supported

maxLogoSize

Maximum logo size

MaxLogo128	1	128x128 pixels
MaxLogo300	2	300x300 pixels

CustomStampingLines

Number of custom stamping lines

© Copyright 2001-2021 Bosch Security Systems

ColorSupport

ColorUnsupported	0	Changing the stamping color is not supported
ColorSupported	1	Changing the stamping color is supported

customFontSupport

customFontUnsupported	0	custom font is not supported
customFontSupported	1	custom font is supported

© Copyright 2001-2021 Bosch Security Systems

2.215 CONF_ETH_LINK

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092d	yes; num=0 is the mode of the external state on single ethernet port units or the internal link mode on multiple ethernet port units; num>=1 is the mode of the corresponding port on multiple ethenet port units (SFP Fiber port is always the last counted port). On module based units, only the master module is capable of setting the external port modes.	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	(half duplex=HD, full duplex =FD) 0=auto, 1=10MbitHD, 2=10MbitFD, 3=100MbitHD, 4=100MbitFD	
Write	t_octet	service	(half duplex=HD, full duplex =FD) 0=auto, 1=10MbitHD, 2=10MbitFD, 3=100MbitHD, 4=100MbitFD, 5=1000MbitFD, 8=10GbitFD	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.216 CONF_ETH_LINK_STATUS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a24	yes; num=0 returns the status of the external state on single ethernet port units or the internal link state on multiple ethernet port units; num>=1 returns the state of the corresponding port on multiple ethenet port units (SFP Fiber port is always the last counted port). On module based units, only the master module is capable of returning the external states.	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	(half duplex=HD, full duplex =FD) 0=No link, 1=10MbitHD, 2=10MbitFD, 3=100MbitHD, 4=100MbitFD, 5=1000MbitFD, 7=Wlan; 8=10GbitFD	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.217 CONF_ETH_LINK_TROUGHPUT

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a80	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Lower 16 bit: downlink KBPS, upper 16 bit: uplink KBPS	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.218 CONF_ETH_TX_PKT_BURST

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0afe	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Eth tx pkts per burst (0:=no limitation)	
Write	t_dword	service	Eth tx pkts per burst (0:=no limitation)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.219 CONF_EVENT_SCRIPT

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09f3	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live	Get the compressed Alarm Task Script, Msg: yes 8 byte for error header; byte 1-4 for type of message: 0 - successful, 1 - error, 2 - warning	
Write	p_octet	service	Write a new Alarm Task Script; note: script must be compressed (zipped)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.220 CONF_EXPORT_SPAN

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a2e	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Retrieve a span of the devices storage.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

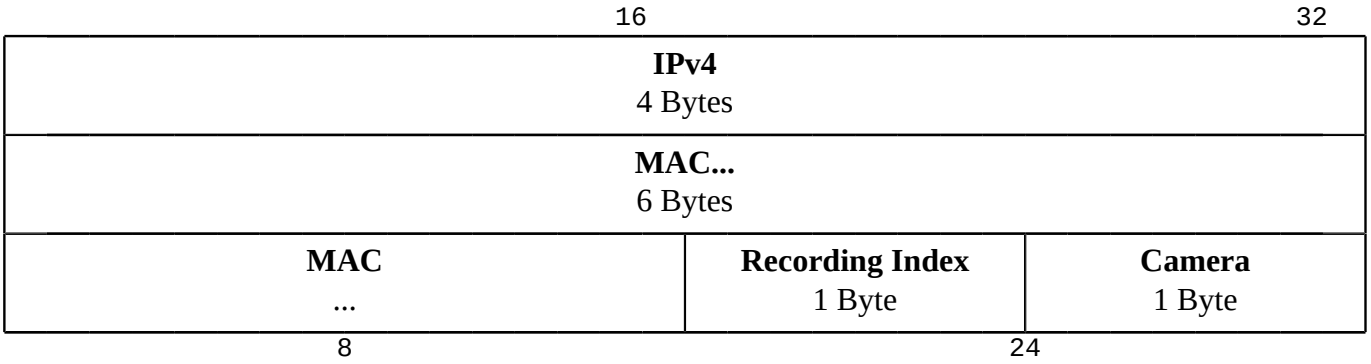
Request Payload Structure

16	32
Recording Camera... (see description)	
Recording Camera ...	
Recording Camera ...	
Lun Addr... (see description)	
Lun Addr ...	
Overwrite Retention Time max. 128 bytes	
Reserved 4 Bytes	
Url... 8 Bytes	
Url ...	
8	24

Recording Camera

The address of the device the span is assigned to.

© Copyright 2001-2021 Bosch Security Systems



IPv4

The ipv4 address of the recording device. If the device hasn't an ipv4 Adress, this field should be set to invalid (zero ip 0.0.0.0). In that case a valid url has to be provided.

MAC

The hardware address of the recording device.

Recording Index

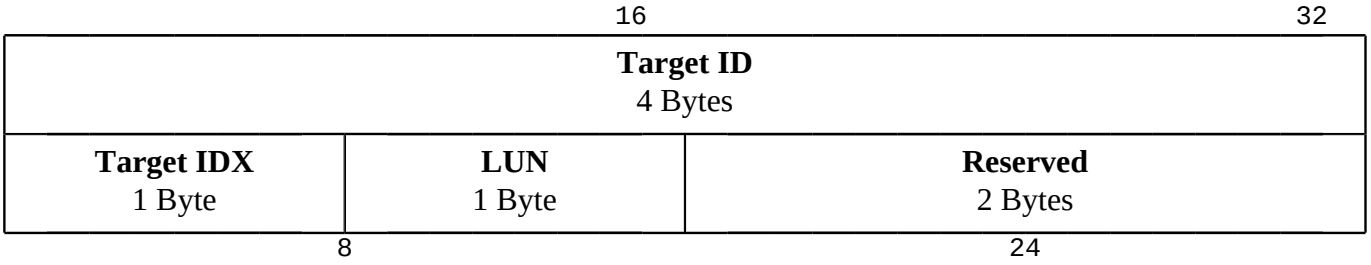
Primary Recording	1
Secondary Recording	2

Camera

The camera index starting from 1.

Lun Addr

The address of the lun a span is desired from.



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

© Copyright 2001-2021 Bosch Security Systems

Target IDX

The index of the target.

LUN

The logical unit number.

Overwrite Retention Time

If this field is set (> 0) and no free spans are available, spans with retention times that are lower or equal to the specified retention time are accounted for export. In this case the recording with the retention time that expires next is overwritten.

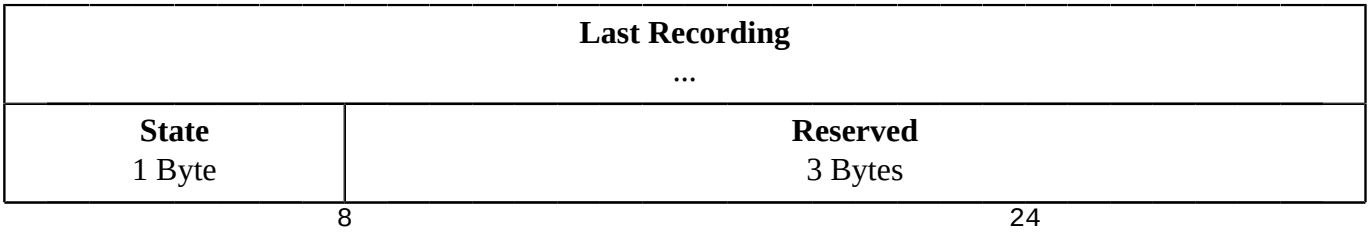
Url

Url of the the device which requests the span as zero terminated ascii string. Max length is 128 bytes including zero termination. This parameter is optional but is requiered if the device doesn't has an ipv4 address or if the IPv4 adress of the Cam id is not used.

Response/Message Payload Structure

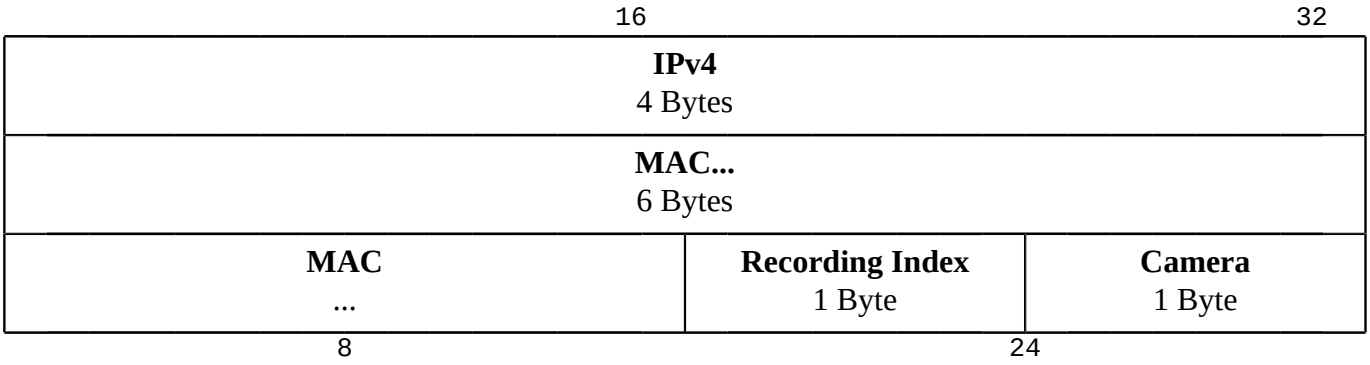
16	32
Recording Camera/Storage Managing Host... 12 Bytes	
Recording Camera/Storage Managing Host ...	
Recording Camera/Storage Managing Host ...	
Span Address... (see description)	
Span Address ...	
Last Recording... (see description)	
Last Recording ...	
Last Recording ...	

© Copyright 2001-2021 Bosch Security Systems



Recording Camera/Storage Managing Host

The address of the device the span is assigned to, only in message payload. In case of message payload only the MAC address will be delivered and all other field will be zero.



IPv4

The ipv4 address of the recording device. If the device hasn't an ipv4 Adress, this field should be set to invalid (zero ip 0.0.0.0). In that case a valid url has to be provided.

MAC

The hardware address of the recording device.

Recording Index

Primary Recording	1
Secondary Recording	2

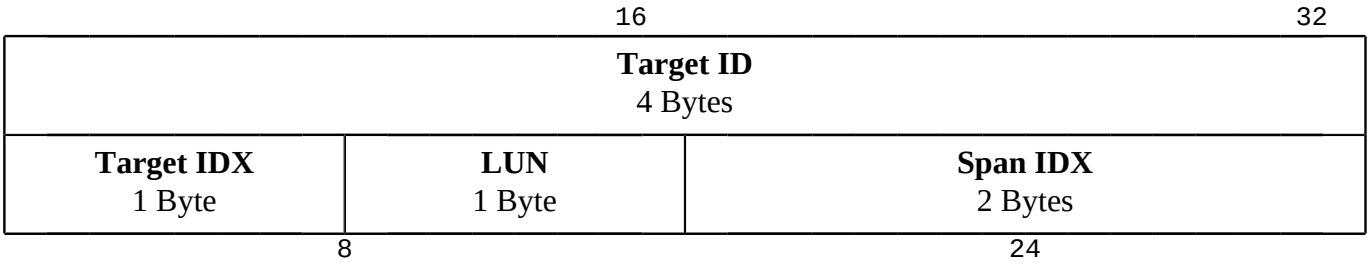
Camera

The camera index starting from 1.

Span Address

The address of the span the device exports. If this field is zero, no span is exported.

© Copyright 2001-2021 Bosch Security Systems



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

Target IDX

The index of the target.

LUN

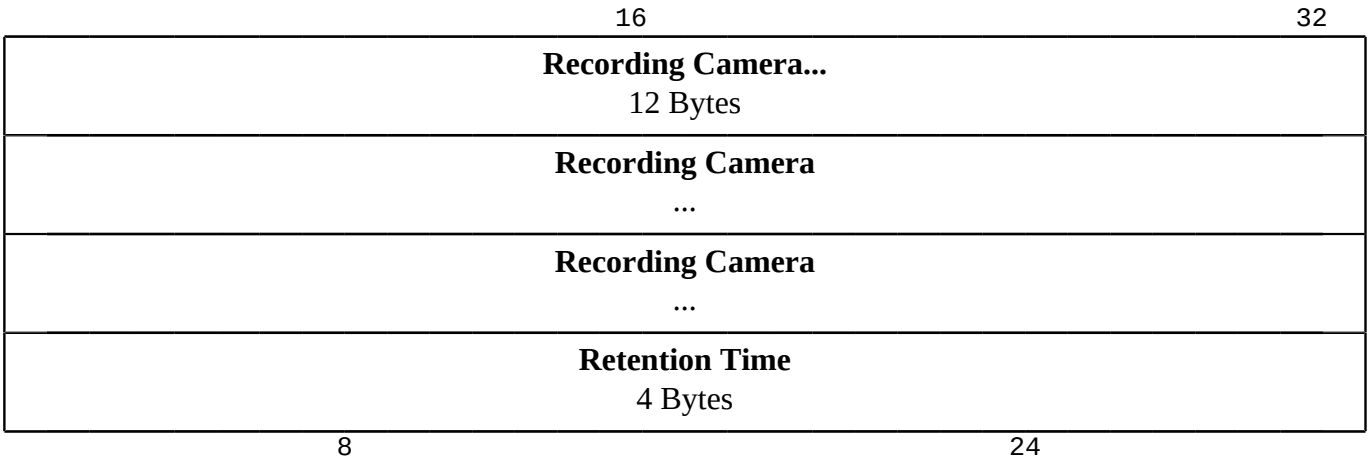
The logical unit number.

Span IDX

The index of the span on the lun.

Last Recording

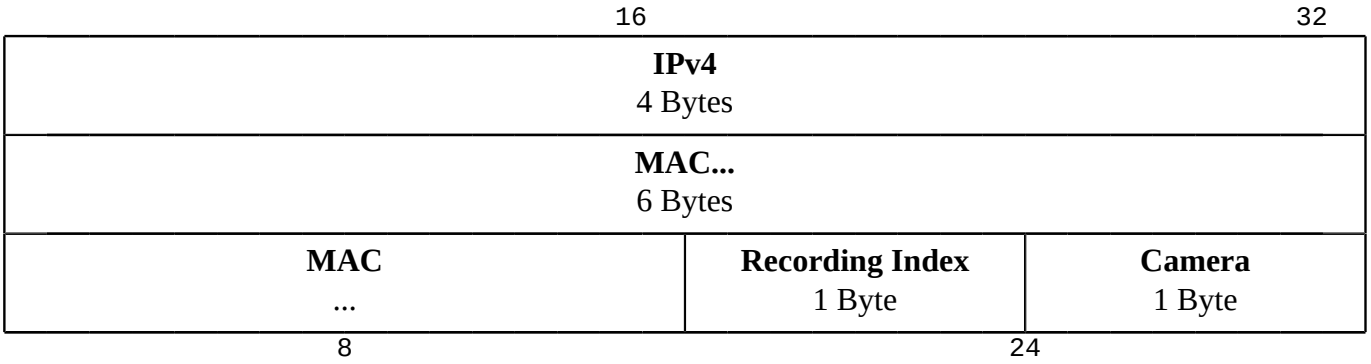
The address of the span the device exports.



Recording Camera

The id of the camera of the recording that is overwritten. Field is zero, if there was no recording on this span.

© Copyright 2001-2021 Bosch Security Systems



IPv4

The ipv4 address of the recording device. If the device hasn't an ipv4 Adress, this field should be set to invalid (zero ip 0.0.0.0). In that case a valid url has to be provided.

MAC

The hardware address of the recording device.

Recording Index

Primary Recording	1
Secondary Recording	2

Camera

The camera index starting from 1.

State

Values:

SUCCESS	0x00
NO SPAN AVAILABLE	0x01
STORAGE OFFLINE	0x02
INVALID SPAN MANAGER ADDRESS	0x03
ACCESS DENIED	0x04

Note: A list of span manager the recording is using can be obtained by the command CONF_REC_SPAN_MGR.

© Copyright 2001-2021 Bosch Security Systems

2.221 CONF_EXT_ENCODER_BITRATE_STATISTICS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c85	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command provides enhanced encoder bitrate statistics every record describes a table of bitrates. The table has a time base in sec typically sec, min, hour, days and weeks is provided. The first entry of the list can also only be active since a part of the cell e.g. cell time base is one day may this is only active since 300 sec. This time is signaled via lastCellIsActiveSinceSec

Definition of the global parameters, these parameters are vaild for all record entrys:

- GlobalParameters.localTimeSecoundsSince2000 local camera time which corresponds to the delivered entrys
- GlobalParameters.UTCSecoundsSince2000 utc time which corresponds to the delivered entrys

Payload Structure

16	32
GlobalParameters.localTimeSecoundsSince2000 4 Bytes	
GlobalParameters.UTCSecoundsSince2000 4 Bytes	
GlobalParameters.Reserved... 24 Bytes	
GlobalParameters.Reserved [...]	
GlobalParameters.Reserved ...	
Record [0] (see description)	

© Copyright 2001-2021 Bosch Security Systems

...
Record [N] (see description)

8 24

GlobalParameters.localTimeSecoundsSince2000

Local camera time which corresponds to the delivered entrys

GlobalParameters.UTCSecoundsSince2000

Utc time which corresponds to the delivered entrys

Record

16 32

TimeBaseInSec 4 Bytes
LastCellIsActiveSinceSec 4 Bytes
ElementCount 4 Bytes
Reserved... 20 Bytes
Reserved [...]
Reserved ...
Bitrate [0] 4 Bytes
...
Bitrate [ElementCount - 1] 4 Bytes

8 24

TimeBaseInSec

Base time for bitrate[1-n] base time of bitrate[0] is signaled via lastCellIsActiveSinceSec

© Copyright 2001-2021 Bosch Security Systems

LastCellIsActiveSinceSec

Bitrate[0] is active since lastCellIsActiveSinceSec

ElementCount

Number of elements

Bitrate

Example:

timeBaseInSec = 60, lastCellIsActiveSinceSec = 20 sec, elementCount = 256

- bitrate[1] : average bitrate{0 s ,...- 20 s}
- bitrate[2] : average bitrate{-20 s,...- 80 s}
- bitrate[3] : average bitrate{-80 s,...- 140 s}
- [...]
- bitrate[elementCount]

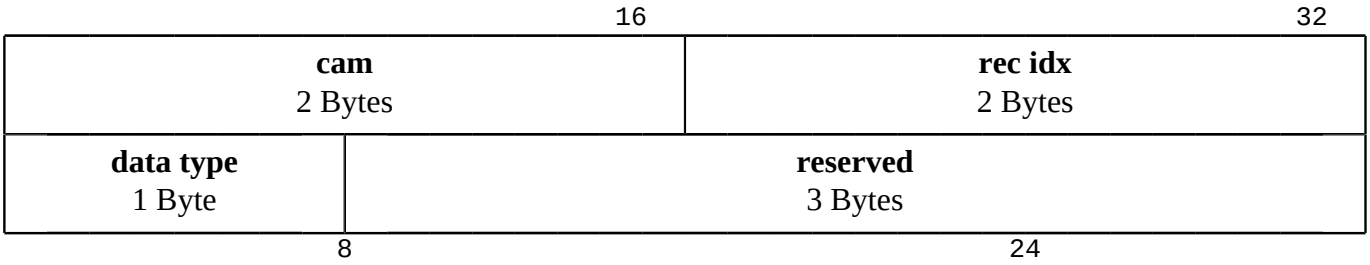
© Copyright 2001-2021 Bosch Security Systems

2.222 CONF_EXT_RECORDER_BITRATE_STATISTICS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c94	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Recording bitrates:	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure



cam

Camera index starting with 1

rec idx

Recording index

Values:

Primary Recording	1
Secondary Recording	2

data type

Values:

Video Data	0
Other Data	1

Response Payload Structure

This payload equals the payload of the command 'CONF_EXT_ENCODER_BITRATE_STATISTICS'

© Copyright 2001-2021 Bosch Security Systems

2.223 CONF_EXTERNAL_CLIENT

[API: extension]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c9e	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

Description

This command lists information about all running external clients or about a specific external client.

Num Descriptor Values

To get the information of a specific external client the num parameter is set to:	
Video App	1
User App	2
All Running Clients	0xFFFF

Response Payload Structure

Client Info [0] (see description)
...
Client Info [N] (see description)

Client Info

16			32
Line 1 Byte	**Instance** 1 Byte	**Length** 1 Byte	**Client Name...** *Length* Bytes
Client Name ...			
8		24	

© Copyright 2001-2021 Bosch Security Systems

Line
Video line (1...n), in case of user app this is 0

Instance
Client instance. In case of Video app this may be 0 or 1, in case of client app this may be 1 to 16.

Length
Length of client name

Client Name
Client name

Read Payload Structure

16 32
8 24

Write Payload Structure

Client Info [0] (see description)
...
Client Info [N] (see description)

Client Info

16 32

Line 1 Byte	**Instance** 1 Byte	**Length** 1 Byte	**Client Name...** *Length* Bytes
Client Name ...			

8 24

Line
Video line (1...n), in case of user app this is 0

© Copyright 2001-2021 Bosch Security Systems

Instance

Client instance. In case of Video app this may be 0 or 1, in case of client app this may be 1 to 16.

Length

Length of client name

Client Name

Client name

© Copyright 2001-2021 Bosch Security Systems

2.224 CONF_EXTERNAL_CLIENTS_LIST

[API: extension]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c9d	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Gets a list of installed external clients	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.225 CONF_FACTORY_DEFAULTS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09a0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Set the configuration to factory defaults and reset the board. ATTENTION: Wait till board has fully rebooted before switching of power, to ensure that defaults are applied completly. Also propagade this to the user, that he has to wait until the reboot is complete.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.226 CONF_FIRE_ALARM

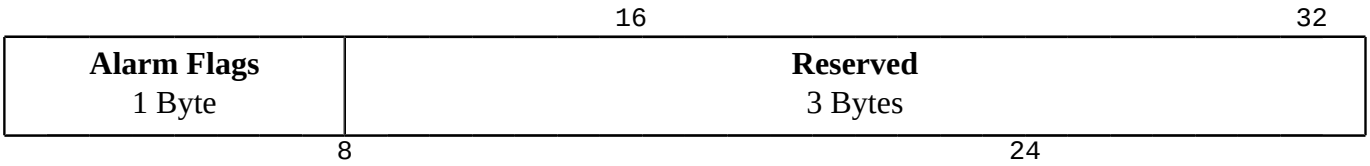
[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c58	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read the current state of each single fire alarm.

Payload Structure



Alarm Flags

These bits can be combined by using the bitwise or-operator.

Values:

	Mask	Name	Description
Bit 3	0x08	General	Fire trouble: General issue
Bit 2	0x04	Illumination	Fire trouble: Ilumination out of range
Bit 1	0x02	Smoke Alarm	Smoke detected
Bit 0	0x01	Flame Alarm	Flame detected

Reserved

Reserved bits, must be zero.

© Copyright 2001-2021 Bosch Security Systems

2.227 CONF_FLUSH_LUN_INFO_CACHE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b09	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Flush the lun info cache of lun: 4 bytes IP as DWORD in network byte order, 1 byte target idx, 1 byte lun, 2 bytes reserved (if ip is 0.0.0.0 or 255.255.255.255 or the payload is less than 6 bytes it is asumed as wildcard, the whole cache will be flushed)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.228 CONF_FORCE_TIME_SET

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the time, 8 bytes payload, offset 0: year (word); offset 2: month (octet); offset 3: day (octet); offset 4: hrs (octet); offset 5: min (octet); offset 6: sec (octet); offset 7: reserved (octet)	
Write	p_octet	service	Set the time, if recording is running it will be stopped and restarted, parameter 8 bytes payload, offset 0: year (word); offset 2: month (octet); offset 3: day (octet); offset 4: hrs (octet); offset 5: min (octet); offset 6: sec (octet); offset 7: reserved (octet)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

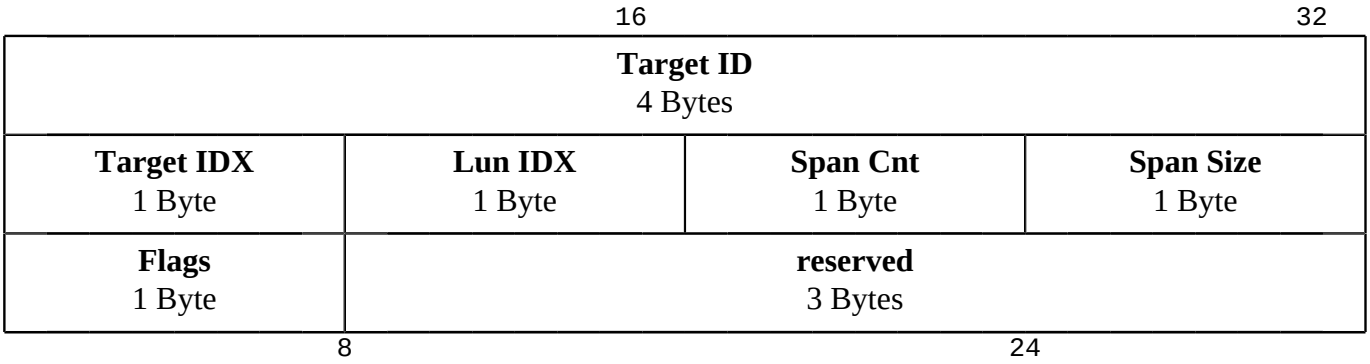
© Copyright 2001-2021 Bosch Security Systems

2.229 CONF_FORMAT_FS_SPAN

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service	Format the lun into spans. (not allowed while recording on this lun)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Target ID

Target ID of ISCSI Server(server ipv4 address if std TargetID resolve rule is applied)

Values:

Local USB	ff.ff.ff.ff	Local connected USB disk.
iSCSIServer	Any IP address	Valid ip address

Target IDX

Index of the iSCSI server target. The index of the desired target can be obtained from the reply to an iSCSI discover.

Lun IDX

Index of the lun of the specified target.

Span Cnt

Number of spans the lun shall be formated with

© Copyright 2001-2021 Bosch Security Systems

Span Size

Size of each span in megabyte. (1 MB = 1024 x 1024 Bytes)

Flags

These flags define the operations to be executed. If no flags are set (default), nothing will be done.

Format options:

REPAIR_FS	0x01	Repair file system without clearing spans
CLEAR_SPANS_ONLY	0x02	Only clear the spans

Command Specific Errors

SPAN_ERR_INTERNAL	0x01
SPAN_ERR_NOT_MNTD	0x05
SPAN_ERR_INV_FS	0x06
SPAN_ERR_INV_LUN_NFO	0x07
SPAN_ERR_RD_ONLY	0x0a
SPAN_ERROR_OLD_LUN_NFO	0x20
ISCSI_ERROR_CONNECT	0x31
ISCSI_ERROR_INV_LUN	0x33
ISCSI_ERROR_LOGIN	0x34
ISCSI_ERROR_INV_TRG_IDX	0x35

Note: The reply packet will contain the values of the actually formated spans and size. If the product of the span count and the span size in the request packet exceeds the size of the current lun, only as many as possible spans are formatted. If the product is less than the available lun space, the back part of the lun will be unused. Size of the write payload is at least 12 bytes, if the optional flags are uses the payload has to be at least 16 bytes.

Note: Note: A list of all defined error types can be found in the Appendix.

© Copyright 2001-2021 Bosch Security Systems

2.230 CONF_FORMAT_FS_SPANS_STATUS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09f5	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message reports the status and progress of a running format. The formatting includes the creation of the FAT32 file system (step: HD_STEP_FORMAT_FS) and clearing of the span header files(step: HD_STEP_CLEAR_SPAN_HEADER). The progress is reported about the whole formatting. A successful format should always end with a message with 100 percent progress at step HD_STEP_CLEAR_SPAN_HEADER and status should be HD_FORMAT_STATUS_FINISHED. A failed format should finish with a status HD_FORMAT_STATUS_FAILED at any progress and any step. After these final messages no futher messages will follow. The whole formatting takes two steps HD_STEP_FORMAT_FS and HD_STEP_CLEAR_SPAN_HEADER. First step HD_STEP_FORMAT_FS will run up to 90 percent and will end with a 90 percent progress message and status HD_FORMAT_STATUS_FINISHED if successful. Then the last step HD_STEP_CLEAR_SPAN_HEADER will follow from 90 to 100 percent and will also end with a HD_FORMAT_STATUS_FINISHED status if successfull, which is also as mentioned earlier the final success message. If the formatting runs with the option of disk erasing, The first step will be the HD_STEP_ERASE_DISK step and will run upto 70 percent if successful.

Payload Structure

16 32

Step 1 Byte	**Status** 1 Byte	**Progress** 2 Bytes
Ip 4 Bytes		
Target_idx 1 Byte	**Lun** 1 Byte	**Reserved** 2 Bytes

8 24

Step

Which step of the formating (FAT32 Formatting including creating spans and clearing span headers)

© Copyright 2001-2021 Bosch Security Systems

Values:

HD_STEP_FORMAT_FS	0x00
HD_STEP_CLEAR_SPAN_HEADER	0x02

Status

Reports the state of formatting. The format can be in state of running, failed or successfully finished. success means the creation of the fat32 file system and the empty spans or clearing the span headers was successfully.

Values:

HD_FORMAT_STATUS_FAILED	0xff
HD_FORMAT_STATUS_RUNNING	0x00
HD_FORMAT_STATUS_FINISHED	0x02
HD_FORMAT_ERASE_DISK	0x03

Progress

Value from 0 to 100 represent the progress of the format including the creation of the empty recording spans and clearing the header files.

Values:

Progress in percent	0 - 100

Ip

Ip of the storage device or lun, that is formatting

Target_idx

Target Index of the storage device from 0 to 255.

Lun

Lun of the storage device from 0 to max. 255.

Reserved

2 Bytes reserved

© Copyright 2001-2021 Bosch Security Systems

2.231 CONF_FORMAT_FS_STATUS

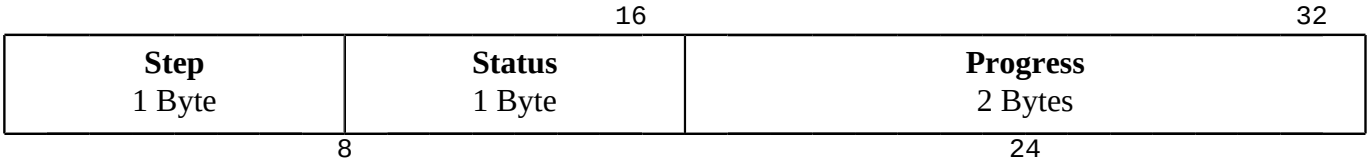
[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d1	file system nr, 1 - recording storage device, 2 - RAMDISK	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

the message reports the status and progress of a running format. The formatting takes two steps that is first formating FAT32 including creating the recording files and second copy the replay tools. The progress is reported about the whole formatting. The status reports the state of each formatting step. A successful format should always end with a message with 100 percent progress, step should be copy toolkit and status should be HD_FORMAT_STATUS_FINISHED or HD_FORMAT_STATUS_SKIPPED if the storage device is too small for recording (e.g. RAMDISK). A failed format should finish with a status HD_FORMAT_STATUS_FAILED at any progress and step. After this final messages no futher message will follow.

Payload Structure



Step

Which step of the formating (FAT32 Formatting including creating recording files or copy replay tools)

Values:

HD_STEP_FORMAT_FS	0x00
HD_STEP_COPY_TOOLKIT	0x01

Status

Reports the state of formatting. The format can be in state of running, failed or successfully finished. The last one is devided in the two states of HD_FORMAT_STATUS_FINISHED which means FAT32 including recording files and replay tools were created and HD_FORMAT_STATUS_SKIPPED which means that only the FAT32 was created because the storage device was not the needed size for recording.

© Copyright 2001-2021 Bosch Security Systems

Values:

HD_FORMAT_STATUS_FAILED	0xff
HD_FORMAT_STATUS_RUNNING	0x00
HD_FORMAT_STATUS_SKIPPED	0x01
HD_FORMAT_STATUS_FINISHED	0x02

Progress

Value from 0 to 100 represent the progress of the format including copy replay tools in percent.

Values:

Progress in percent	0 - 100

© Copyright 2001-2021 Bosch Security Systems

2.232 CONF_FTP_CDUP

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1f	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Changes to parent directory	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.233 CONF_FTP_CWD

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1e	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Changes the working directory	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.234 CONF_FTP_FILE_NAME

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b35	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	User defined ftp file name for the first recording or live instance: request: 1st OCTET inst, 2nd OCTET mode(0->rec, 1->live), 64 OCTETs reserved response : 1st OCTET inst, 2nd OCTET mode(0->rec, 1->live), 64 OCTETs filename in arbitrary order of : b begin date / time, %e end date / time, %c camera name, %i export job id, %f file nbr, %a alarm description, %dSubDirectory sub directory to write, %nName name %l line; optionally seperated by _\" or \"-\")	
Write	p_octet	service	See read	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.235 CONF_FTP_PWD

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1b	SessionId: optional parameter to identifiy a already existing ftp client session	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Gets the working directory of the configured ftp server	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.236 CONF_FTP_SERVER_PORT

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b9f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the Port of the camera internal FTP server for loading files to the camera; Value 0 turns server off	
Write	t_word	service	Set the Port of the camera internal FTP server for loading files to the camera; Value 0 turns server off	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.237 CONF_FW_MIN_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bab	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the required minimum firmware version. Format: (hexadecimal) 0xrrrrVVMM, where 'r' is reserved, 'VV' is the major version and 'MM' the minor version number. (E.g. 0x00000590 for version 5.90)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.238 CONF_GATEWAY_IP_STR

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x007f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the gateway IP using string notation (xxx.xxx.xxx.xxx)	
Write	p_string	service	Set the gateway IP using string notation (xxx.xxx.xxx.xxx)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.239 CONF_GATEWAY_IP_V6_STRING

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b11	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	IPv6 Gateway IP string or domain name	
Write	p_string	service	IPv6 Gateway IP string or domain name	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.240 CONF_GB28181

[API: GB28181]

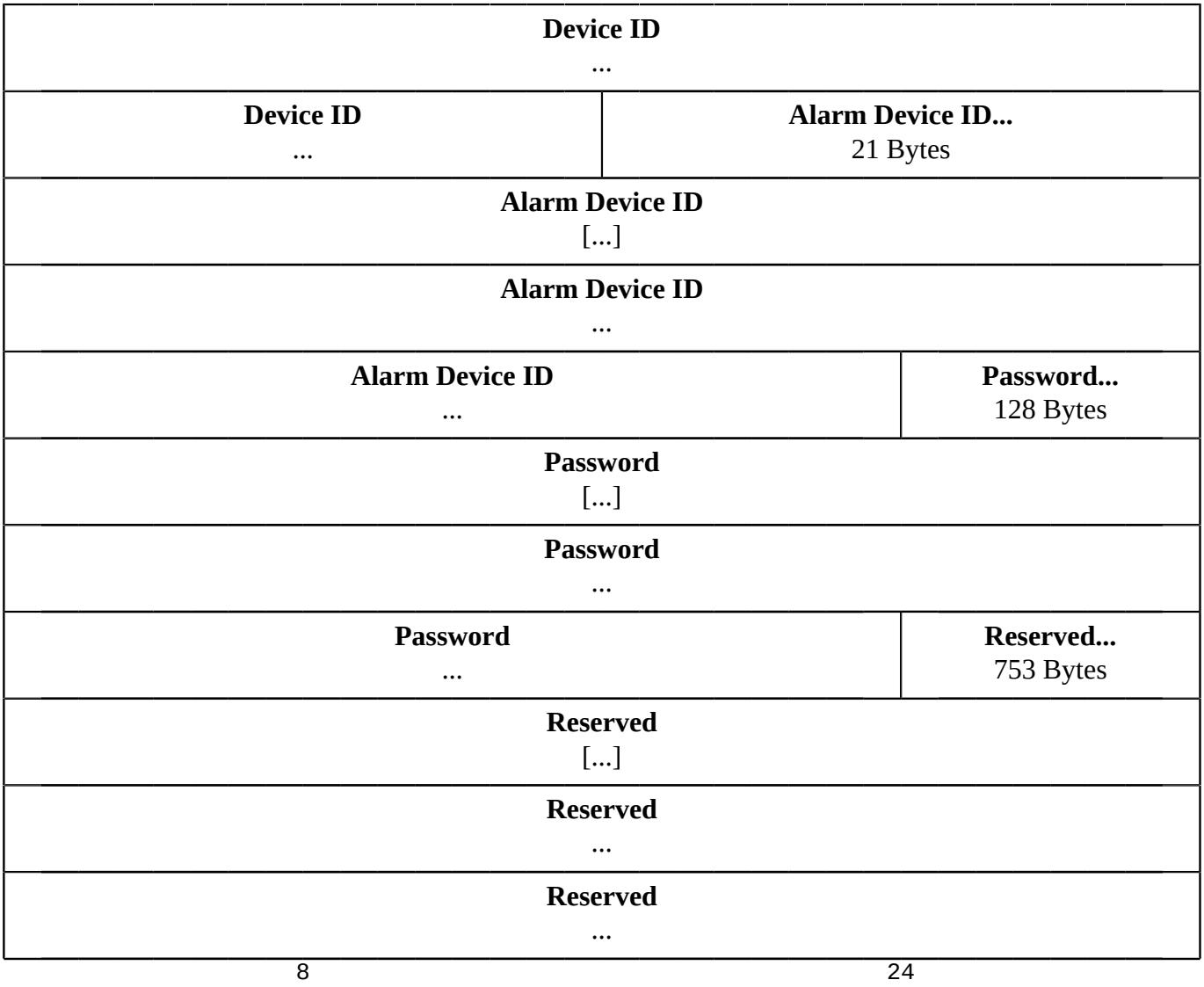
	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16 32

Enable 1 Byte	**H264 Stream Cfg** 1 Byte	**Reserved** 2 Bytes
Heart beat time out 4 Bytes		
Registration time out 4 Bytes		
Server port 2 Bytes	**Device Port** 2 Bytes	
Server URL... 64 Bytes		
Server URL [...]		
Server URL ...		
Server ID... 21 Bytes		
Server ID [...]		
Server ID ...		
Server ID ...	**Device ID...** 21 Bytes	
Device ID [...]		

© Copyright 2001-2021 Bosch Security Systems



Enable

1: Enable GB28181 client 0: Disable

H264 Stream Cfg

If 1, use h264 elementary stream (including startcodes) directly packed into RTP packets (non-RFC conform, but needed for GB28181 test tool)

Heart beat time out

Define heart beat time out in seconds max. one day min. 5 seconds

Registration time out

Define registration time out in seconds max. one day min. 5 seconds

© Copyright 2001-2021 Bosch Security Systems

Server port

Currently 5060 or 5511 can be used (server and device port must be equivalent server port is also used for device port)

Device Port

Currently 5060 or 5511 can be used (server and device port must be equivalent server port is also used for device port)

Server URL

URL of GB28181 server

© Copyright 2001-2021 Bosch Security Systems

2.241 CONF_GET_ENC_DYN_SCENE_CTRL_INFO

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb8	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	See detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

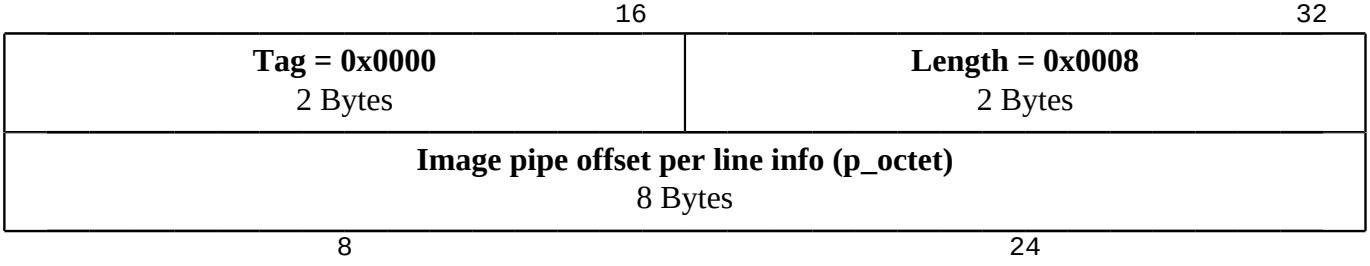
Length

Length of tagged value without length and tag field

Tag

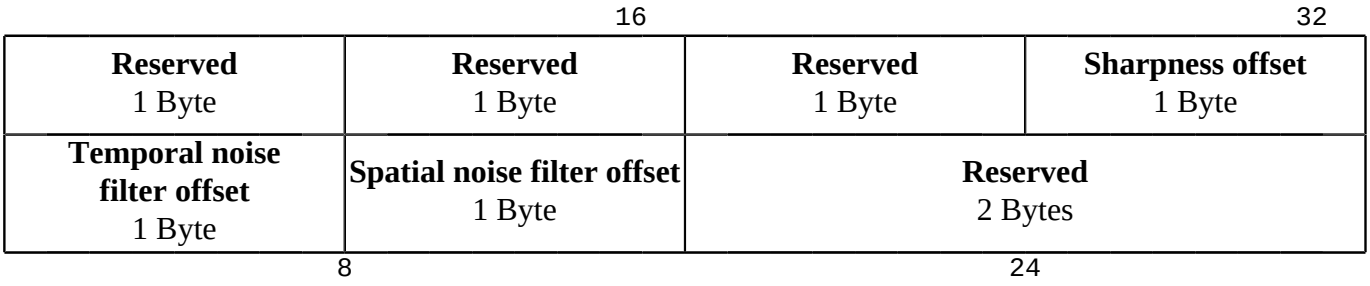
Tag specifying the encoding and meaning of the value

Tag 0: Image pipe offset per line info



© Copyright 2001-2021 Bosch Security Systems

Image pipe offset per line info



Tag 1: Image pipe bitrate modifier info

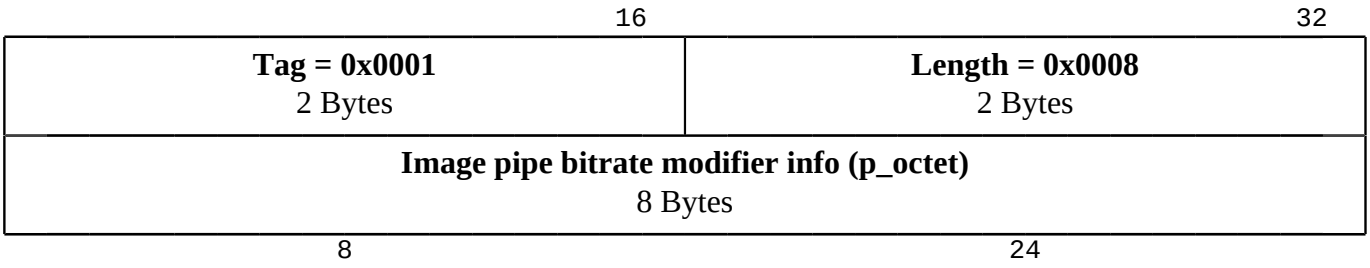
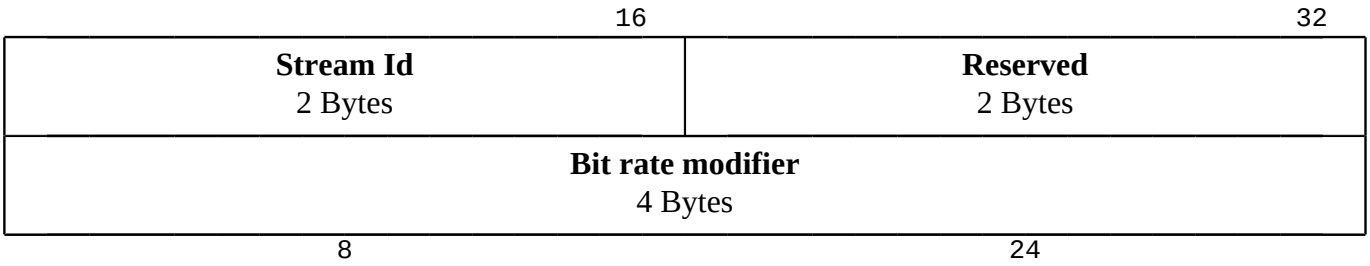


Image pipe bitrate modifier info



Stream Id

1...n associated stream

Bit rate modifier

Base unit 1/1000 e.g. 900 -> 0.9 bitrate after interaction of encoder with image pipe is reduced by up to 10 %

© Copyright 2001-2021 Bosch Security Systems

2.242 CONF_GET_RTSP_SESSION_ID

[API: rtsp settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae8	random value from (Rtsp Session setup)	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	live	Gets the rcp session id of the rtsp session (identified by the random value)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.243
CONF_GET_VIDEO_ENC_P_REF_LIST_SIZE_LIMIT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0628	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the largest value supported by VIDEO_ENC_P_REF_LIST_SIZE; 0 indicates that the value is not user-adjustible	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.244 CONF_GETPROFILE_ALGO_PRESET_NAME_LIST

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6c	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command returns a list of VCA profile specific information. These information contains the VCA algorithm type, the associated dome preset (if existent otherwise zero) and the profile name.

Payload Structure

16	32
Num Profiles 1 Byte	**Reserved** 1 Byte	**Profile [0]** (see description)
...	**Profile [Num Profiles - 1]** (see description)
8	24

Num Profiles

Reserved

Profile

16	32
Algorithm Type 2 Bytes	**Preset** 2 Bytes
Profile Name (p_unicode)... 32 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Profile Name [...]
Profile Name ...

8 24

Algorithm Type

	Mask	Name	Description
Bit 2	0x0004	Block Based	Block Based VCA (Motion)
Bit 1	0x0002	Flow Based	
Bit 0	0x0001	Object Based	Object Based VCA (IVA, EVA)

© Copyright 2001-2021 Bosch Security Systems

2.245 CONF_GOP_STRUCTURE_OPTIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bef	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get a list of options, which can be selected by CONF_MPEG4_AVC_GOP_STRUCTURE options are defined by MPEG4_AVC_GOP_STRUCTURE (Format: Byte 0 (len); Byte (1-len) supported options)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.246
CONF_H264_ENC_BASE_OP_MODE_CAPS_VERBOSE

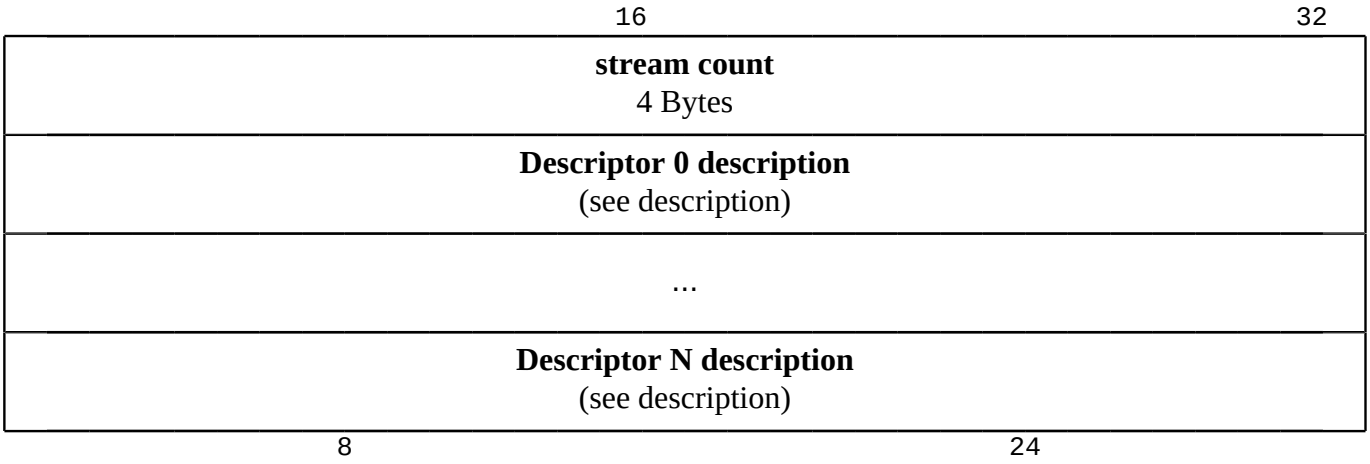
[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c7c	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

encBaseModeId: id used in CONF_VIDEO_H264_ENC_BASE_OPERATION_MODE height: video format height of base mode with encBaseModeId 0 -> height is not fixed width: video format width of base mode with encBaseModeId 0 -> width is not fixed copy: base mode is copy of an other stream sd: base mode is sd, base mode resolution is determined by encoder profile sdFramerateDependent: base mode resolution depends on frame rate crop: base mode is a croped view skip: base mode is frame rate reduced (applys a skip on the full frame rate) roi: base mode is a region of interest stream typically ptz is supported ptz: base mode allows ptz dualStream: base mode supports "dual stream" two encoders are used for the stream (e.g. 2 different ptz views are possible) exclusive: only used in combination with 'dualStream' property. Indicates if the additional stream of the 'dual stream' is for exclusive access (only one client at a time) or not.

Payload Structure



stream count

Array index in text

© Copyright 2001-2021 Bosch Security Systems

Descriptor description

Description for stream [0] (see description)
…
Description for stream [stream count - 1] (see description)

Description for stream

16 32

encBaseModeId 4 Bytes			
height 4 Bytes			
width 4 Bytes			
copy 1 Byte	**sd** 1 Byte	**sdFramerateDependent** 1 Byte	**crop** 1 Byte
skip 1 Byte	**roi** 1 Byte	**ptz** 1 Byte	**dualStream** 1 Byte
exclusive 1 Byte	**reserved ...** 15 Bytes		
reserved ...			
reserved ...			
reserved ...			

8 24

encBaseModeId

Id used in CONF_VIDEO_H264_ENC_BASE_OPERATION_MODE

© Copyright 2001-2021 Bosch Security Systems

height

Video format height of base mode with encBaseModeId 0 -> height is not fixed

width

video format width of base mode with encBaseModeId 0 -> width is not fixed

copy

Base mode is copy of another stream

No	0
Yes	1

sd

Base mode is sd, base mode resolution is determined by encoder profile

No	0
Yes	1

sdFramerateDependent

Base mode resolution depends on frame rate

No	0
Yes	1

crop

Base mode is a croped view

No	0
Yes	1

skip

Base mode is frame rate reduced (applys a skip on the full frame rate)

No	0
Yes	1

roi

Base mode is a region of interest stream typically ptz is supported

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

ptz

Base mode allows ptz

No	0
Yes	1

dualStream

Base mode supports 'dual stream' two encoders are used for the stream (e.g. 2 different ptz views are possible)

No	0
Yes	1

exclusive

only used in combination with 'dualStream' property. Indicates if the additional stream of the 'dual stream' is for exclusive access (only one client at a time) or not.

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

2.247 CONF_HARDWARE_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string t_dword	always	Read the hardware version	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.248 CONF_HD_FILE_INFO

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x091d	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message is used for updating actual recording files, it will be send periodically about every second by the recording device, that is running the recording on that file. The message informs about the start and end time changes of an file. For uniqe idetification, the message includes the span address and a file id. it also has the information about the recording cam (attention: in older firmware the file info msg only has the first 4 payload fields(from start time to file id))

Payload Structure

16 32

start time 4 Bytes		
end time 4 Bytes		
flags 4 Bytes		
file id 4 Bytes		
target id 4 Bytes		
target idx 1 Byte	**lun** 1 Byte	**span idx** 2 Bytes
cam 1 Byte	**recording idx** 1 Byte	**reserverd** 2 Bytes

8 24

start time

Start time in seconds since 2000 of the recording file

© Copyright 2001-2021 Bosch Security Systems

end time

End time in seconds since 2000 of the recording file

flags

File info flags, see CONF_HD_PARTITION_FILE_INFO

file id

File ID on the file unique per span

target id

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

target idx

Target index of the span

lun

Lun of the span

span idx

Span index

cam

Camera from 1 to ...

recording idx

Recording index (primary or secondary)

Values:

primary	1
secondary	2

reserverd

© Copyright 2001-2021 Bosch Security Systems

2.249 CONF_HD_FILE_INFO_SECONDARY

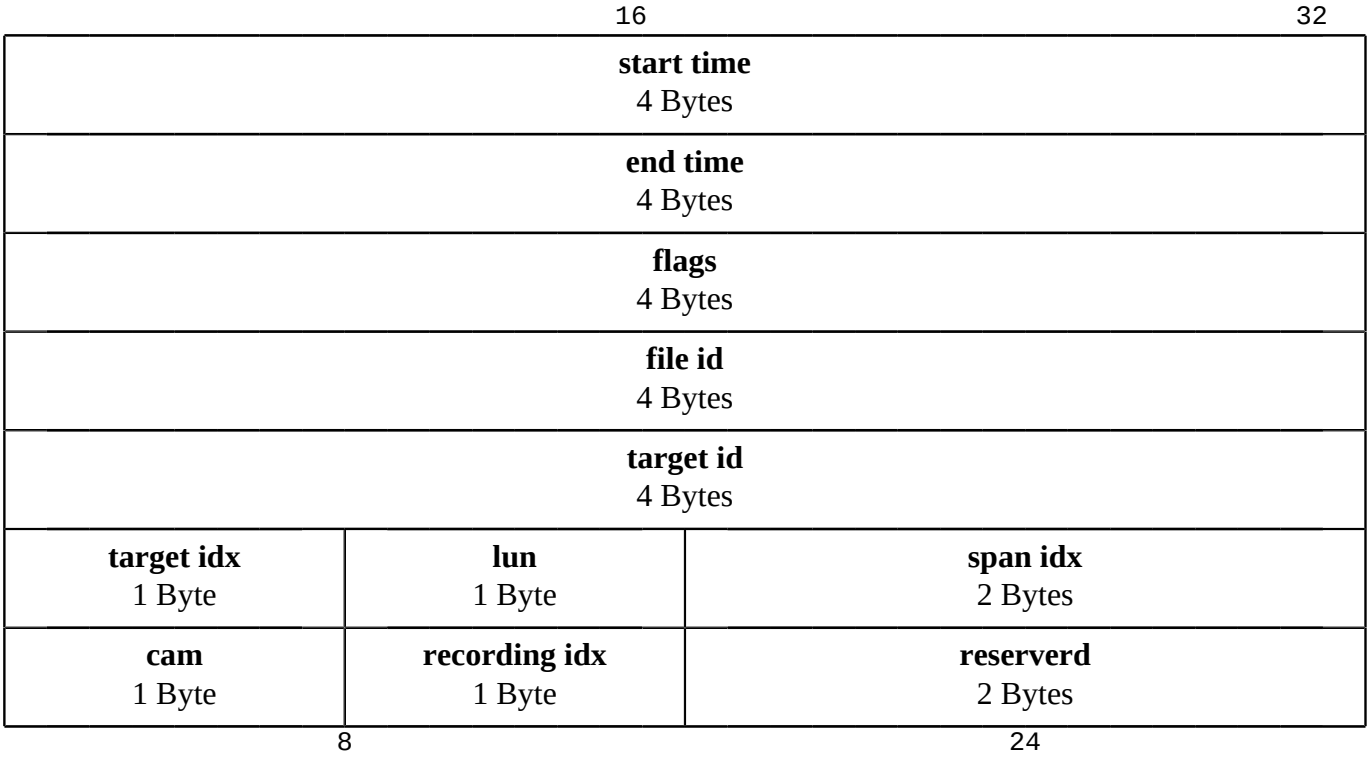
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a64	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message is used for updating actual recording files, it will be send periodically about every second by the recording device, that is running the recording on that file. The message informs about the start and end time changes of an file. For uniqe idetification, the message includes the span address and a file id. it also has the information about the recording cam (attention: in older firmware the file info msg only has the first 4 payload fields(from start time to file id))

Payload Structure



start time

Start time in seconds since 2000 of the recording file

© Copyright 2001-2021 Bosch Security Systems

end time

End time in seconds since 2000 of the recording file

flags

File info flags, see CONF_HD_PARTITION_FILE_INFO

file id

File ID on the file unique per span

target id

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

target idx

Target index of the span

lun

Lun of the span

span idx

Span index

cam

Camera from 1 to ...

recording idx

Recording index (primary or secondary)

Values:

primary	1
secondary	2

reserverd

© Copyright 2001-2021 Bosch Security Systems

2.250 CONF_HD_FORMAT_STORAGE_ERROR_LOG_CNT

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c65	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of failed local storage formats	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.251 CONF_HD_MGR_REC_STATUS

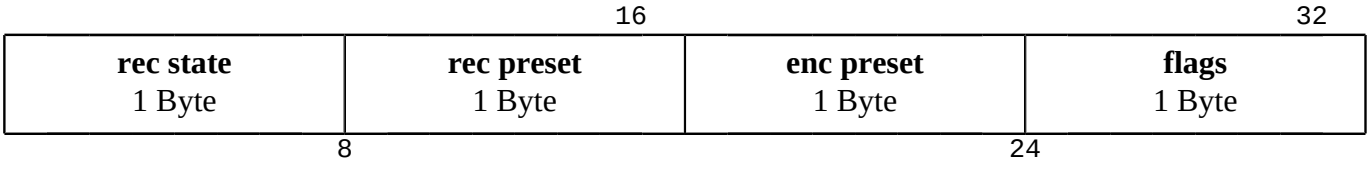
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aae	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command shows the state for a recording. It can be read and it will be send at state changes. State changes are the change of rec state or rec preset only. Within the Msg there is no distiction between the state OFF and NO RECORDING. That means no msg will be send on the state change between this two states. The msg will never contain the state OFF but the read response maybe. OFF state means the recording is configured to off by set it to stop. NO RECORDING can be caused by many things e.g. no recording on the schedule, no storage present and so on. STAND BY means there is recording on the schedule but not at the moment. The recording scheduler waits for the time to start the recording. All the other states indicating a running recording. In case of alarm recording in prealarm state, the message state will be PRE ALARM RECORDING even if the pre alarm recording takes place in the ram only, if at least the storage is connected. If there is no storahge connected and recording is configured for running, the state will alway be NO RECORDING.

Payload Structure



rec state

State of the recording, state OFF not in payload of msg

Values:

OFF	0
NO RECORDING	1
STAND BY	2
PRE ALARM RECORDING	3
ALARM RECORDING	4
POST ALARM RECORDING	5

© Copyright 2001-2021 Bosch Security Systems

rec preset

Actual used recording preset from 1 to 10 or 0 if no preset is used

enc preset

Actual used encoder preset from 1 to 8 or 0 if no preset is used

flags

These flags show the alarm states and the recording mode

	Mask	Name
Bit 7	0x80	reserved for extension
Bit 4	0x10	virtual alarm
Bit 3	0x08	video loss
Bit 2	0x04	motion alarm
Bit 1	0x02	input alarm
Bit 0	0x01	alarm recording mode

© Copyright 2001-2021 Bosch Security Systems

2.252 CONF_HD_MGR_REC_STATUS_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aaf	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Same as CONF_HD_MGR_REC_STATUS but for secondary recording (see cmd CONF_HD_MGR_REC_STATUS)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

This payload equals the payload of the command 'CONF_HD_MGR_REC_STATUS'

© Copyright 2001-2021 Bosch Security Systems

2.253 CONF_HD_MGR_SIGNAL_ALARM

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0915	yes (cam or all cams for 0)	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Signal alarm: only one alarm per second is allowed,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.254 CONF_HD_MGR_START

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0913	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Return TRUE when manager is on or when primary span recording is on	
Write	f_flag	service	Start/stop the recording manager and set config to recording on:1/ off:0. Can cause recording if time/ alarm recording selected.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.255 CONF_HD_MGR_START_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a46	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Return TRUE when secondary span recording is on	
Write	f_flag	service	Start/stop the recording manager and set config to recording on:1/ off:0. Can cause recording if time/ alarm recording selected.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.256 CONF_HD_MGR_STOP

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0914	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Return TRUE when manager is off or when primary span recording is off	
Write	f_flag	service	Stop/start the recording manager and set config to primary recording off:1/on:0.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.257 CONF_HD_MGR_STOP_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a47	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Return TRUE when secondary span recording is off	
Write	f_flag	service	Stop/start the recording manager and set config to secondary recording off:1/on:0.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.258 CONF_HD_PARTITION_FILE_INFO

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0901	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns a list of files of a replay session (session id needed). This command works for local replay only. See detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command initiates a search to get the times where data is available within a recording.

The Session ID paramter must be set (a connect primitive must have been preceded). Recording mode can be time recording (1) or alarm recording (2 and 3), only for command CONF_SPAN_PARTITION_FILE_INFO there are seperagte files for pre (2) and post (3) alarm recording.

Search is performed from a start point on the timeline towards an end point. If the end point is prior to the start point search is performed backwards. To get all available results of an recording a search from 0 to 0xFFFFFFFF needs to be done.

The number of results is limited to 256 per response. The result are always in ascending order independent if the search is performed forward or backward. The File ID always increases on span recording regions if new files will be created. To get all results the command has to be called multiple times.

In that case the start or stop times needs to be updated with every call: If a forward search is executed the start time of the next request must be the stop time of the last retrieved result. If a bachward search is executed the start time of the next request must be the start time of the first retrieved result.

A search is completed if the number of returned results is smaller than 256 or if the endpoint of the search is greater or equal than than the endpoint of the last result. In case of forward search a virtual start point (equal to the start point of the search) will be inserted if the search start point is range with recording.

Request Payload Structure

16	32
Search Start Time 4 Bytes	
Search Stop Time 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Max Entries 4 Bytes
Flags 4 Bytes

8 24

Search Start Time

Seconds since 2000

Search Stop Time

Seconds since 2000

Max Entries

Max Number of entries, limited to 256

Entries 1 - 256

Flags

	Mask	Name	Description
Bit 1	0x00000002	UTC_FLAG	Start and stop time are utc times (request)
Bit 0	0x00000001	YOUNGER_OR_EQUAL_FLAG	Must be set in case of forward search

Response Payload Structure

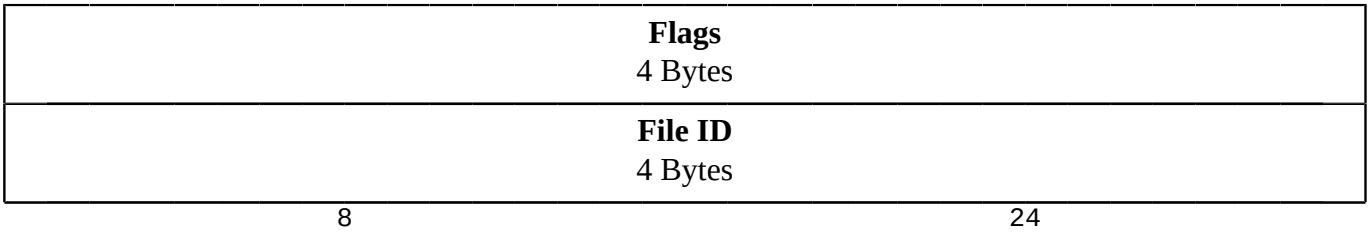
RecordingEntry [0] ... (see description)
RecordingEntry ...
RecordingEntry ...
RecordingEntry ...

RecordingEntry

16 32

Start Time 4 Bytes
Stop Time 4 Bytes

© Copyright 2001-2021 Bosch Security Systems



Start Time

Seconds since 2000

Stop Time

Seconds since 2000

Flags

	Mask	Name	Description
Bit 29	0x20000000	Time Zone Sign	
Bit 28	0x10000000	Time Zone	
Bit 27	0x08000000	Time Zone	
Bit 26	0x04000000	Time Zone	
Bit 25	0x02000000	Time Zone	
Bit 24	0x01000000	Time Zone	
Bit 23	0x00800000	Time Zone	
Bit 22	0x00400000	Protected	(VRM only)
Bit 21	0x00200000	Offline	(VRM only)
Bit 18	0x00040000	reserved	
Bit 17	0x00020000	reserved	
Bit 16	0x00010000	Alarm Remote	(there are virtual/remote alarms in this file, see CONF_HD_MGR_SIGNAL_ALARM)
Bit 15	0x00008000	Track Fill Level	
Bit 14	0x00004000	Track Fill Level	
Bit 13	0x00002000	Track Fill Level	
Bit 12	0x00001000	Track Fill Level	
Bit 11	0x00000800	Track Fill Level	
Bit 10	0x00000400	Track Fill Level	
Bit 9	0x00000200	Track Fill Level	
Bit 8	0x00000100	Track Fill Level	
Bit 7	0x00000080	Recording mode	
Bit 6	0x00000040	Recording mode	
Bit 5	0x00000020	Video Loss	(there are video loss in this file)
Bit 4	0x00000010	New Alarm	(obsolete)
Bit 3	0x00000008	Alarm Motion	(there are motion alarms in this file)
Bit 2	0x00000004	Alarm Input	(there are input alarms in this file)
Bit 1	0x00000002	Recording Overwriting	(recording takes place in a ring and old recording data will be overwritten)
Bit 0	0x00000001	Recording Running	(actual recording is running on this file or recording not closed regulary)

© Copyright 2001-2021 Bosch Security Systems

Combined Values

Mask	Name	Description
0x1F800000	Time Zone	Quarter hours
0x0000FF00	Track Fill Level	Fill level in percent, always 100 % on filled ring recording
0x0000C000	Recording Mode	Time Recording 1 Alarm recording (pre alarm) 2 Alarm recording (post alarm) 3 Values 2 and 3 only in CONF_SPAN_PARTITION_FILE_INFO distinguishable, for CONF_PARTITION_FILE_INFO these two values have the meaning of a full alarm rec file

Example

Forward search (data is available from time A to time B):

- A search is executed from time 0 to time B. In that case the reponse is: secStart=A, secStop=B
- A search is executed from time 0 to time 0xFFFFFFFF. In that case the reponse is: secStart=A, secStop=B
- A search is executed from time C (C > A and C < B) to time B. In that case the reponse is: secStart=C, secStop=B
- A search is executed from time A to time C (C > A and C < B). In that case the reponse is: secStart=A, secStop=C

Backward search (data is available from time A to time B):

- A search is executed from time 0xFFFFFFFF to time A. In that case the reponse is: secStart=A, secStop=B
- A search is executed from time 0xFFFFFFFF to time 0. In that case the reponse is: secStart=A, secStop=B
- A search is executed from time C (C < B and C > A) to time A. In that case the reponse is: secStart=A, secStop=C
- A search is executed from time B to time C (C < B and C > A). In that case the reponse is: secStart=C, secStop=B

© Copyright 2001-2021 Bosch Security Systems

2.259 CONF_HD_PARTITION_RECORDING

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a03	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Recording status of cam for primary recording (1 = running, 0 = not running)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.260
CONF_HD_PARTITION_RECORDING_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a4d	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Recording status of cam for secondary recording (1 = running, 0 = not running)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.261 CONF_HD_PARTITIONS_RECORDING

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x091a	yes (1 - primary recording, 2 - secondary recording)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List of cams with running recording	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.262 CONF_HD_REC_BUFFER

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b8f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command gets information about the recording buffer size, it includes the full amount of buffer for all recordings, the allocated, and the minimum amount of buffer of the recording specified by the in payload. The "full buffer bytes" is usually preallocated memory at startup and won't change it's size as long as the device is running and is the same for all recordings (except streaming gateway). The "min buffer bytes" is the amount of buffer the device will at least allocate if running. The "allocated buffer bytes" is the amount of buffer the recording has actually allocated. This value can be higher than "min buffer bytes", if nt all possible recrdings are running. The unused buffer of non running recordings will be used by the running ones. If the recording is deaktivated, the value will be 0. When running, the value wil be usually at least "min buffer bytes", but it can be temporary less than that value, if e.g. the recording was started shortly before and another already running recording has allocated the whole buffer. In that case it could take some time free the buffer by the running recording, so the allocated buffer will increase over the time. The "filled buffer bytes" is the amount of buffer memory of the the "allocated buffer bytes", which contains data for recording or in case of alarm recording, the amout to be recorded, if an alarm would occur in this moment. The "filled_buffer bytes" can therefor be used to calculate the fill level of the recording buffer.

Request Payload Structure

16	32
Cam 2 Bytes	**Rec idx** 2 Bytes
8	24

Cam

Camera starting with 1 (in payload)

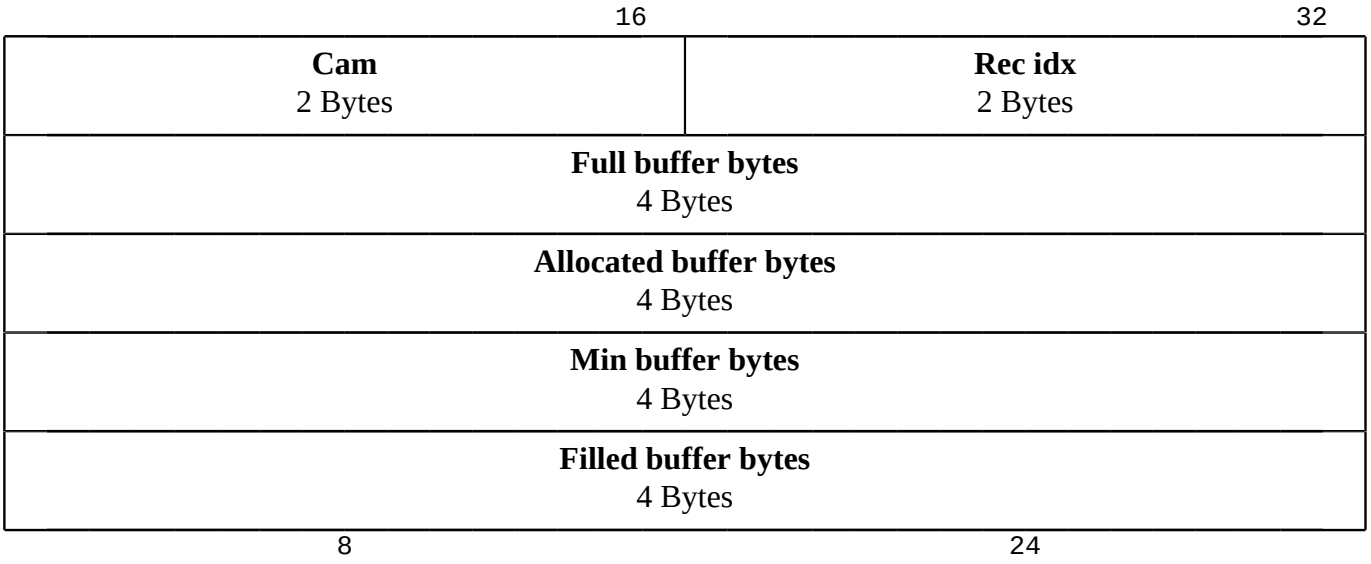
Rec idx

Recording index (in payload)

Primary Recording	1	
Secondary Recording	2	(active low)

© Copyright 2001-2021 Bosch Security Systems

Response Payload Structure



Cam

Camera starting with 1 (in payload)

Rec idx

Recording index (in payload)

Primary Recording	1	
Secondary Recording	2	(active low)

Full buffer bytes

Full amount of buffer in bytes for recording on this device

Allocated buffer bytes

Actual allocated buffer for this recording

Min buffer bytes

The minimum amount of buffer the recording would allocate

Filled buffer bytes

The amount of buffer that contains data for recording (not the data which are recorded but still in buffer)

© Copyright 2001-2021 Bosch Security Systems

2.263 CONF_HD_RECORD_HOLIDAYS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read holiday schedule list for primary recording, see detailed description	
Write	p_octet	service	Write holiday schedule list (effect takes place imidately), see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

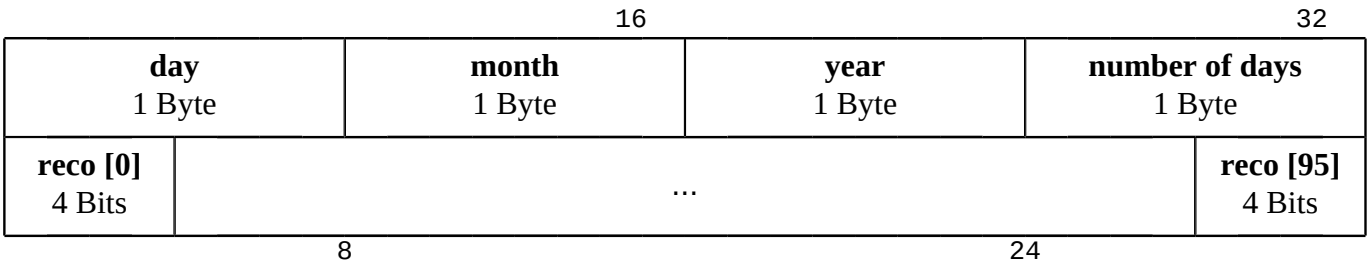
This command reads/writes the Holiday table for the primary recording. On holidays the recording uses the special schedule instead of the standard schedule. The table contains N entries (max. 25) each entry specifies one or more holidays in a row followd by a recording schedule. the payload sends only valid holiday entries, so the size of N depends on the number of valid entries in the table.

Payload Structure

holiday schedule entry [0] (see description)
...
holiday schedule entry [N] (see description)

holiday schedule entry

Schedule entry specifing a holiday and it's recording schedule. max. 25 entries.



© Copyright 2001-2021 Bosch Security Systems

day

Day of the month, which is a holiday

Values:

invalid	0
day of the month	1 - 31

month

Month of the year

Values:

invalid	0
january	1
february	2
march	3
april	4
may	5
june	6
july	7
august	8
september	9
october	10
november	11
december	12

year

Year since 2000

Values:

year 2000 to 2255	0 - 255

number of days

Number of days that follow the specified holiday, which are also scheduled as holidays. If only the specified date is the holiday, this field should be set to 1.

Values:

invalid	0
number of days	1 - 255

reco

This is the recocording schedule containing 96 entries of recording profile numbers for the holiday, each represents the recording profile for a 15 min time period. First entry is from 00:00 to 00:15. The following entries are for the following 15 min time periods until 24:00.

© Copyright 2001-2021 Bosch Security Systems

Values:

recording off	0
recording profile numbers	1 - 10

© Copyright 2001-2021 Bosch Security Systems

2.264 CONF_HD_RECORD_PROFILES

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0d	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read recording profiles, see detailed description	
Write	p_octet	service	Set recording profiles (effect takes place imidiatly), see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

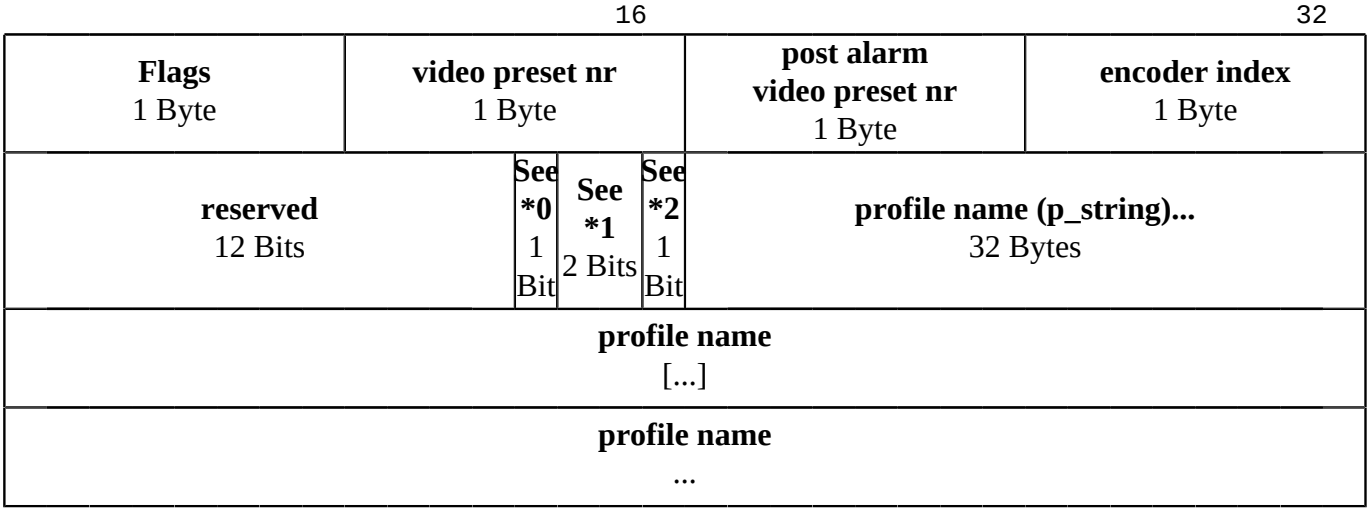
Description

Read/Write up to ten recording profiles (at least one) for a camera, payload upto 560 bytes total. The profiles will be written to the device config. These profiles are used as default profiles for each span that is mounted by this cam for span recording. If less than 10 profiles will be written, the remaining profiles stay unchanged.

Payload Structure

Recording Profile [0] (see description)
...
Recording Profile [9] (see description)

Recording Profile



© Copyright 2001-2021 Bosch Security Systems

profile name (p_string) ...	**max pre alarm time** 2 Bytes
pre alarm time 2 Bytes	**post alarm time** 2 Bytes
motion alarm 4 Bytes	
alarm input and virtual alarm 4 Bytes	
video loss alarm 4 Bytes	

8 24

Flags

	Mask	Name
Bit 6	0x40	MANAGED_BY_ONVIF
Bit 5	0x20	IMMEDIATE_ALARM_BACKUP
Bit 4	0x10	ALARM_RECORDING_PRE_ALARM_BUFFER
Bit 3	0x08	ALARM_FILE_BACKUP
Bit 2	0x04	META_RECORDING_DISABLE
Bit 1	0x02	AUDIO_RECORDING_DISABLE
Bit 0	0x01	ALARM_RECORDING_PRE_ALARM_RING
	Combined Values	
	Mask	**Name**
	0x11	ALARM_RECORDING_PRE_ALARM_AUTO

video preset nr

Video preset number used for recording

Values:

no recording on this profile	0
video preset nr	1 - 8

post alarm video preset nr

Video preset number used for recording after occurence of an alarm

Values:

Not set	0	uses video preset nr if set
video preset nr for post alarm	1 - 8	

© Copyright 2001-2021 Bosch Security Systems

encoder index

Index of the encoder stream starting with 0, index 255 is for the backup output

cont rec backup

Only for continuous buffered recording, if set the records will be backuped to the recording, its an alternativ way of configuration to using a backup account of typ record, that way no account is wasted in the configuration

Back Up Account

Backup account Number from 0 (first account) to 3(fourth account), for choosing the back up account in case activated ALARM_FILE_BACKUP flag

profile name

Profile name is a zero terminated string

max pre alarm time

Only relevant for backup recording, it needs to know the max pre alarm time, in other cases it should be set to zero

pre alarm time

Only relevant for alarm recording, if set, this time in seconds is the time the recording will be stored up to the alarm event

post alarm time

If set, this time in seconds is the time after an alarm event for that the post alarm video preset is used if set. After this time elapses, the recording will return to the standard video preset nr, if alarm recording, the recording will also stop the recording on the actual track and start a new one

motion alarm

The bits represent the activation

	Mask	Name
Bit 31	0x80000000	Alarm Nbr. 32
...		
Bit 1	0x00000002	Alarm Nbr. 2
Bit 0	0x00000001	Alarm Nbr. 1

alarm input and virtual alarm

The bits represent the activation.

	Mask	Name
Bit 31	0x80000000	Virtual Nbr. 0
...		

© Copyright 2001-2021 Bosch Security Systems

Bit 17	0x00020000	Virtual Nbr. 14
Bit 16	0x00010000	Virtual Nbr. 15
Bit 15	0x8000	Alarm Nbr. 16
...		
Bit 1	0x0002	Alarm Nbr. 2
Bit 0	0x0001	Alarm Nbr. 1

video loss alarm

The bits represent the activation

	Mask	**Name**
Bit 31	0x80000000	Alarm Nbr. 32
...		
Bit 1	0x00000002	Alarm Nbr. 2
Bit 0	0x00000001	Alarm Nbr. 1

© Copyright 2001-2021 Bosch Security Systems

2.265 CONF_HD_RECORD_PROFILES_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a91	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read recording profiles for secondary recording, payload is the same as in CONF_HD_RECORD_PROFILES	
Write	p_octet	service	Set recording profiles for secondary recording (effect takes place imidiatly), payload is the same as in CONF_HD_RECORD_PROFILES	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

This payload equals the payload of the command 'CONF_HD_RECORD_PROFILES'

© Copyright 2001-2021 Bosch Security Systems

2.266 CONF_HD_RECORD_PROFILES_V2

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad0	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read recording profiles	
Write	p_octet	service	Write recording profiles	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read/Write the ten recording profiles for a camera, payload upto 560 bytes total. The profiles will be written to the device configuration. These profiles are used as default profiles for each span that is mounted by this cam for span recording. If less than 10 profiles will be written, the remaining profiles stay unchanged.

Payload Structure

Recording Profile [0] (see description)
...
Recording Profile [9] (see description)

Recording Profile

16 32

Flags 1 Byte	**stream config nr** 1 Byte				**post alarm stream config nr** 1 Byte	**encoder index** 1 Byte
post alarm profile 1 Byte	**reserved** 4 Bits	**See *0** 1 Bit	**See *1** 2 Bits	**See *2** 1 Bit	**profile name...** 32 Bytes	
profile name [...]						
profile name ...						
profile name ...					**max pre alarm time** 2 Bytes	

© Copyright 2001-2021 Bosch Security Systems

pre alarm time 2 Bytes	**post alarm time** 2 Bytes
motion alarm 4 Bytes	
alarm input and virtual alarm 4 Bytes	
video loss alarm 4 Bytes	

8 24

Flags

	Mask	Name
Bit 6	0x40	MANAGED_BY_ONVIF
Bit 5	0x20	IMMEDIATE_ALARM_BACKUP
Bit 4	0x10	ALARM_RECORDING_PRE_ALARM_BUFFER
Bit 3	0x08	ALARM_FILE_BACKUP
Bit 2	0x04	META_RECORDING_DISABLE
Bit 1	0x02	AUDIO_RECORDING_DISABLE
Bit 0	0x01	ALARM_RECORDING_PRE_ALARM_RING
	Combined Values	
	Mask	**Name**
	0x11	ALARM_RECORDING_PRE_ALARM_AUTO

stream config nr

Video preset number used for recording

Values:

no recording on this profile	0
stream configuration nr for pre alarm	1 - 2
I-frames only from stream 1 for pre alarm	3

post alarm stream config nr

Video preset number used for recording after occurence of an alarm

Values:

not set	0	uses stream config nr if set
stream configuraton nr for post alarm	1 - 2	

© Copyright 2001-2021 Bosch Security Systems

I-frames only from stream 1 for post alarm	3	

encoder index

Index of the encoder stream starting with 0, index 255 is for the backup output

post alarm profile

Values:

not set	0	no modification of the actual enc profile
encoder profile nr for post alarm configuration	1 - 8	

cont rec backup

Only for continuous buffered recording, if set the records will be backuped to the recording, its an alternativ way of configuration to using a backup account of typ record, that way no account is wasted in the configuration

Back Up Account

Backup account Number from 0 (first account) to 3(fourth account), for choosing the back up account in case activated ALARM_FILE_BACKUP flag

profile name

Profile name is a zero terminated string

max pre alarm time

Only relevant for backup recording, it needs to know the max pre alarm time, in other cases it should be set to zero

pre alarm time

Only relevant for alarm recording, if set, this time in seconds is the time the recording will be stored up to the alarm event

post alarm time

If set, this time in seconds is the time after an alarm event for that the post alarm video preset is used if set. After this time elapses, the recording will return to the standard video preset nr, if alarm recording, the recording will also stop the recording on the actual track and start a new one

motion alarm

The bits represent the activation

© Copyright 2001-2021 Bosch Security Systems

	Mask	Name
Bit 31	0x80000000	Alarm Nbr. 32
...		
Bit 1	0x00000002	Alarm Nbr. 2
Bit 0	0x00000001	Alarm Nbr. 1

alarm input and virtual alarm

The bits represent the activation.

	Mask	Name
Bit 31	0x80000000	Virtual Nbr. 0
...		
Bit 17	0x00020000	Virtual Nbr. 14
Bit 16	0x00010000	Virtual Nbr. 15
Bit 15	0x8000	Alarm Nbr. 16
...		
Bit 1	0x0002	Alarm Nbr. 2
Bit 0	0x0001	Alarm Nbr. 1

video loss alarm

The bits represent the activation

	Mask	Name
Bit 31	0x80000000	Alarm Nbr. 32
...		
Bit 1	0x00000002	Alarm Nbr. 2
Bit 0	0x00000001	Alarm Nbr. 1

© Copyright 2001-2021 Bosch Security Systems

2.267
CONF_HD_RECORD_PROFILES_V2_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad1	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get recording profiles for secondary recording, payload is the same as in CONF_HD_RECORD_PROFILES_V2	
Write	p_octet	service	Set recording profiles for secondary recording, payload is the same as in CONF_HD_RECORD_PROFILES_V2	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.268 CONF_HD_RECORD_SCHEDULE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read primary recording schedule, see detailed description	
Write	p_octet	service	Set primary recording schedule(effect takes place imidiately), see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command reads/writes the recording schedule for all cameras (primary and secondary recording). The entries are 4 bit each and represent a recording profile for a 15 min time period(payload 336 bytes total). The recording profiles with same profile number can be different for each camera. See command HD_RECORD_PROFILES. The schedule will be written to the storage medium. In case of span recording mode (see CONF_RECORD_MODE_SPANS) the schedule will be only read from or stored in the config. This schedule is used as default schedule for each span that is mounted by this device for recording.

Payload Structure

Saturday [0] 4 Bits	...	**Saturday [95]** 4 Bits
Sunday [0] 4 Bits	...	**Sunday [95]** 4 Bits
Monday [0] 4 Bits	...	**Monday [95]** 4 Bits
Tuesday [0] 4 Bits	...	**Tuesday [95]** 4 Bits
Wednesday [0]	...	**Wednesday [95]**

© Copyright 2001-2021 Bosch Security Systems

4 Bits		4 Bits
Thursday [0] 4 Bits	...	**Thursday [95]** 4 Bits
Friday [0] 4 Bits	...	**Friday [95]** 4 Bits

Saturday

96 entries of recording profile numbers, each represents the recording profile for a 15 min time period. First entry is from 00:00 to 00:15. The following entries are for the following 15 min time periods until 24:00.

recording off	0
recording profile number	1 - 10

Sunday

See 'Saturday' for details.

Monday

See 'Saturday' for details.

Tuesday

See 'Saturday' for details.

Wednesday

See 'Saturday' for details.

Thursday

See 'Saturday' for details.

Friday

See 'Saturday' for details.

© Copyright 2001-2021 Bosch Security Systems

2.269 CONF_HD_RECORD_SCHEDULE_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a49	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read secondary recording schedule, see detailed description	
Write	p_octet	service	Set secondary recording schedule(effect takes place imidiately), see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command reads/writes the secondary recording schedule for the camera specified by the num parameter. This command only works for span recording. The entries are 4 bit each and represent a recording profile for a 15 min time period(payload 336 bytes total). The recording profiles with same profile number can be different for each camera. See command HD_RECORD_PROFILES. The schedule will be only read from or stored in the config. This schedule is used as default schedule for each span that is mounted by this cam for secondary span recording.

Payload Structure

Saturday [0] 4 Bits	...	**Saturday [95]** 4 Bits
Sunday [0] 4 Bits	...	**Sunday [95]** 4 Bits
Monday [0] 4 Bits	...	**Monday [95]** 4 Bits
Tuesday [0] 4 Bits	...	**Tuesday [95]** 4 Bits
Wednesday [0]	...	**Wednesday [95]**

© Copyright 2001-2021 Bosch Security Systems

4 Bits		4 Bits
Thursday [0] 4 Bits	...	**Thursday [95]** 4 Bits
Friday [0] 4 Bits	...	**Friday [95]** 4 Bits

Saturday

96 entries of recording profile numbers, each represents the recording profile for a 15 min time period. First entry is from 00:00 to 00:15. The following entries are for the following 15 min time periods until 24:00.

recording off	0
recording profile number	1 - 10

Sunday

See 'Saturday' for details.

Monday

See 'Saturday' for details.

Tuesday

See 'Saturday' for details.

Wednesday

See 'Saturday' for details.

Thursday

See 'Saturday' for details.

Friday

See 'Saturday' for details.

© Copyright 2001-2021 Bosch Security Systems

2.270 CONF_HD_RECORDING_ACTIVE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0908	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.271 CONF_HD_RECORDING_REPORT

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a1c	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get recording report from actual primary recording on a cam,	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This comand reads the recording report of an actual running recording of a cam. The recording report contains information about RTP and VDP packets as well as VDP allocation of a running recording. The cam is specified by the num parameter.

Payload Structure

16	32
Version 1 Byte	**Reserved** 3 Bytes
Start Seconds 4 Bytes	
Start Milliseconds 4 Bytes	
VDP Allocation No Wait 4 Bytes	
VDP Allocation Wait 4 Bytes	
VDP Allocation Fail 4 Bytes	
Encoder Data... (see description)	
Encoder Data [...]	
Encoder Data ...	
Storage Data... (see description)	

© Copyright 2001-2021 Bosch Security Systems

Storage Data [...]
Storage Data ...
Network Data... (see description)
Network Data [...]
Network Data ...

8 24

Version

Version information.

Start Seconds

Timestamp in seconds since 2000 when the counting of recording data started.

Start Milliseconds

Milliseconds of the timestamp when the counting of recording data started.

VDP Allocation No Wait

Number of VDP allocation operation performed without waiting.

VDP Allocation Wait

Number of VDP allocation operation performed with waiting.

VDP Allocation Fail

Number of VDP allocation operation failed.

Encoder Data

16 32

Video RTP Packet Count 4 Bytes
Video Byte Count 4 Bytes
Reserved 4 Bytes

© Copyright 2001-2021 Bosch Security Systems

Audio RTP Packet Count 4 Bytes
Audio Byte Count 4 Bytes
Reserved 4 Bytes
Meta RTP Packet Count 4 Bytes
Meta Byte Count 4 Bytes
Reserved... 16 Bytes
Reserved ...
Reserved ...
Reserved ...

8 24

Video RTP Packet Count

Number of video RTP packets that are deliverd to the recording.

Video Byte Count

Number of video bytes that are deliverd to the recording.

Audio RTP Packet Count

Number of audio RTP packets that are delivered to the recording.

Audio Byte Count

Number of audio bytes that are deliverd to the recording.

Meta RTP Packet Count

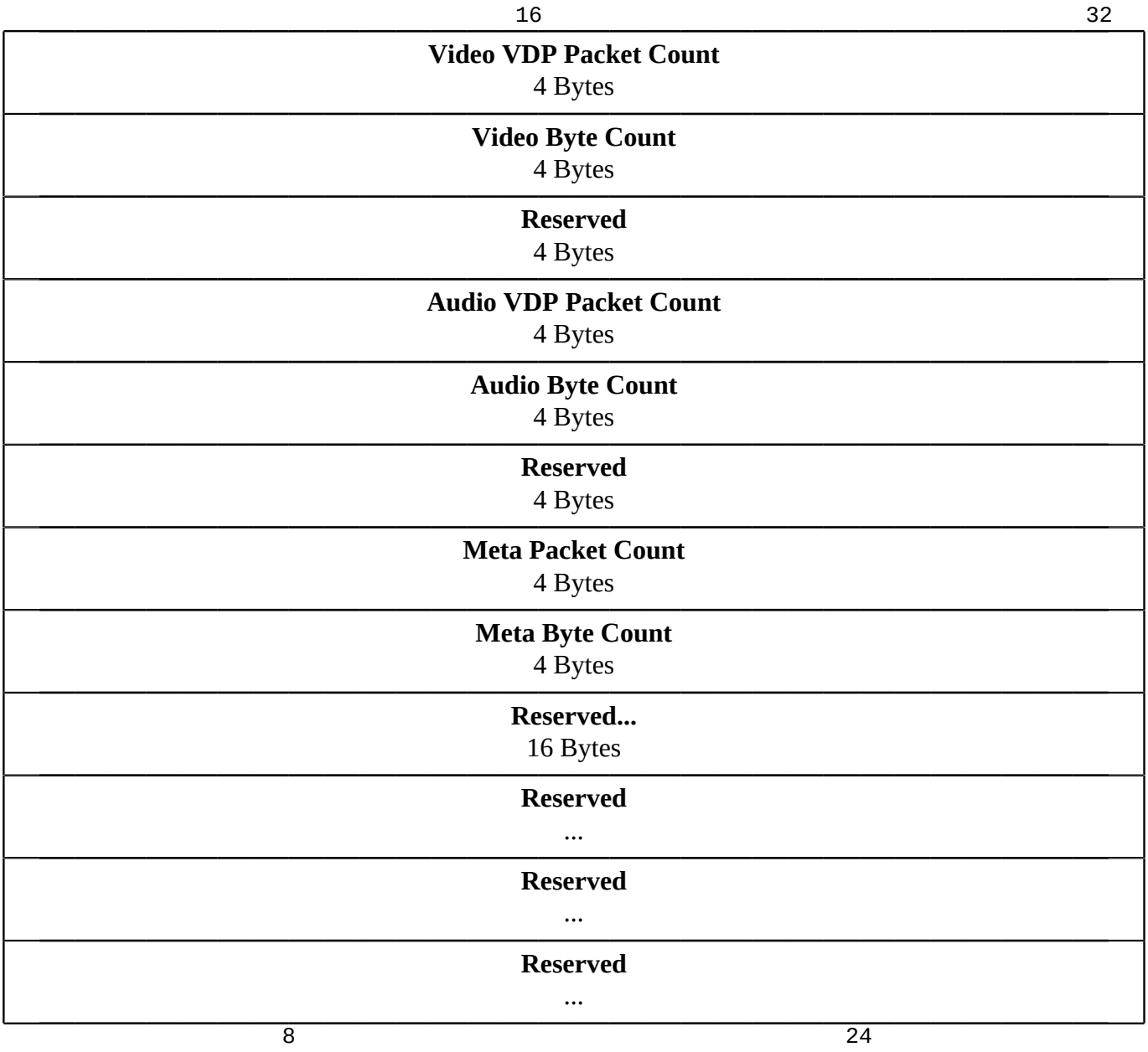
Number of meta RTP packets that are delivered to the recording.

Meta Byte Count

Number of meta bytes that are deliverd to the recording.

© Copyright 2001-2021 Bosch Security Systems

Storage Data



Video VDP Packet Count

Number of video + audio VDP packets that are written to storage (since fw 4.0 there are vdp packets containing video and audio data mixed, these packets will be counted here and not in the 'Audio VDP Packet Count' field).

Video Byte Count

Number of video + audio bytes that are written to storage (since fw 4.0 there are vdp packets containing video and audio data mixed, these packet bytes will be counted here and not in the 'Audio Byte Count' field).

© Copyright 2001-2021 Bosch Security Systems

Audio VDP Packet Count

Number of audio VDP packets that are written to storage.

Audio Byte Count

Number of audio bytes that are written to storage.

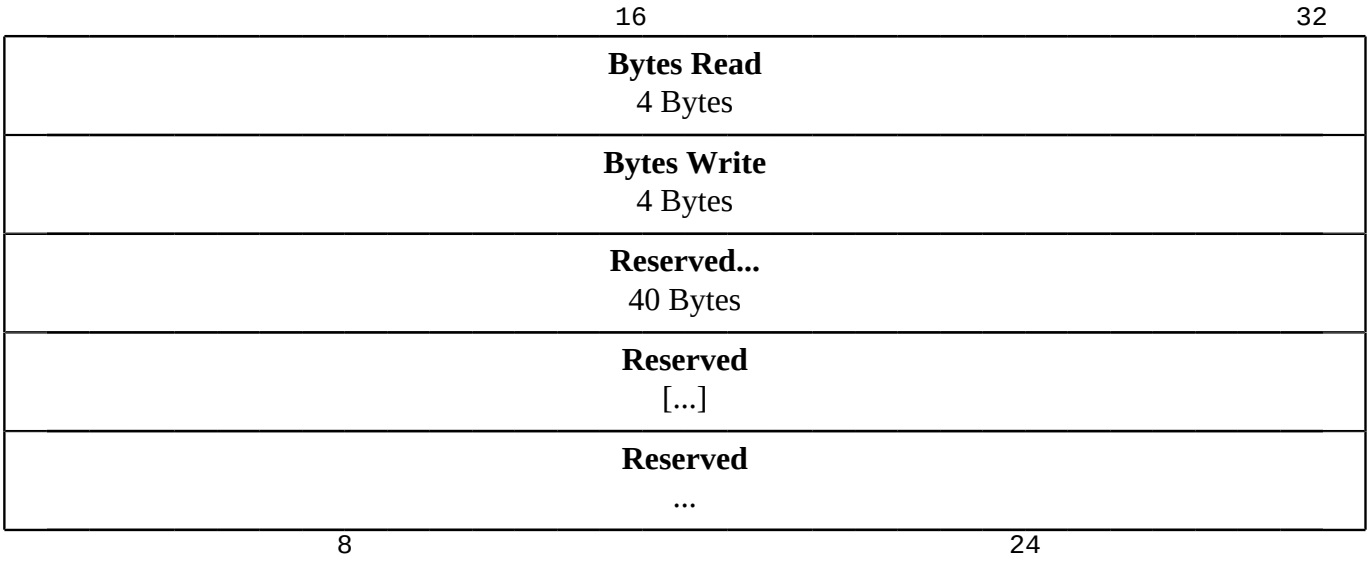
Meta Packet Count

Number of meta VDP packets that are written to storage.

Meta Byte Count

Number of meta bytes that are written to storage.

Network Data



Bytes Read

Number of Bytes that are read from storage.

Bytes Write

Number of Bytes that are written to storage.

© Copyright 2001-2021 Bosch Security Systems

2.272
CONF_HD_RECORDING_REPORT_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a4f	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get recording report from actual secondary recording on a cam, payload same as in command CONF_HD_RECORDING_REPORT	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

This payload equals the payload of the command 'CONF_HD_RECORDING_REPORT'

© Copyright 2001-2021 Bosch Security Systems

2.273 CONF_HD_RELOAD_PARTITION_FILE_INFO

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x091e	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.274 CONF_HD_REPLAY_AUTHENTICITY_OUT

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c9f	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This message will be send from a replay session, if authenticity ckeck is enabled for that session

Payload Structure

16			32
hash type 1 Byte	**hash status** 1 Byte	**signature status** 1 Byte	**certificate status** 1 Byte
start time sec 4 Bytes			
start time ms 4 Bytes			
start tz qh 1 Byte	**reserved** 1 Byte	**start rtp sq nr** 2 Bytes	
start rtp clock 4 Bytes			
last time sec 4 Bytes			
last time ms 4 Bytes			
last tz qh 1 Byte	**reserved** 1 Byte	**last rtp sq nr** 2 Bytes	
last rtp clock 4 Bytes			
8			24

hash type

Hash type for authenticity check

© Copyright 2001-2021 Bosch Security Systems

none	0
md5	1
sha1	2
sha265	3

hash status

Hash status for authenticity check

missing	0
not checked	1
invalid	2
valid	3

signature status

Signature status

missing	0
not checked	1
missing certificate	2
invalid	3
valid	4

certificate status

Signing certificate status

not checked	0
unknown	1
invalid	2
trusted	3
owned	4

start time sec

Start time in seconds since 2000 utc of the sequence

start time ms

Start time milli seconds offset

start tz qh

Start time time zone offset in quarter hours (7 lowest bits), and sign (highest bit)

© Copyright 2001-2021 Bosch Security Systems

reserved

start rtp sq nr

Rtp sequence number of the first packet in the sequence

start rtp clock

Rtp clock of the first packet in the sequence

last time sec

Slast time in seconds since 2000 utc of the sequence

last time ms

Last time milli seconds offset

last tz qh

Last time time zone offset in quarter hours (7 lowest bits), and sign (highest bit)

reserved

last rtp sq nr

Rtp sequence number of the last packet in the sequence

last rtp clock

Rtp clock of the last packet in the sequence

© Copyright 2001-2021 Bosch Security Systems

2.275 CONF_HD_REPLAY_CERTIFICATES_LIST

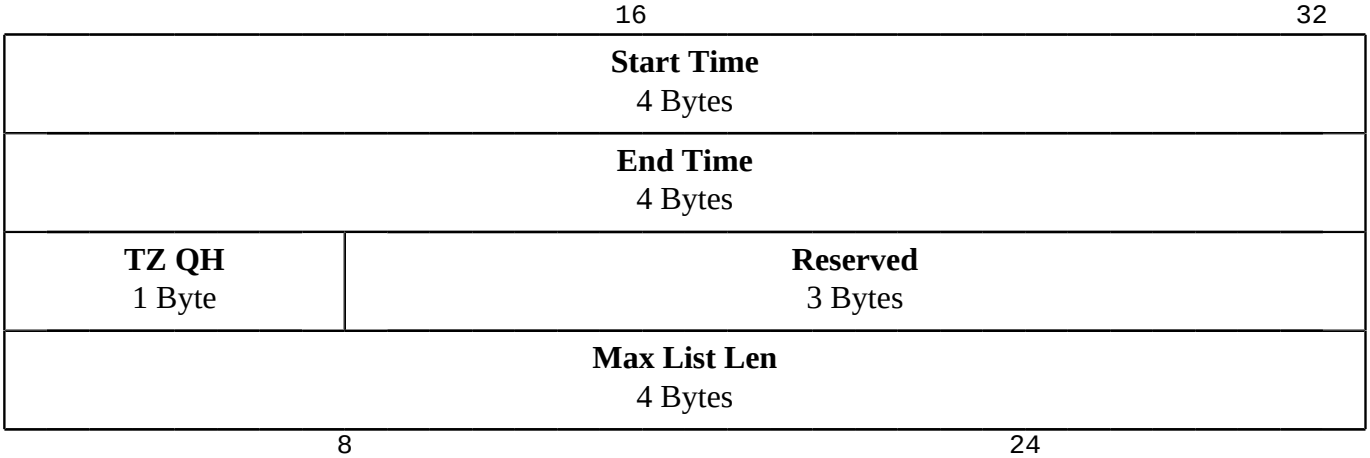
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c12	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get certificates list of records, Replay Session ID is needed, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to query the certificates from the records of a camera via replay session (session ID required), in order to verify the signed video record data within that records. The response will deliver all certificates of the records and time intervall addressed by the specidfied replay session. The maximum size of the response can be limited by the caller but it will not exceed 16 kb for the certificates list. If the response payload size isn't enough to hold all valid certificates, a flag in the response will signal the existence of further certificates, which didn't fit in the response payload. A second query in that case with a smaller time interval may help to get the remaining certificates. The response will have a certificates list with several enries of different size, there are no gaps between the entries.

Request Payload Structure



Start Time

Start time of the intervall in seconds since 2000 local time based on the Timezone offset in TZ QH

End Time

End time of the intervall in seconds since 2000 local time based on the Timezone offset in TZ QH

© Copyright 2001-2021 Bosch Security Systems

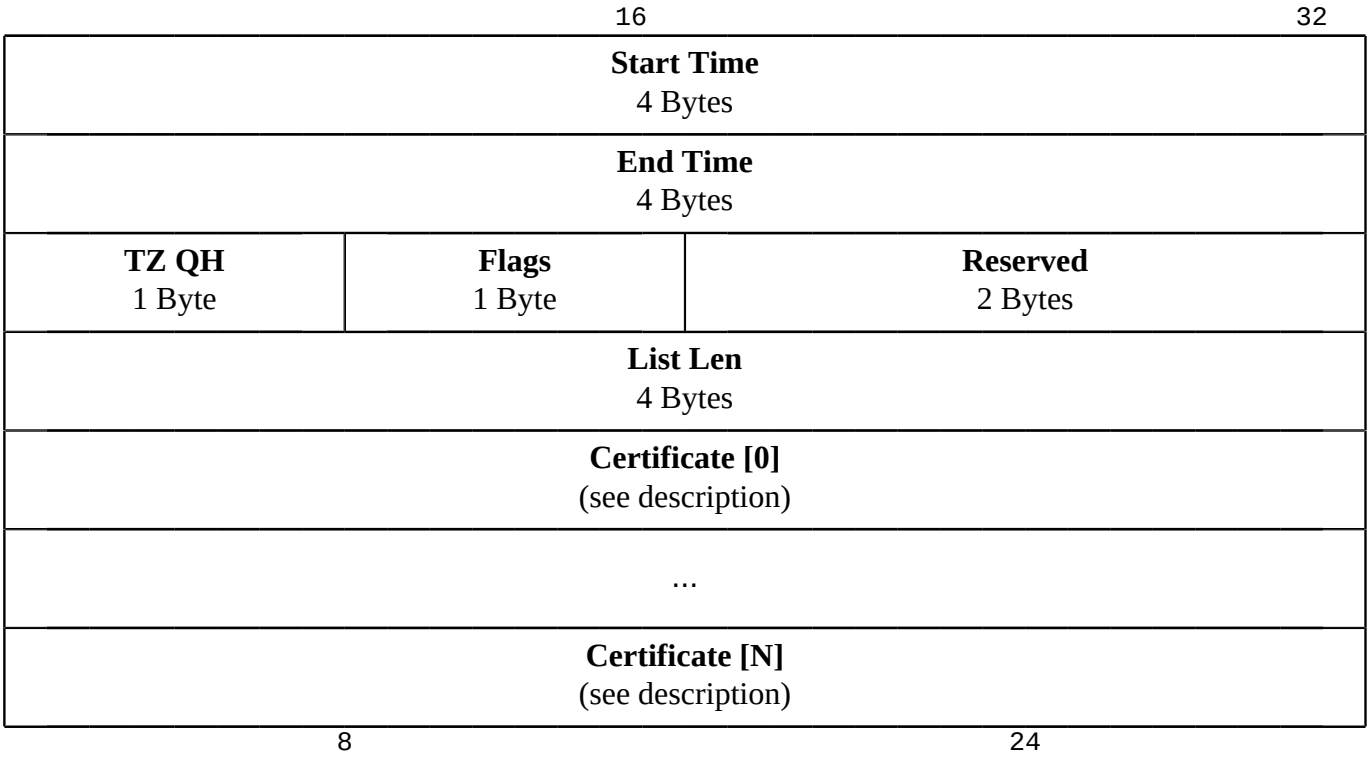
TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Max List Len

Maximum size of the certificates list in response payload in bytes

Response Payload Structure



Start Time

Start time of the intervall in seconds since 2000 local time based on the Timezone offset in TZ QH

End Time

End time of the intervall in seconds since 2000 local time based on the Timezone offset in TZ QH

TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Flags

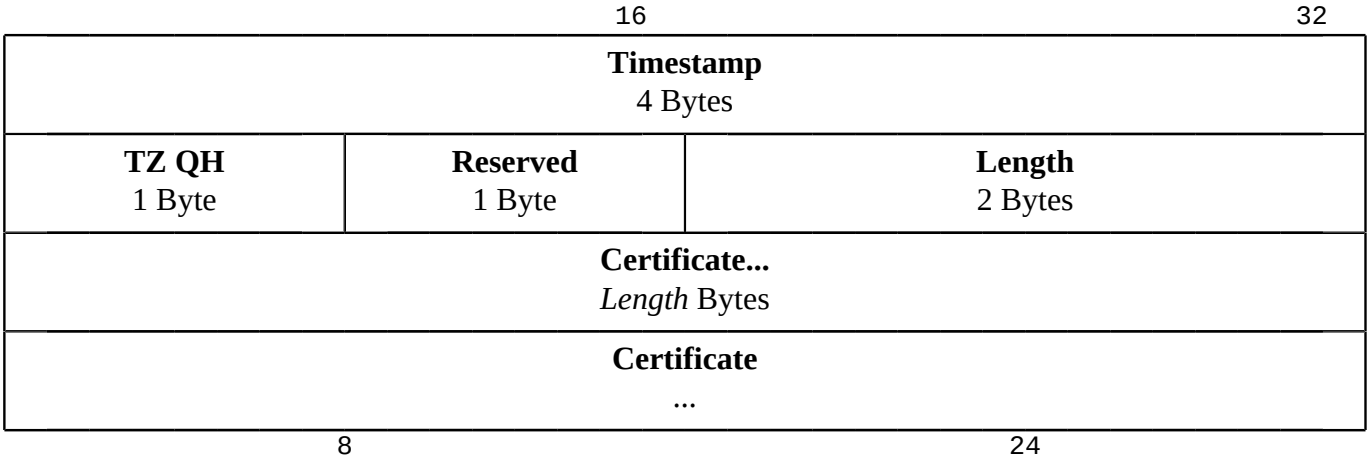
	Mask	Name	Description
Bit 0	0x01	Max Length Exceeded	More certificates available, but the max length of the certificates list was exceeded

© Copyright 2001-2021 Bosch Security Systems

List Len

Size of the certificates list in response payload in bytes

Certificate



Timestamp

Local time in seconds since 2000

TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Length

Length of the Certificates List Entry (certificate including these 8 bytes header infos)

Certificate

One certificate

© Copyright 2001-2021 Bosch Security Systems

2.276 CONF_HD_REPLAY_CUSTOM_SETTINGS

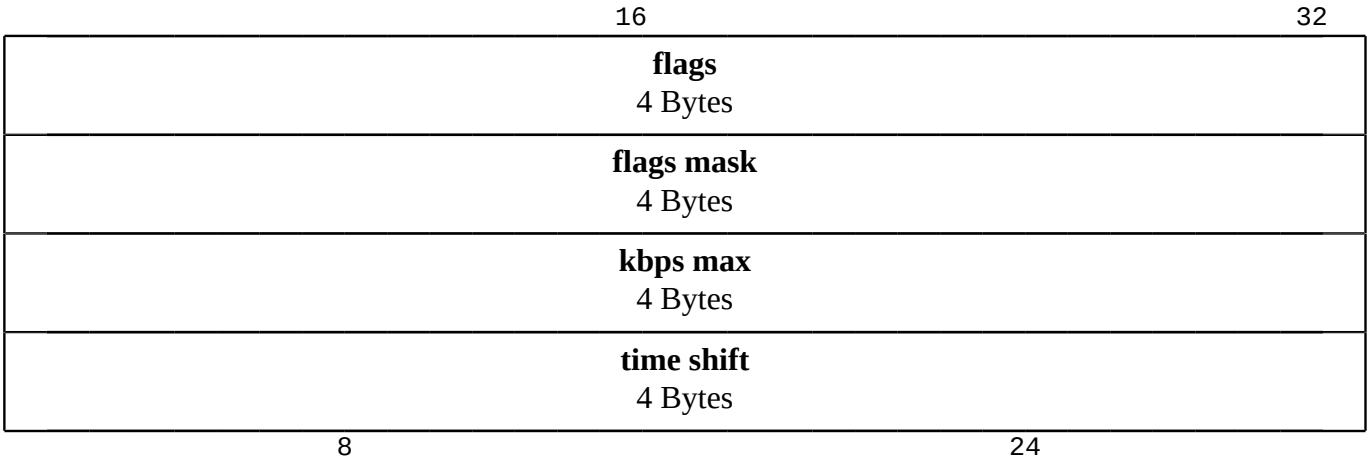
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b56	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read replay custom settings, see detailed,	
Write	p_octet	user	Set replay custom settings, see detailed,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used to configure the behaviour of an existing replay session. A valid replay session id has to be provided.

Payload Structure



flags

Flag field for configuring custom behavior of an replay session

	Mask	Name
Bit 13	0x00002000	aac 16 khz rtp clock
Bit 12	0x00001000	ignore start and end of files
Bit 11	0x00000800	transcode
Bit 10	0x00000400	set time shift
Bit 9	0x00000200	patch pic id
Bit 8	0x00000100	add rtp hdr extension
Bit 7	0x00000080	use UTC
Bit 6	0x00000040	L16 to G711
Bit 5	0x00000020	set kbps max

© Copyright 2001-2021 Bosch Security Systems

Bit 4	0x00000010	no rcp message
Bit 3	0x00000008	auto start
Bit 2	0x00000004	h264 only
Bit 1	0x00000002	dont send dummy pkt
Bit 0	0x00000001	patch rtp hdr

flags mask

Mask for the flag field, it is used to set only choosen values without changing the other values. To set a choosen value in the flag field, the corresponding flag in the flags mask has to be set also. All other setting will be left unchanged

kbps max

Set the max send data rate in kbit per seconds, 'set kbps max' flag has to be set in order to change that value.

time shift

Set time shift in seconds, 'set time shift' flag has to be set in order to change that value.

© Copyright 2001-2021 Bosch Security Systems

2.277 CONF_HD_REPLAY_ENCRYPTED_DATA

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c89	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.278 CONF_HD_REPLAY_EVENT_INFO

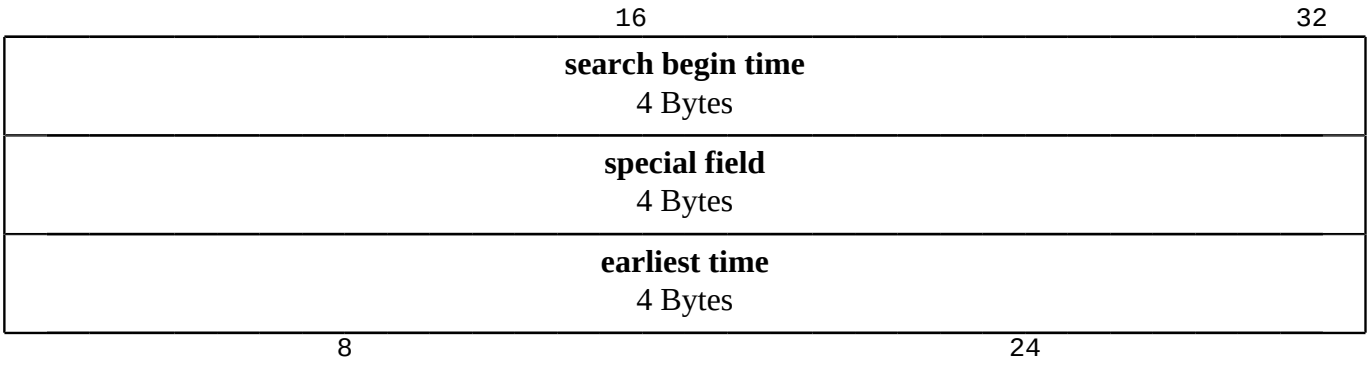
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x091f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command returns a list of recorded alarm events in backward order, since fw 4.0 events are only alarm state changes. Max size of the list is 128 entries. The command needs a valid replay session. The events start from the given end time (first 4 bytes of payload) in secconds since 2000. If the payload consists of at least 8 bytes and the bytes 5-8 are filled with the value 0xeeeeeeee the extended event structure is used in the response payload. Payload size with all optional parameters is 12 bytes (see "In Payload"). The response contains a flag field, containing the alarm flags, which give information about the alarm, which are activated at this event or if all alarms went to off state, if no one of the alarm flags were set. the flags also contains the time zone information with sign and number of quarter hours.

Request Payload Structure



search begin time

Start time of the search in seconds since 2000. The search will procceed into the past.

special field

Optional parameter, if set to 0xeeeeeeee, the reply will contain data in extended event structure format. Any other value has no meaning yet.

© Copyright 2001-2021 Bosch Security Systems

earliest time

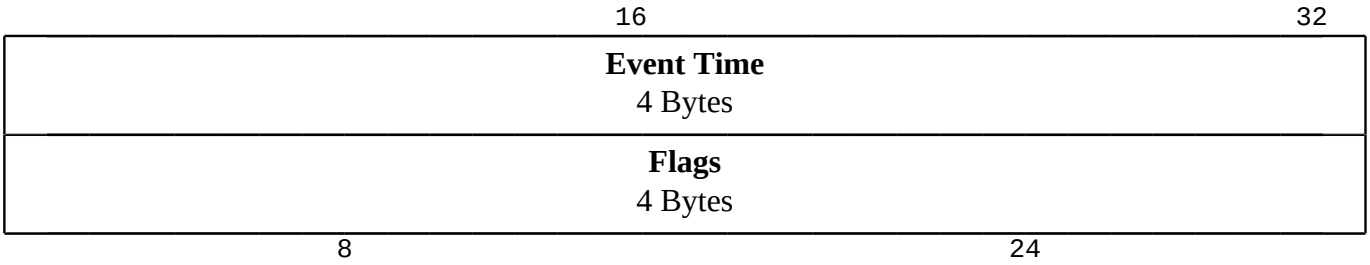
Optional parameter, Erliest time in seconds since 2000. The search won't search in record files, which lie before that time. The search will be performed on file which lie completely or partialy within the search interval of earliest time and search begin time. If not set, the default will be 0.

Response Payload Structure

Event [0] (see description)
...
Event [N] (see description)

Event

'Event' payload for '??' = 'Simple Event' (??)



Event Time

Seconds since 2000.

Flags

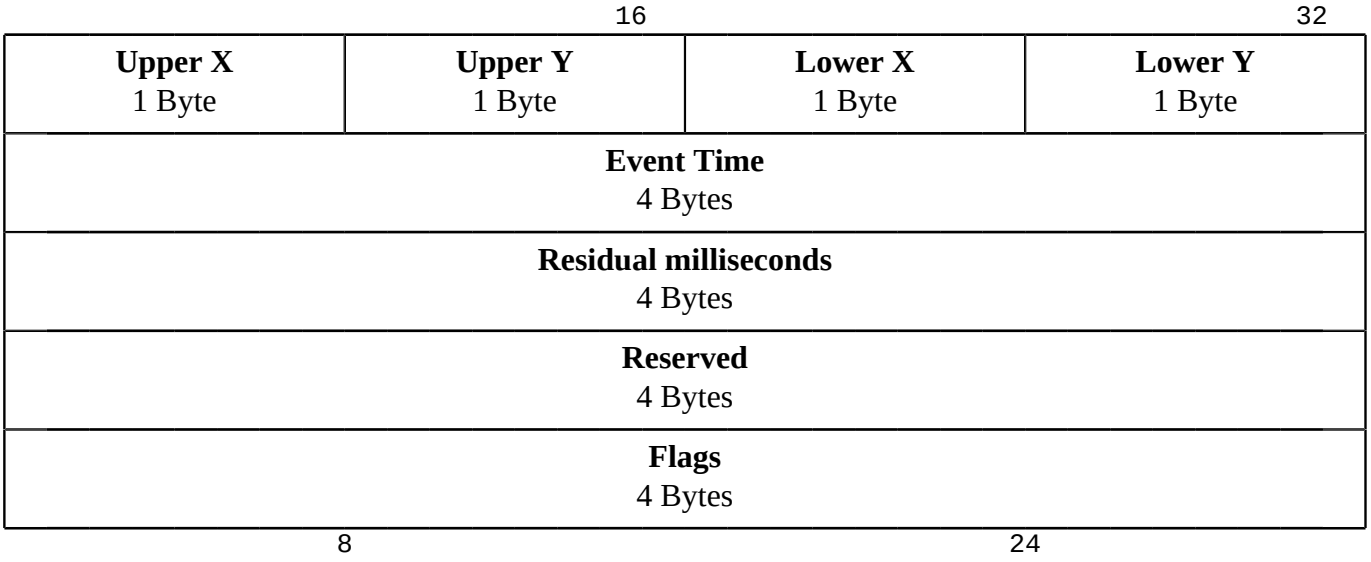
	Mask	Name	Description
Bit 31	0x80000000	time zone quarter hours sign	1 - negativ
Bit 30	0x40000000	time zone quarter hours	
Bit 29	0x20000000	time zone quarter hours	
Bit 28	0x10000000	time zone quarter hours	
Bit 27	0x08000000	time zone quarter hours	
Bit 26	0x04000000	time zone quarter hours	
Bit 25	0x02000000	time zone quarter hours	
Bit 24	0x01000000	time zone quarter hours	
Bit 3	0x00000008	Virtual alarm	(see CONF_VIRTUAL_ALARM_STATE)
Bit 2	0x00000004	Video Loss	
Bit 1	0x00000002	Input Alarm	
Bit 0	0x00000001	Motion Alarm	

© Copyright 2001-2021 Bosch Security Systems

Combined Values

Mask	Name	Description
0x7f000000	time zone quarter hours	

'Event' payload for '??' = 'Extended Event' (??)



Upper X

Specifies a bounding box (upper left corner and lower right corner). If an object can be associated with the alarm then the bounding box will be around this object otherwise the whole image is the bounding box. The values are normalized to the image size and are between 0 and 255. The origin of the coordinate system is the upper left corner of the image.

Upper Y

See 'Upper X' for details.

Lower X

See 'Upper X' for details.

Lower Y

See 'Upper X' for details.

Event Time

Seconds since 2000.

© Copyright 2001-2021 Bosch Security Systems

Flags

	Mask	Name	Description
Bit 31	0x80000000	time zone quarter hours sign	1 - negativ
Bit 30	0x40000000	time zone quarter hours	
Bit 29	0x20000000	time zone quarter hours	
Bit 28	0x10000000	time zone quarter hours	
Bit 27	0x08000000	time zone quarter hours	
Bit 26	0x04000000	time zone quarter hours	
Bit 25	0x02000000	time zone quarter hours	
Bit 24	0x01000000	time zone quarter hours	
Bit 3	0x00000008	Virtual alarm	(see CONF_VIRTUAL_ALARM_STATE)
Bit 2	0x00000004	Video Loss	
Bit 1	0x00000002	Input Alarm	
Bit 0	0x00000001	Motion Alarm	

Combined Values

Mask	Name	Description
0x7f000000	time zone quarter hours	

© Copyright 2001-2021 Bosch Security Systems

2.279 CONF_HD_REPLAY_FAST_INTRA_DELAY

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x095e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the delay (in 10ms) between the frames in intra only replay mode	
Write	t_dword	user	Set the delay (in 10ms) between the frames in intra only replay mode	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.280 CONF_HD_REPLAY_FAST_INTRA_FPS

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ac2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get max frames per second for intra only replay mode	
Write	t_dword	user	Set max frames per second for intra only replay mode (0 = default)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.281
CONF_HD_REPLAY_FORENSIC_SEARCH_CANCEL

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b50	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	user	Cancel a forensic search, t_dword is the search id (result of search setup), session id is required	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.282
CONF_HD_REPLAY_FORENSIC_SEARCH_RESULT

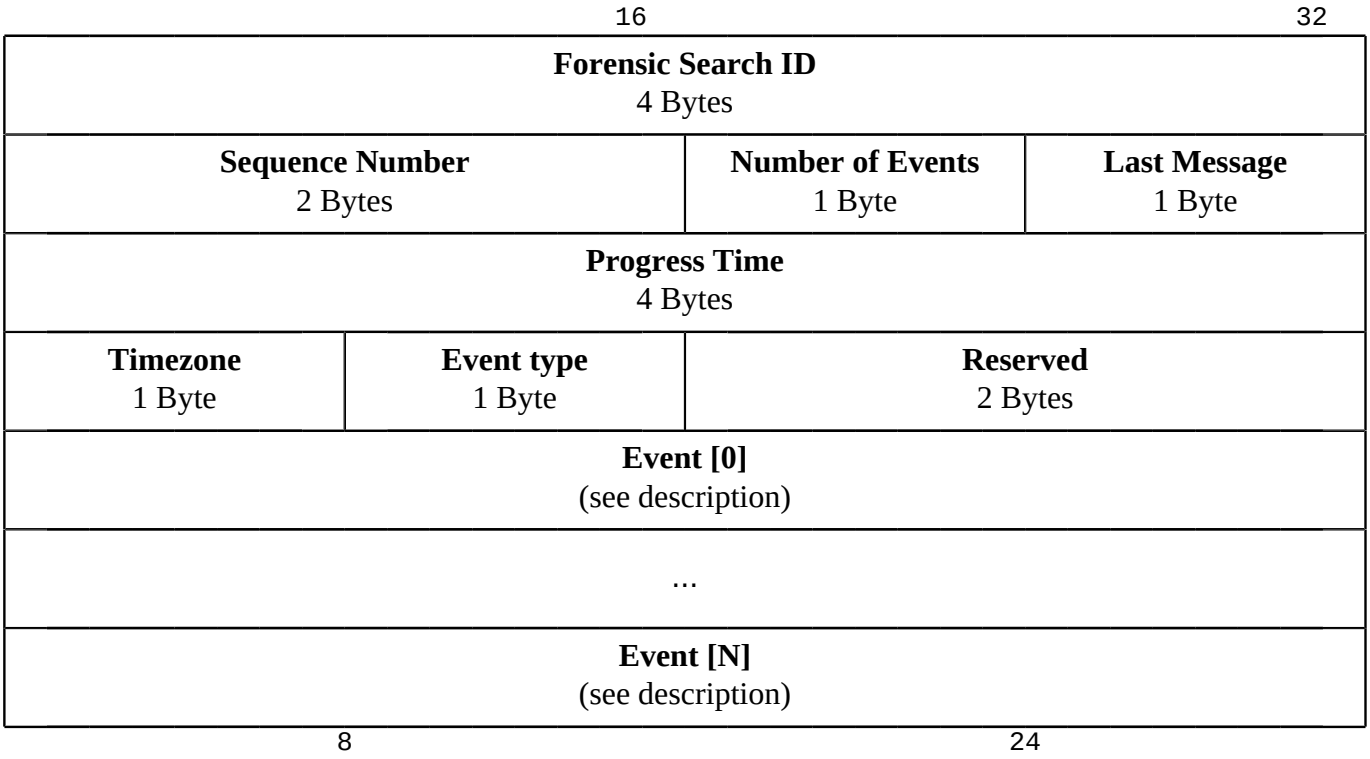
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b0b	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is a messages command. It successively delivers the results of a forensic search request.

Payload Structure



Forensic Search ID

With the forensic search ID this event message can be assigned to the corresponding search request.

Sequence Number

The sequence number is increasing and missing packets can be detected.

© Copyright 2001-2021 Bosch Security Systems

Last Message

The last message byte is signalling that the search has been accomplished with this message.

Progress Time

This time informs about the progress of the search. The time is given in seconds since 1-1-2000 (local time). This time is updated every second (search time) and forces to throw this message. Hence, it can be expected that at least every second a message will be thrown.

Timezone

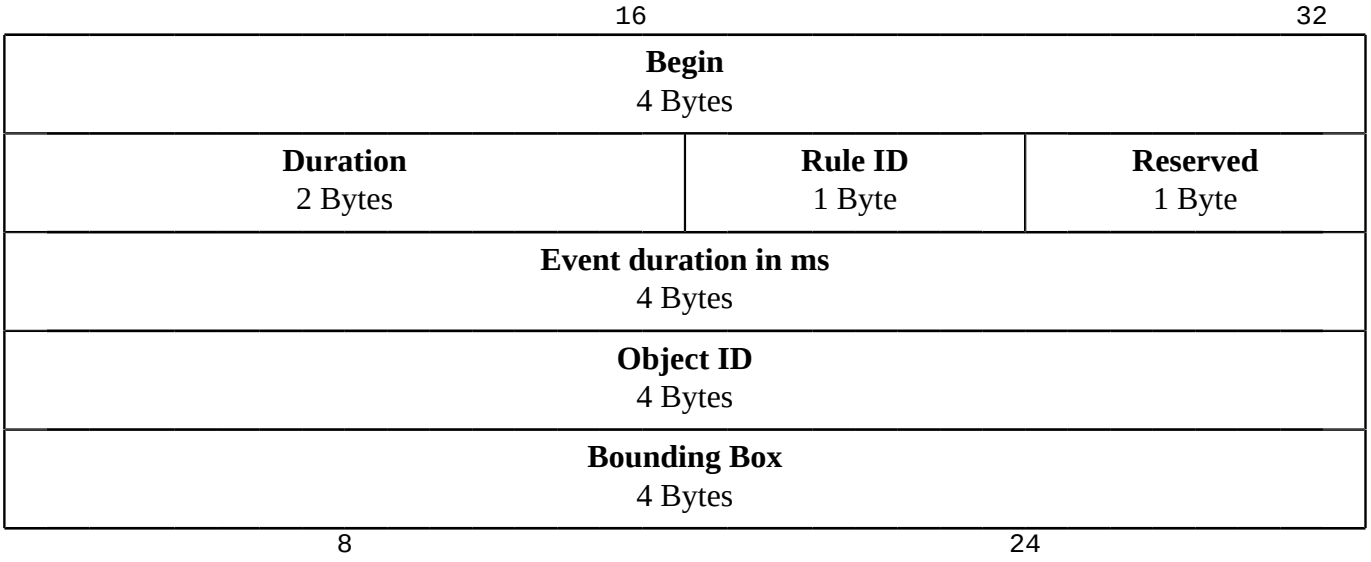
Timezone in quarter hours (signed byte in two's complement representation).

Event type

Object Event	0
Error Event	1

Event

'Event' payload for 'Event type' = 'Object Event' (0)



Begin

The beginning of the event is provided in local time in seconds since 2000 and milliseconds.

Duration

Duration of the event in ms.

© Copyright 2001-2021 Bosch Security Systems

Rule ID

The Rule ID informs which rule is related to this event.

Object ID

Object ID which has caused the alarm.

Bounding Box

Bounding box of object in normalized coordinates from 0 to 255. First, the upper left point is encoded then the lower right point of the bounding box. The coordinate (0,0) is the upper left corner of the image and the point (255,255) is the lower right corner of teh image. The bounding box is from the begin time.

'Event' payload for 'Event type' = 'Error Event' (1)

Error type 1 Byte	**Error text...** 64 Bytes
Error text ...	
Error text ...	
Error text ...	

Error type

Values:

e_ForensicSearchErrorNone	0x00
e_ForensicSearchErrorInternal	0x02
e_ForensicSearchErrorNoMetaData	0x07
e_ForensicSearchErrorDataDoesNotMatch	0x08

Error text

The error text is a detailed description of the error. E.g. for syntax errors the line of the script and error type are mentioned.

© Copyright 2001-2021 Bosch Security Systems

2.283
CONF_HD_REPLAY_FORENSIC_SEARCH_SETUP

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b0a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user	Setup a forensic search, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is related to a replay session. The replay session will start a forensic search with the provided ruleengine script.

In the successful mode the return payload consists of an unique 4 byte ForensicSearch ID which can be refound in the corresponding message (HD_REPLAY_FORENSIC_SEARCH_RESULT). In the case of an error the RCP_ERROR_COMMAND_SPECIFIC error is returned with an error code (see HD_REPLAY_FORENSIC_SEARCH_RESULT).

Request Payload Structure

<table>
<tr><td colspan="4">16 32</td></tr>
<tr><td colspan="4">Start Time
4 Bytes</td></tr>
<tr><td colspan="4">End Time
4 Bytes</td></tr>
<tr><td colspan="4">Reserved
4 Bytes</td></tr>
<tr><td colspan="2">Script Length
2 Bytes</td><td>Script Mode
1 Byte</td><td>VCA Profile ID
1 Byte</td></tr>
<tr><td>Layer Mode
1 Byte</td><td>Flags
1 Byte</td><td colspan="2">Layer Mask
2 Bytes</td></tr>
<tr><td colspan="4">Script Data...
(see description)</td></tr>
<tr><td colspan="4">Script Data
...</td></tr>
<tr><td colspan="4">8 24</td></tr>
</table>

© Copyright 2001-2021 Bosch Security Systems

Start Time

Start time of forensic search in seconds since 2000 (local time).

End Time

End time of forensic search in seconds since 2000 (local time).

Script Length

Length of the script. If the length of the script is zero a default search is performed, i.e. 'Detect any object or Detect any Flow or Detect any Motion'.

Script Mode

The mode defines how the configuration data has to interpreted. See detailed description of configuration data for each case below.

Rule Engine Configuration	0x00
Rule Engine Script Only	0x01
In Field Search	0x02
Crossing Line Search	0x03

VCA Profile ID

VCA profile id search is supported. The recommended profile id is zero resulting in an exhausting search over the given interval. With a non zero profile id the seacrh speed can be increased because meta data recorded with different a profile are skipped.

Layer Mode

Flags

	Mask	Name	Description
Bit 3	0x08	Flag Clear Cache	Collected VCD packets will be deleted, next search will start new replay
Bit 2	0x04	Flag Cache Data	Collects compressed VCD data, next search will use collected VCD packets
Bit 1	0x02	Flag Verbose	Additional printouts
Bit 0	0x01	Flag Send	Send layer data

Layer Mask

The layer mask is only used if layer mode is set to 1.

Script Data

© Copyright 2001-2021 Bosch Security Systems

'Script Data' payload for 'Script Mode' = 'Rule Engine Configuration' (0x00)

In this mode the configuration data contains the whole rule engine configuration (eg including camera calibration). For more detailed information see the rcp command: CONF_VCD_OPERATOR_PARAMS.

'Script Data' payload for 'Script Mode' = 'Rule Engine Script Only' (0x01)

If the script mode is selected to "rule engine script only" the configuration data contains the rule engine script. The script is encoded in plain text. One of these examples can be used as a valid script.

Example: Any motion / flow / object in Field

```
Field #1 := { Point(50, 50) Point(100, 50) Point(100, 100) Point(50,
 100) };
FlowDetector #1 := { Field #1 };
MotionDetector #1 := { Field #1 };
external SimpleState #1 := DetectedFlow #1;
external SimpleState #2 := DetectedMotion #1;
external ObjectState #3 := InsideField #1;
```

Example: Object in Field

```
Field #1 := { Point(50, 50) Point(100, 50) Point(100, 100) Point(50,
 100) };
external ObjectState #1 := InsideField #1;
external Event #2 := OnSet ObjectState #1;
```

Example: Crossing Line

```
Resolution := { Min(-1,-1) Max(1,1) };
Line #1 := { Point(-0.5, -0.5) Point(0.5, 0.5) DebounceTime(0.50)
 Direction(1) };
external Event#2 := { CrossedLine #1 };
```

Example: Following Route

```
Route #2 := { Point(41, 117) Distance(5) Point(51, 87) Distance(9)
 Point(79, 74) Distance(5) Direction(1) MinPercentage(80)
 MaxGap(10) };
external Event #3 := { FollowedRoute #2 };
```

For more details, read the Bosch Query Language Specification.

© Copyright 2001-2021 Bosch Security Systems

'Script Data' payload for 'Script Mode' = 'In Field Search' (0x02)

16	32
Upper left x-coordinate 2 Bytes	**Upper left y-coordinate** 2 Bytes
Lower right x-coordinate 2 Bytes	**Lower right y-coordinate** 2 Bytes
Debounce time 2 Bytes	**Field flags** 2 Bytes
8	24

A field search is define by two normalized points. Each point is normalized between 0 and 65535. The upper left corner of the image has the coordinates (0,0) and the lower right corner (65535,65535). Additionally, debounce time and intersection mode can be addressed.

Debounce time

The debounce time is given in units of 10 ms.

Field flags

	Mask	Name	Description
Bit 1	0x0002	Field mode	BaryCenter (0, default) or Boundingbox (1)
Bit 0	0x0001	Object mode	Intersection (0, default) or Covering (1)

'Script Data' payload for 'Script Mode' = 'Crossing Line Search' (0x03)

16	32	
Upper left x-coordinate 2 Bytes	**Upper left y-coordinate** 2 Bytes	
Lower right x-coordinate 2 Bytes	**Lower right y-coordinate** 2 Bytes	
Debounce time 2 Bytes	**Direction** 1 Byte	**Reserved** 1 Byte
8	24	

A line crossing search is define by two normalized points. Each point is normalized between 0 and 65535. The upper left corner of the image has the coordinates (0,0) and the lower right corner (65535,65535).

Debounce time

The debounce time is given in units of 10 ms.

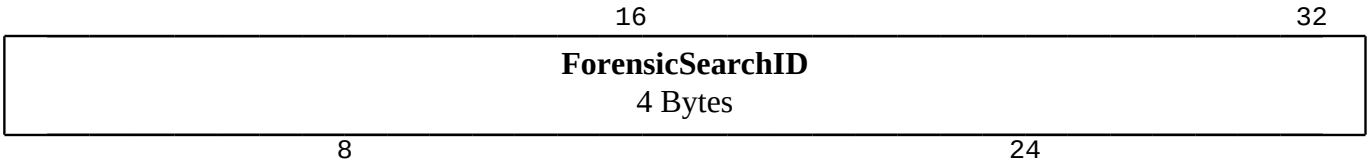
© Copyright 2001-2021 Bosch Security Systems

Direction

One can choose whether any object which passes the line triggers an event or whether only objects which pass from left to right respectively right to left are relevant:

Any Direction	0	Any object which passes the line triggers an event
Left To Right	1	Only objects which pass from left to right triggers an event
Right To Left	2	Only objects which pass from right to left triggers an event

Response Payload Structure



Command Specific Errors

e_ForensicSearchErrorNone	0x00
e_ForensicSearchErrorScriptTooLong	0x01
e_ForensicSearchErrorInternal	0x02
e_ForensicSearchErrorNoRuleSelected	0x03
e_ForensicSearchErrorREConfiguration	0x04
e_ForensicSearchErrorSyntax	0x05
e_ForensicSearchErrorMemory	0x06

© Copyright 2001-2021 Bosch Security Systems

2.284 CONF_HD_REPLAY_LIVE

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0963	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get replay connection live mode	
Write	t_dword	user	Set a replay connection to live mode (only supported with transcoder) 1:=on, 0:= off replay commands like start or seek terminate the live mode	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.285 CONF_HD_REPLAY_MOTION_SAMPLES

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x095d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Obsolete	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.286 CONF_HD_REPLAY_PREFETCH_JPEGS

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b55	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user	SessionID required, 16bytes header (1byte jpeg quality 0-100, 1byte reserved, 2bytes jpeg height in pixel, 12bytes reserved), N times seconds_since_2000 (each 4bytes)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.287 CONF_HD_REPLAY_SEEK_IFRAME

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0907	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_int	user	Set the replay pointer to the beginning of the next frame/last I-Frame(s); parameter t_int -1 back to last iframe, 1 to next frame; Session ID is needed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.288 CONF_HD_REPLAY_SEEK_TIME

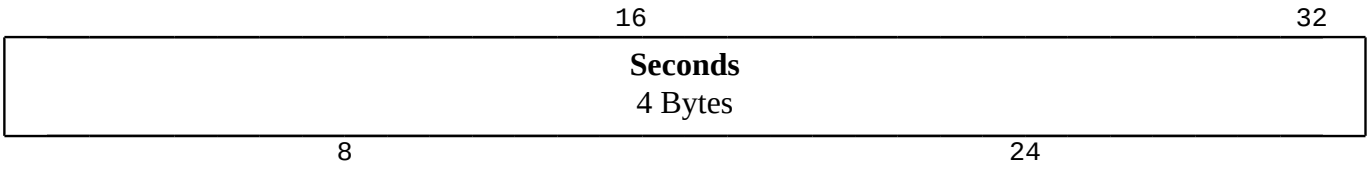
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0905	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	user		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command will return a write error if the timeposition is outside a recording set. The Session ID is needed.

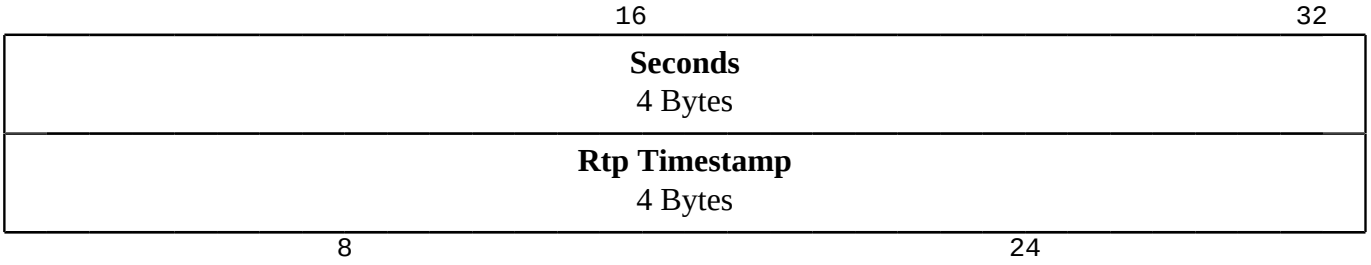
Response Payload Structure



Seconds

Absolute time in seconds since 1.1.2000 00:00h.

Message Payload Structure



Seconds

Absolute time in seconds since 1.1.2000 00:00h.

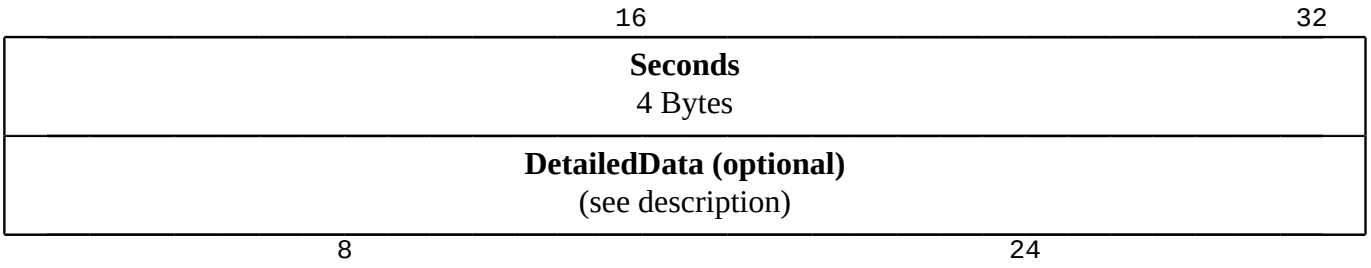
Rtp Timestamp

Rtp timestamp of the first replayed RTP packet of this second.

Write Payload Structure

© Copyright 2001-2021 Bosch Security Systems

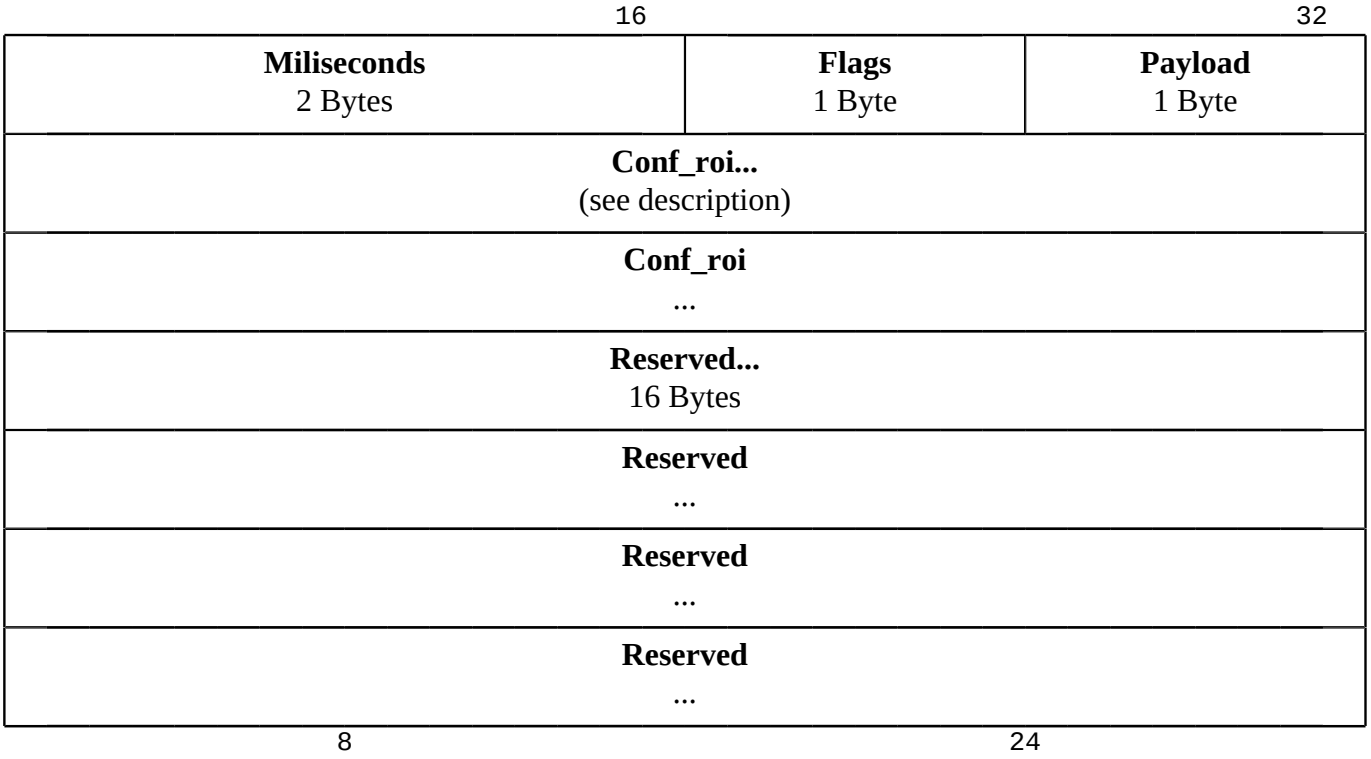
Optional, instead of above payload structure



Seconds

Absolute time in seconds since 1.1.2000 00:00h.

DetailedData (optional)



Flags

	Mask	Name	Description
Bit 2	0x04	Sec Accuracy	seek time accuracy to seconds (transcoder case only)
Bit 1	0x02	Iframe Preview	sends the Iframe on the seek position (transcoder case only)
Bit 0	0x01	Time Zone	payload contains the timezone as quarter hours offset (signed char)

© Copyright 2001-2021 Bosch Security Systems

Payload

One payload byte, conntend depends on the flags field

Conf_roi

Select region of interest hPos,vPos,hSize,vSize (each entry 2 bytes): starting left upper edge, each 2bytes 0..32768, vSize==0 means keep aspect ratio

16	32
hPos 2 Bytes	**vPos** 2 Bytes
hSize 2 Bytes	**vSize** 2 Bytes
8	24

© Copyright 2001-2021 Bosch Security Systems

2.289 CONF_HD_REPLAY_SEQUENCE_VERIFY

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c23	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Verfies authenticity of the replayed video records if enabled, sets the enabled verification flags: 0x1 - hash verify, 0x2 - signature verify, 0x4 - certificate verify	
Write	t_dword	user	Verfies authenticity of the replayed video records if enabled, sets the enabled verification flags: 0x1 - hash verify, 0x2 - signature verify, 0x4 - certificate verify	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.290 CONF_HD_REPLAY_SIZE_INFO

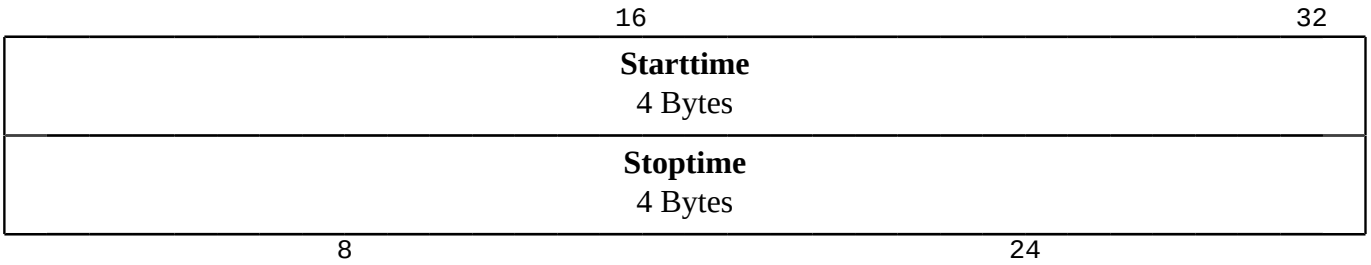
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0906	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Replay session id required. Start time and stop time has to be within one file.

Request Payload Structure



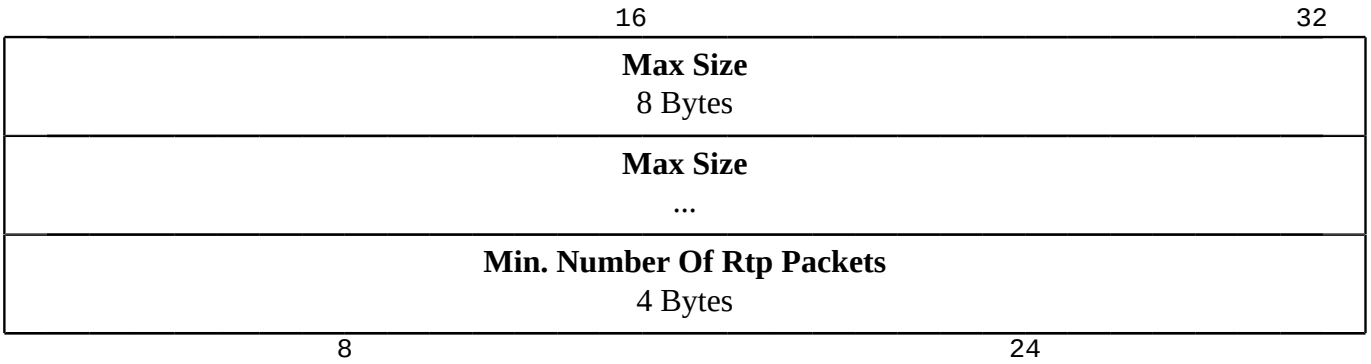
Starttime

In seconds since 2000.

Stoptime

In seconds since 2000.

Response Payload Structure



© Copyright 2001-2021 Bosch Security Systems

Max Size

Max size in bytes on storage, including recording overhead, replay stream size will be less.

Min. Number Of Rtp Packets

Minimum number of rtp packets Will tend to be more

© Copyright 2001-2021 Bosch Security Systems

2.291 CONF_HD_REPLAY_START

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0902	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Returns t_int parameter in percent of realtime replay (default +100%); 0 if suspended or stopped; Session ID is needed	
Write	t_int	user	Start a HD replay at the current position; t_int parameter in percent of realtime replay (default +100%); negative values will result in a reverse replay; Session ID is needed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.292 CONF_HD_REPLAY_START_EX

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c74	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	user	Start a HD replay at the current position; 32 bytes: 4Bytes int parameter in percent of realtime replay (default +100%); negative values will result in a reverse replay; 1byte flags, 1 byte cseq, remaining reserved, Session ID is needed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.293 CONF_HD_REPLAY_STOP

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0903	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	1=replay is stopped; 0=replay is in progress	
Write	f_flag	user	Stop a current HD replay; replay pointer will not be affected; Session ID is needed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.294 CONF_HD_REPLAY_STOP_TIME

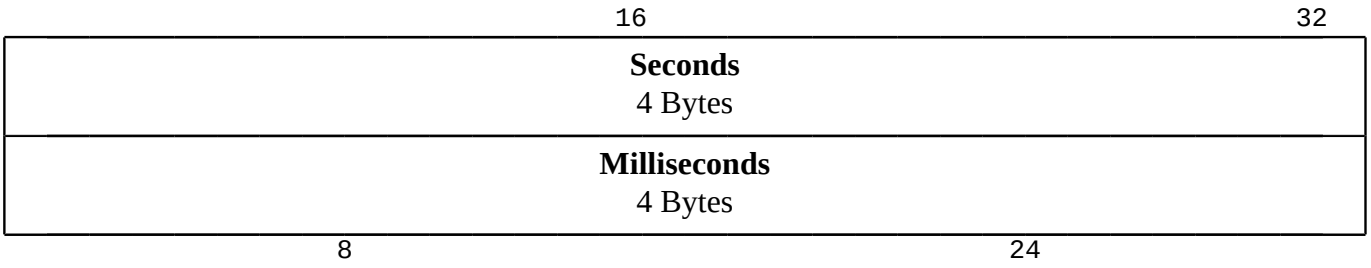
[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0904	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	user		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command will return a write error if the timeposition is outside a recording set. The Session ID is needed. A value of zero clears the stop marker.

Payload Structure



Seconds

Absolute time in seconds since 1.1.2000 00:00h.

© Copyright 2001-2021 Bosch Security Systems

2.295 CONF_HD_REPLAY_VCD_CONFIG_ID

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a5f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Requested vcd config 0=all data, 1=config1 data, 2=config2 data, ..; replay session id required	
Write	t_dword	user	Requested vcd config 0=all data, 1=config1 data, 2=config2 data, ..; replay session id required	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.296 CONF_HD_REPLAY_VCD_LAYER

[API: replay control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09c8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Requested vcd layer 0=no vcd data, 1=layer1, 2=layer2, ..; replay session id required	
Write	t_dword	user	Requested vcd layer 0=no vcd data, 1=layer1, 2=layer2, ..; replay session id required	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

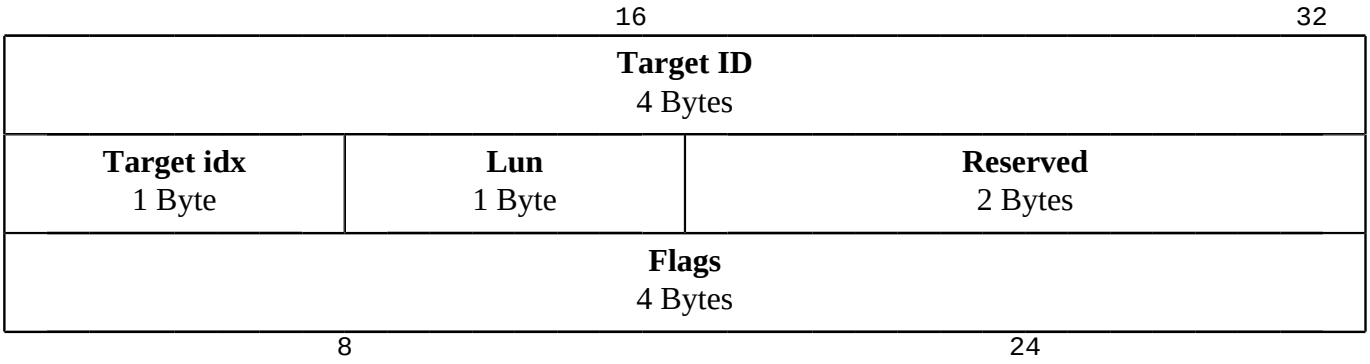
© Copyright 2001-2021 Bosch Security Systems

2.297 CONF_HD_SET_VRM_LOCK

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a5d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Write or clear the vrm lock.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Target ID

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

Target idx

Target index of the storage device

Lun

Lun of the storeage device

Flags

	Mask	Name
Bit 0	0x00000001	SET_VRM_LOCK

© Copyright 2001-2021 Bosch Security Systems

2.298 CONF_HD_SIZE_MB

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x090c	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the total size of a local storage in megabytes	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

usb	5
iof	8
cf	9
sd	11
span files	12
sd2	18

© Copyright 2001-2021 Bosch Security Systems

2.299
CONF_HDD_RECORD_ALARM_RING_INIT_SIZE

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cdf	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Initial alarm ring size on storage in MB (min 1 MB, max 128 MB, default 16 MB)	
Write	t_dword	service	Initial alarm ring size on storage in MB (min 1 MB, max 128 MB, default 16 MB)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.300 CONF_HDD_RECORD_ENCRYPTION

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c80	record index 1 (primary) or 2 (secondary)	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Hdd record encryption (span encryption only) : bit 0 -> 1ine 1; bit 1 -> line 2; .... bit n - 1 -> line n	
Write	t_dword	service	Enable the hdd encryption (span encryption only) : bit 0 -> 1ine 1; bit 1 -> line 2; .... bit n - 1 -> line n (0 - disabled (default streaming gateway), 1 - enabled (standard default)), additional a valid certificate for usage record encryption required in order to have a working record encryption	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.301 CONF_HDD_VCD_CACHE_SIZE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b79	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to configure the size of the vcd cache on a span for recording. The size parameters are rellativ to 1 GB span size. If recording is running on soans with diffrent sizes than 1 GB, the values will be internally scaled to the propper size, so it will also work for diffrent span sizes. The usage of a vcd cache on spans will cause space overhead on a storage, so it is possible to switch it of by setting both values to 0. If enabled, the recording will use these values to adjust the size of the vcd cache on spans. The used adjusted sizes will be than limited by these two values, upper limit and lower limit. The adjusment starting value will be between these to values. While recording, the uses space for cached vcd data will be measured and after each span switch, the size will be adjusted by using the measurement values for new mounted spans in order to avoid overhead. It is also possible to set a fix size for the vcd cache by this command by setting the upper and lower limit to equal values. The default value for the command is 1 for lower limit and 128 for upper limit. For read direction an input payload is needed same as the described payload but only line and rec idx are required, the remaining part of the payload can be clipped away.

Payload Structure

16	32
line 2 Bytes	**rec idx** 2 Bytes
lower size limit 2 Bytes	**upper size limit** 2 Bytes
8	24

line

Line from 1 to n.

rec idx

Rec index: 1 - primary recording, 2 - secondary recording.

© Copyright 2001-2021 Bosch Security Systems

lower size limit

Lower size limit of used vcd cache buffer on recording spans in 64 kb units per 1 GB span. Allowed values from 1 to 4096, 0 to disable cache.

upper size limit

Upper size limit of used vcd cache buffer on recording spans in 64 kb units per 1 GB span. Allowed values from 1 to 4096, 0 to disable cache.

© Copyright 2001-2021 Bosch Security Systems

2.302 CONF_HEATER_MODE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b0c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Read the heater mode	
Write	t_octet	user	Write the heater mode	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

off	0
on	1
auto	2

© Copyright 2001-2021 Bosch Security Systems

2.303 CONF_HOME_EVENT

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d05	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Event from home app in a tagged format	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Number of tagged elements 4 Bytes
Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

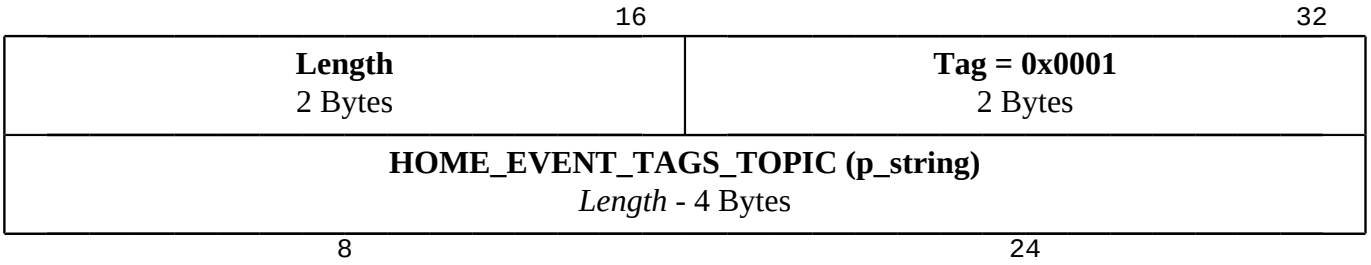
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: HOME_EVENT_TAGS_TOPIC

onvif topic



© Copyright 2001-2021 Bosch Security Systems

2.304 CONF_HOST_NAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cda	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Read the host name	
Write	p_unicode	service	Set host name (max 128 unicode character)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.305 CONF_HSTS_ENABLED

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c07	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the HSTS state (0=OFF, 1=ON 2=on + Http port redirect to HTTPS when HTTP port is also set to 0	
Write	t_dword	service	Set the HSTS state (0=OFF, 1=ON 2=on + Http port redirect to HTTPS when HTTP port is also set to 0	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.306 CONF_HTTP_LIVE_AUDIO

[API: http live streaming]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b46	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Audio enabled for HTTP Live Streaming yes or no	
Write	f_flag	service	Audio enabled for HTTP Live Streaming yes or no	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.307 CONF_HTTP_LIVE_BITRATE

[API: http live streaming]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b45	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the bitrate in kBit the HTTP Live Streaming has allocated buffers for	
Write	t_dword	service	Set the bitrate in kBit the HTTP Live Streaming should allocate buffers for (max bitrate given in init gen)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.308 CONF_HTTP_SESSION_COOKIE_NAME

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xD07c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read session cookie name	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.309 CONF_HUMIDITY_VALUE

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c61	humidity sensor (1...x)	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Value of the humidity sensor specified by numdes (humidity 0 ... 100.00 % maps onto values 0 ... 10000)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.310 CONF_IGMP_VERSION

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e5	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read the igmp(internet group management protocol) version,	
Write	t_octet	service	Set the igmp version,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Automatic from network	0
Version1	1
Version2	2
Version3	3

© Copyright 2001-2021 Bosch Security Systems

2.311 CONF_ILLUMINATION_INTENSITY

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c22	illuminator nbr	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the white-light illumination intensity; range is from 1-100; 1->darkest; 100->brightest	
Write	t_word	user	Set the white-light illumination intensity; range is from 1-100; 1->darkest; 100->brightest; (notice: to switch the illuminator on/off, use command CONF_ILLUMINATOR_STATE)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.312 CONF_ILLUMINATOR_OPTIONS

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c1c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the number of installed white-light illuminators and their capabilities; 1st WORD: number of illuminators; 2nd WORD: capabilities of first illuminator, 3rd WORD: capabilities of second illuminator..., etc.; Available capabilities: bit0=dimmable (by CONF_ILLUMINATION_INTENSITY)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.313 CONF_ILLUMINATOR_STATE

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c1d	illuminator nbr	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get state of white-light illumination; 0=illuminator off; 1=illuminator on	
Write	t_dword	user	Switch white-light illumination; 0=illuminator off; 1=illuminator on; (notice: to change the illumination intensity, see command CONF_ILLUMINATION_INTENSITY)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.314 CONF_INFO_STAMP_VAL

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	t_octet	user	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.315 CONF_INPUT_PIN_NAME

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0108	alarm input	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Read the description for an input alarm	
Write	p_unicode	service	Set the description for an input alarm (max 32 unicode characters)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.316 CONF_INPUT_PIN_STATE

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01c0	alarm input	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=alarm input off; 1= alarm input on	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.317 CONF_INPUT_SOURCE_VAL

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0086	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	1: camera; 2: vcr; 3: color_plane; 4: auto	
Write	t_octet	service	1: camera; 2: vcr; 3: color_plane; 4: auto	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.318 CONF_INSTALLER_SEQUENCE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b0f	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal		
Write	t_dword	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

installer sequence deactivated	0
installer sequence allowed	1

© Copyright 2001-2021 Bosch Security Systems

2.319 CONF_INTERNAL_STORAGE_ENCRYPTION

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c43	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Enable : 1 or disable : 0 the xts encryption (no influence on device with permanent encryption)	
Write	t_octet	service	Enable : 1 or disable : 0 the xts encryption (no influence on device with permanent encryption)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.320 CONF_IP

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0001	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	always_legacy	Read the unit's IP address	
Write	t_dword	service	Set the unit's IP address	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Default	0	
Primary IP	1	
Auto IP	2	read direction

© Copyright 2001-2021 Bosch Security Systems

2.321 CONF_IP_STR

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x007c	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	always_legacy	Read the unit's IP address using string notation (xxx.xxx.xxx.xxx)	
Write	p_string	service	Set the unit's IP address using string notation (xxx.xxx.xxx.xxx)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Default	0	
Primary IP	1	
Auto IP	2	read direction

© Copyright 2001-2021 Bosch Security Systems

2.322 CONF_IP_V6_PREFIX_LEN

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b05	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get unit's IPv6 address prefix length (Manually assigned IP)	
Write	t_octet	service	Set unit's IPv6 address prefix length (Manually assigned IP)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.323 CONF_IP_V6_STR

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b06	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	always_legacy	Manually assigned IPv6 String or domain name	
Write	p_string	service	Manually assigned IPv6 String or domain name	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.324 CONF_IPV4_ENABLE

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Enable or disable IPv4 support,	
Write	t_dword	service	Enable or disable IPv4 support,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

IPv4 enabled	1
IPv4 disabled	0

© Copyright 2001-2021 Bosch Security Systems

2.325 CONF_IPV4_FILTER

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b3c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the list of 2 allowed pairs of one IPv4 address and a corresponding IPv4 mask (addresses and masks in network byte order; filled completely with 0.0.0.0 entries, if disabled; in case only first entry is used, the second is filled with 0.0.0.0 / 0.0.0.0, order is address1, mask1, address2, mask2)	
Write	p_octet	service	Write the list of 2 allowed pairs of one IPv4 address and a corresponding IPv4 mask (addresses and masks in network byte order; filled completely with 0.0.0.0 entries, if disabled; in case only first entry is used, the second is filled with 0.0.0.0 / 0.0.0.0, order is address1, mask1, address2, mask2)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.326 CONF_IPV6_ENABLE

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d18	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Enable or disable IPv6 support,	
Write	t_dword	service	Enable or disable IPv6 support,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

IPv6 enabled	1
IPv6 disabled	0

© Copyright 2001-2021 Bosch Security Systems

2.327 CONF_ISCSI_AUTH

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ab0	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Authentication Descriptor (see description)

Authentication Descriptor

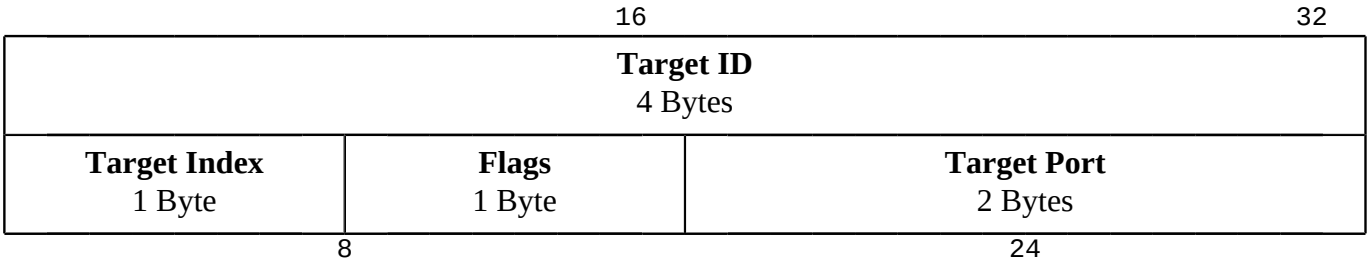
This Authentication Descriptor can, but must not, repeat up to 2 times on camera and encoder devies an up to 16 times on Windows Generic variants.

16	32
Target Address... (see description)	
Target Address ...	
Type 4 Bytes	
Params... (see description)	
Params [...]	
Params ...	
8	24

Target Address

The address of the iscsi target. If this field is set to zero, it is used as the default entry for authenticaion.

© Copyright 2001-2021 Bosch Security Systems



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES). If this field is set to -1 (0xFFFFFFFF), this descriptor is used for all remaining targets, for that no explicit descriptor is provided (default).

Target Index

The target index of the iscsi target. If this field is set to -1 (0xFF), this descriptor is used for all remaining iscsi targets with the same ip, for that no explicit descriptor is provided (default).

Flags

	Mask	Name
Bit 1	0x02	Not For Target
Bit 0	0x01	HTTP Tunnel

Target Port

The port of the iscsi target. If this value is set to zero, the port configured in CONF_ISCSI_PORT is used.

Note: For an iscsi session, all authentication descriptors are searched for the target address. If this address is found, the authentication information is used. If that descriptor is not found, but the array contains a descriptor with the same ip and the target index set to default (0xFF), the information in this record is used instead. If that descriptor is not found, but the array contains an descriptor with the ip address set to default (0xFFFFFFFF), the information in this record is used instead. If that descriptor is not found, no authentication will be performed for that iscsi session.

Type

None	0
CHAP	1
SMB (not supported)	2

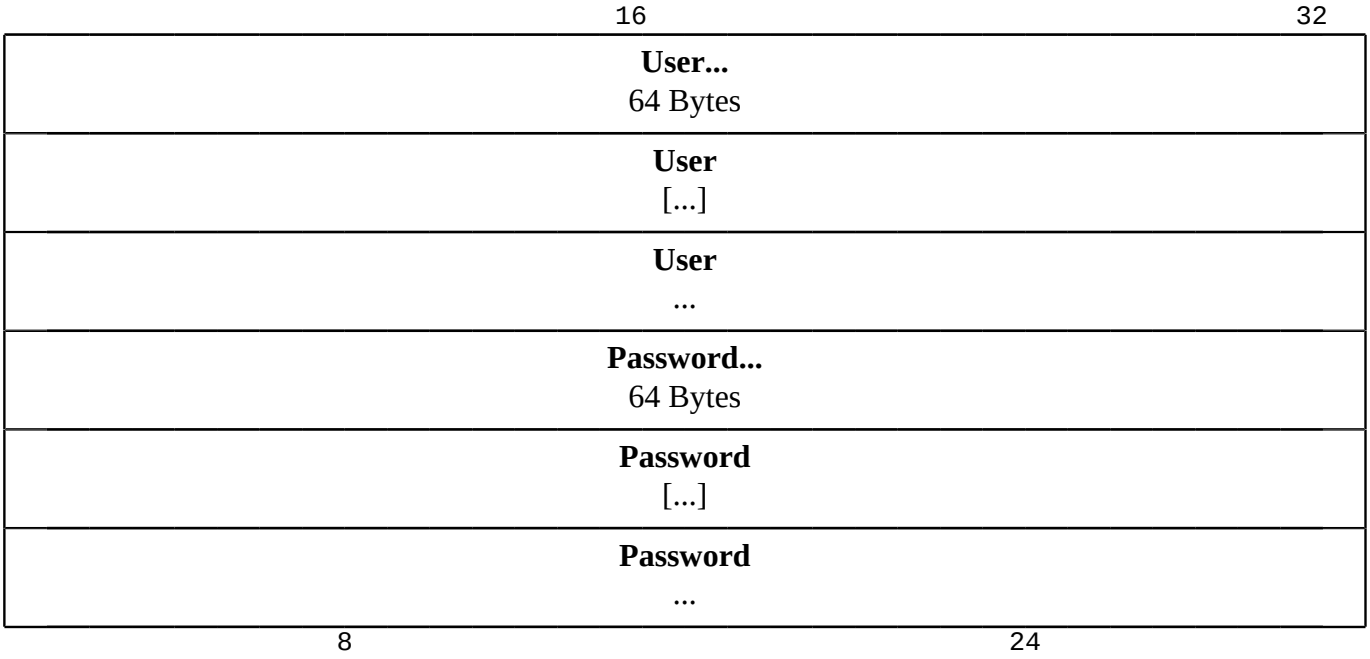
Params

This field with the the two entries for user and password only need to be in place when "Type" is set to "CHAP" (1).

Username and password is presented as ASCII string with fixed length of 64 characters. Not used trailing characters must be filled with null character.

© Copyright 2001-2021 Bosch Security Systems

'Params' payload for 'Type' = 'CHAP'



User

The user name that is used for authentication.

Password

The password that is used for authentication (CHAP: N = 64 Bytes, SMB: N = 32 Bytes).

Note: On a read command, the characters of the password are replaced by the '*'. On a write command, the password is only stored, if not all characters equal the '*' sign. Otherwise the old stored value is retained.

© Copyright 2001-2021 Bosch Security Systems

2.328 CONF_ISCSI_DATARATE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b00	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	KBit/s the ISCSI should transmit as maximum (0:=no limit send all data at once)	
Write	t_dword	service	KBit/s the ISCSI should transmit as maximum (0:=no limit send all data at once)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.329 CONF_ISCSI_DISCOVERY

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09cc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Performs a discovery and returns the result in an XML-like string; parameter ip (DWORD) and reserved (64 char) structure in p_octet	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.330 CONF_ISCSI_DISV_CACHE

[API: iscsi]

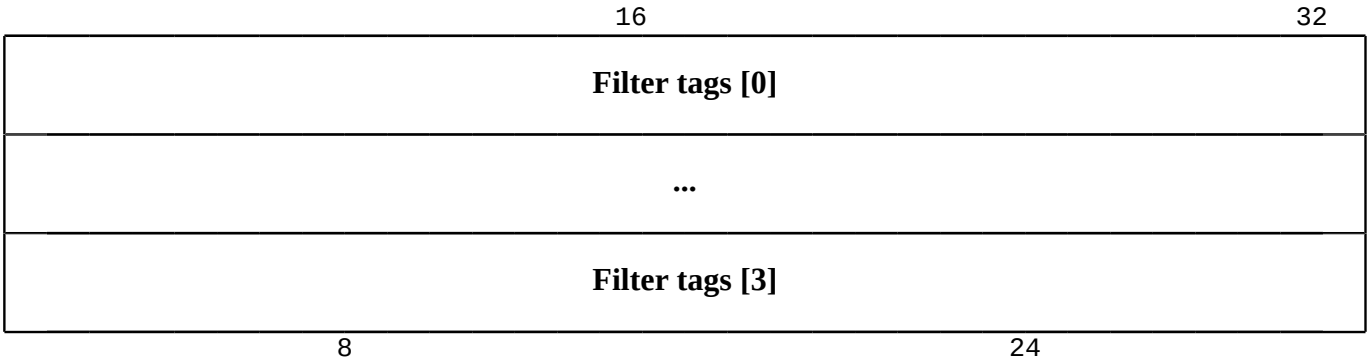
	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c30	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read out discovery cache entry.	
Write	p_octet	service	Write one discovery cache entry.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to read and write individual entries of the discovery cache.

The command supports reading and writing cache entries.

Read Payload Structure



On input you can provide one, two or all three of the following tags. With these tags you can define filters for the cache entries returned. The payload of the response is sequence of Cache Entry structure as described below.

If provide no parameters (no tag) the command returns all known IP of all target indexes and LUNs in the discovery cache. Be aware, that the output could get too long for the output buffer and there might be missing entries.

If you provide only the IP address tag, the command will return all targets and all known LUNs and the paths for the given IP address in the discovery cache.

If you provide both IP address and a target index tag, the command will return all known LUNs and all the paths for this target in the discovery cache.

If you only provide a LUN number, the paths for this LUN in all discovery cache entries will be returned. There is a reserved LUN number 0xFFFFFFFF which will always return only the first LUN of an entry in the discovery cache. If you provide this reserved LUN number without an IP address, you will get all entries in the discovery cache with only the paths for the first LUN. This is a good way to get the IP and target index of all known targets in the discovery cache without overflowing the output buffer.

If you provide an IP address, a target index and a LUN number the command will return only the paths for this target and the given LUN.

© Copyright 2001-2021 Bosch Security Systems

If there is no cache entry available which matches the given filter, the command returns the 'Tag 0x0003: No entry available' tag. In case of this command the tag always return 0x00000000 as error code (no error).

Filter tags

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

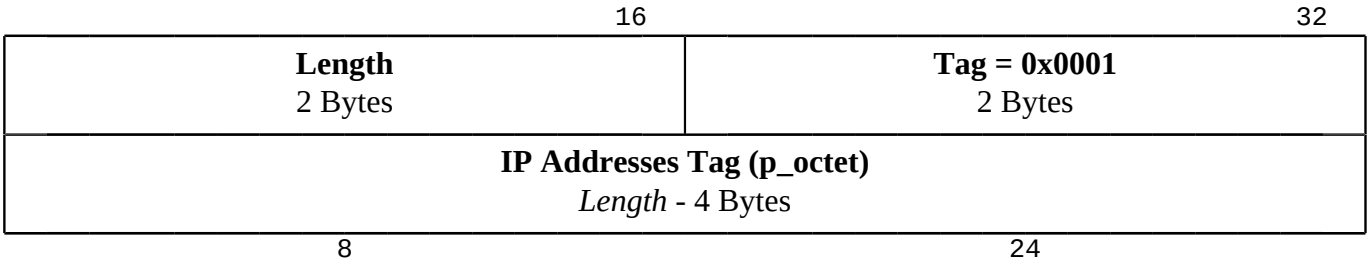
Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

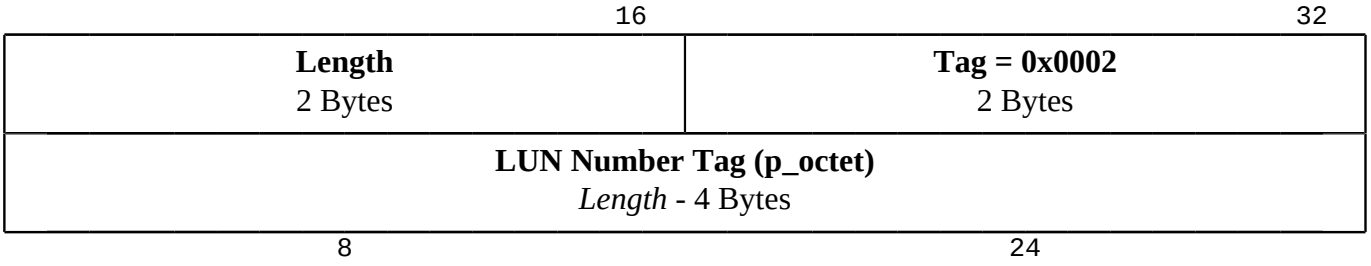
Tag 0x0001: IP Addresses Tag

Can have the length 0x0008 (IPv4) or 0x0014 (IPv6).



Tag 0x0002: LUN Number Tag

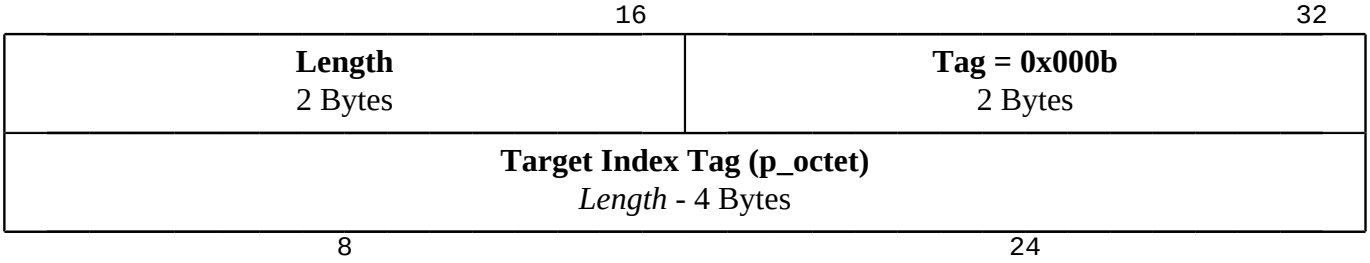
must have the length 0x0008 and must contain a 32bit LUN number in network order.



© Copyright 2001-2021 Bosch Security Systems

Tag 0x000b: Target Index Tag

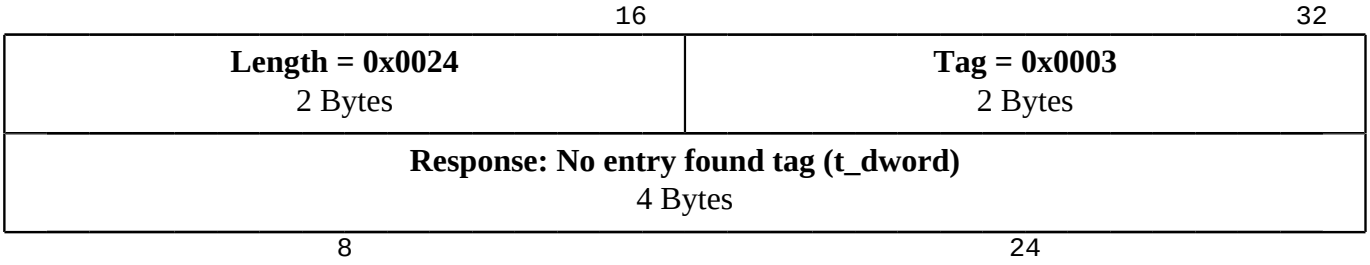
Must have the length 0x0008 and must contain a target index in network byte order.



Tag 0x0003: Response: No entry found tag

If this tag is present as first tag, no other tags will be present in the message. It tells the receiver, that there are no matching discovery cache entries.

The tag is followed by an error code (4 byte in network order). If the error code is 0x00000000 (ISCSI_ERR_NONE) there was simply no existing cache entry available. For CONF_ISCSI_DISV_CACHE command the error code is always 0x00000000. For CONF_ISCSI_MP_DISCOVERY any of the defined iSCSI error codes can occur.



Error Codes:

ISCSI_ERR_NONE	0x00
ISCSI_ERR_CONNECT	0x31
ISCSI_ERR_INV_LUN	0x33
ISCSI_ERR_LOGIN	0x34
ISCSI_ERR_INV_TARG_IDX	0x35
ISCSI_ERR_PWD	0x36
ISCSI_ERR_PROTO	0x37
ISCSI_ERR_TARG_NOT_REACH	0x38
ISCSI_ERR_NO_MEM	0x3a
ISCSI_ERR_SESS_CREATE	0x3b
ISCSI_ERR_INV_PARAMS	0x3c
ISCSI_ERR_SESS_NOT_FOUND	0x3d
ISCSI_ERR_DISCONN	0x3e
ISCSI_ERR_TIMEOUT	0x3f
ISCSI_ERR_TARGET_NOT_SUPP	0x40
ISCSI_ERR_TARGET_SESSION_LIMIT	0x41
ISCSI_ERR_CMD_NOT_SUPP	0x42

© Copyright 2001-2021 Bosch Security Systems

ISCSI_ERR_TARGET_NOT_FOUND	0x43
ISCSI_ERR_SOCK	0x5f
ISCSI_ERR_TARG_PM	0x6f
ISCSI_SOCK_CLOSED	0x7f
ISCSI_ERR_TCP_CONN_RST	0x8f
ISCSI_ERR_INTR_NOT_SUPP	0x9f
ISCSI_ERR_IP_ZERO	0xa0
ISCSI_ERR_OUT_OF_RES	0xa1

Response/Write Payload Structure

CacheEntry

In write direction the command expects the Cache Entry encoding described below. This encoded cache entry is written to the discovery cache. Cache entries which are written with this command are stored in persistent cache and are persistent over a device reset.

In write direction this command must carry a valid external instance bitmask which has to be unequal 0. If it is 0 the command is returned with an error.

To delete such an entry you must provide the Cache Entry encoding with only the target IP, target index and external instance tag. The entry is cleared, when all external instance bits are cleared.

Every cache entry is started with a surrounding start tag (0x8009) followed by the IP for which this entry was generated (tag 0x000a). After this IP entry the target index is indicated to resolve a unique target on this IP (tag 0x000b). A surrounding tag (0x8008) marks one LUN entry which has several path entries, each surrounded by a 0x8004 tag.

If there is no entry the 'Tag 0x0003: No entry available' tag is returned."

CacheEntry

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0x8009: Cache Entry Tag

Each cache entry start with this tag and is followed by an target IP tag. If there is no multipathing information, there will be no following 'LUN entry tags'.

16	32
Length = 0x0020 2 Bytes	**Tag = 0x8009** 2 Bytes
8	24

Tag 0x000a: Target IP Tag

Original or main IP address of the iSCSI target. Through this IP the session was initially created. If the length field is 8 the payload is a IPv4 address (network byte order). If the length field is 20 the payload contains IPv6.

16	32
Length = 0x0024 2 Bytes	**Tag = 0x000a** 2 Bytes
Target IP Tag (t_dword) 4 Bytes	
8	24

Tag 0x000b: Target Index Tag

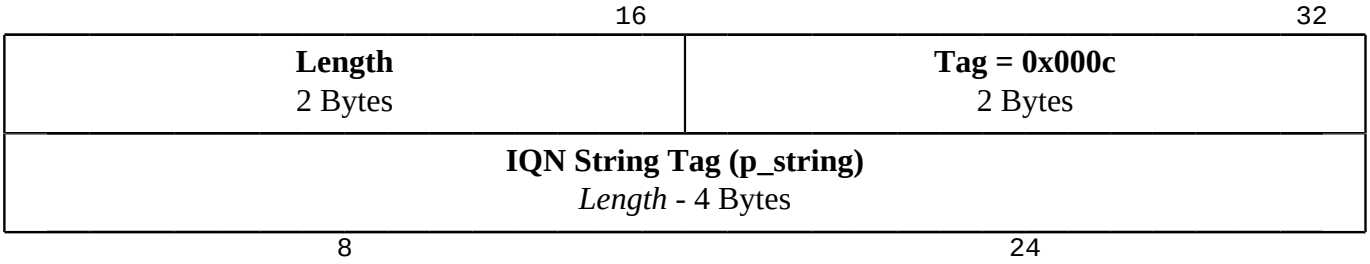
Target index on the main IP. This is the index of the target if there are multiple targets available on one IP.

16	32
Length = 0x0024 2 Bytes	**Tag = 0x000b** 2 Bytes
Target Index Tag (t_dword) 4 Bytes	
8	24

© Copyright 2001-2021 Bosch Security Systems

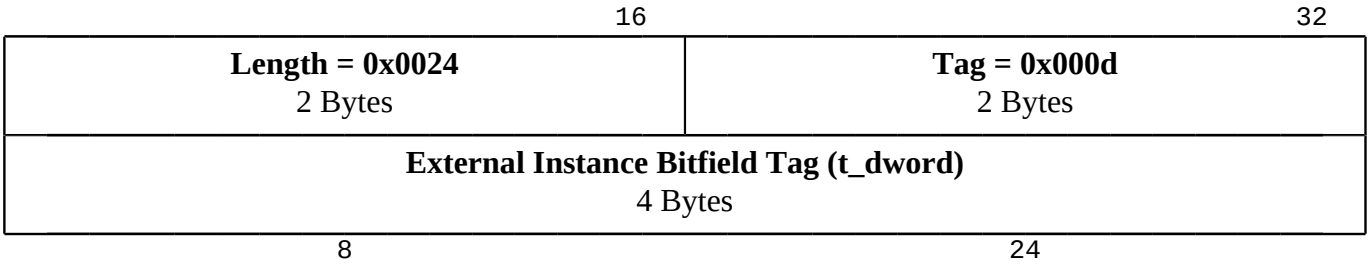
Tag 0x000c: IQN String Tag

IQN string of the target. This is the unique name identifier the target returns at discovery and which is needed at login state to address the target. In multipathing this IQN is used for all connection IP addresses.



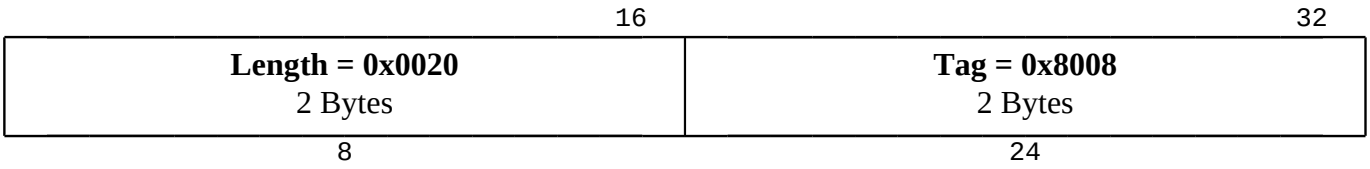
Tag 0x000d: External Instance Bitfield Tag

The external instance is a 32 bit wide bitfield. It is used to identify who has set this cache entry. In VRM environment the first VRM may set bit 0. The secondary VRM will use bit 1. With this bitfield it is tracked, who wrote this cache entry. If one instance deletes the cache entry, only the bit used by this instance is cleared. If any other bits are still set, the entry is not deleted. When this bitfield becomes zero at a delete request, the entry is really deleted. ATTENTION: This field must always have a value other then 0. Writing a cache entry from outside world with external instance equal 0 will cause a write error.



Tag 0x8008: LUN Entry Group Tag

Surrounding tag for one LUN entry. This tag will have several 'Path entry' tags as subtags.

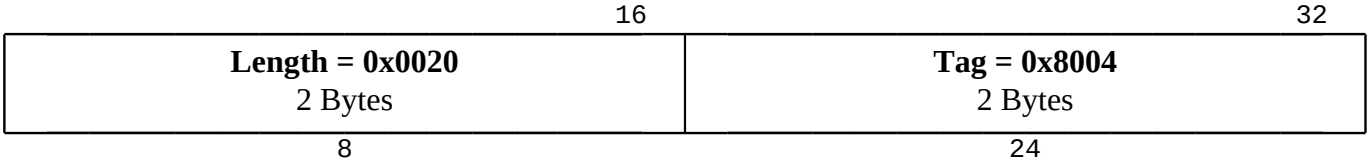


Encloses tags: Path Entry Group Tag

© Copyright 2001-2021 Bosch Security Systems

Tag 0x8004: Path Entry Group Tag

Tag which marks one path entry. The first path entry will always be the path on the main path.

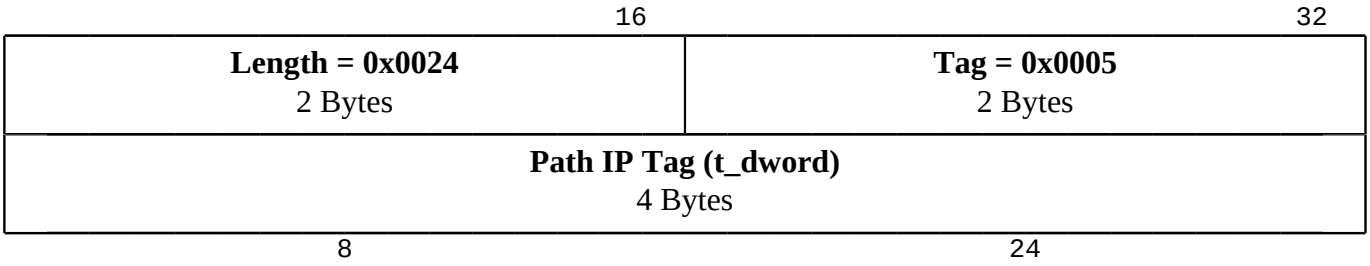


Encloses tags: Path IP Tag, LUN Number Tag, Flags Tag

Enclosed by tag: LUN Entry Group Tag

Tag 0x0005: Path IP Tag

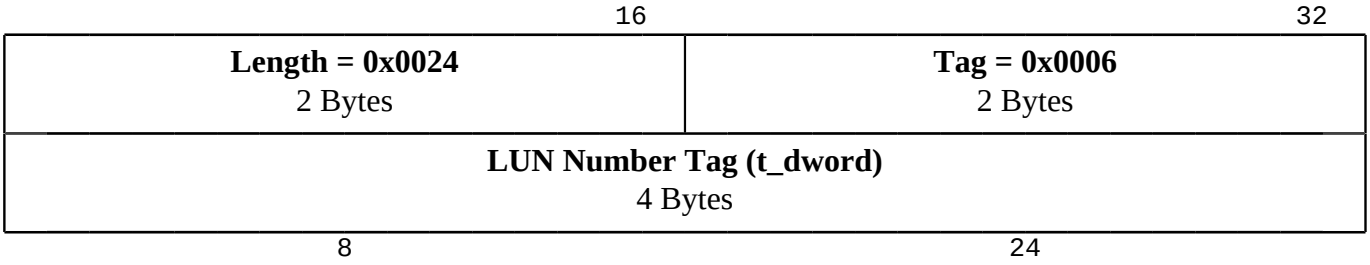
IP address of this path. If the length field is 8 the payload is a IPv4 address (network byte order). If the length field is 20 the payload contains IPv6.



Enclosed by tag: Path Entry Group Tag

Tag 0x0006: LUN Number Tag

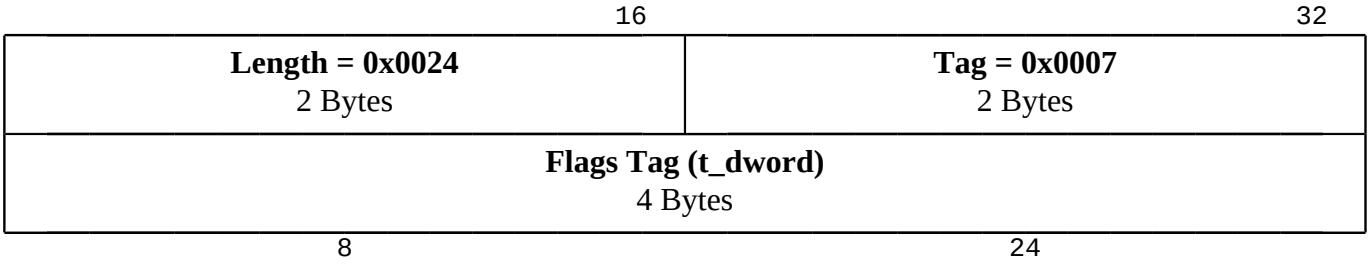
The number of the LUN the LUN will have on this path. The first entry in the paths list marks the LUN number on the main path.



Enclosed by tag: Path Entry Group Tag

© Copyright 2001-2021 Bosch Security Systems

Tag 0x0007: Flags Tag



Enclosed by tag: Path Entry Group Tag

	Mask	Name	Description
Bit 3	0x00000008	Fallback Path	This path is not optimal and should only be used as backup path in case of all preferred paths fail.
Bit 2	0x00000004	Active Path	This path is an active path (but maybe not preferred if bit 1 is cleared).
Bit 1	0x00000002	Preferred Path	This path is a prefered path
Bit 0	0x00000001	TPGS Supported	Target supports TPGS (Target Portal Group Support) on this path. Necessary for proper multipathing support.

© Copyright 2001-2021 Bosch Security Systems

2.331 CONF_ISCSI_FLUSH_DISVCACHE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c19	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Flush the ISCSI discovery cache. See detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command flushes one entry in iSCSI discovery cache. Providing an IP and target index is mandatory.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

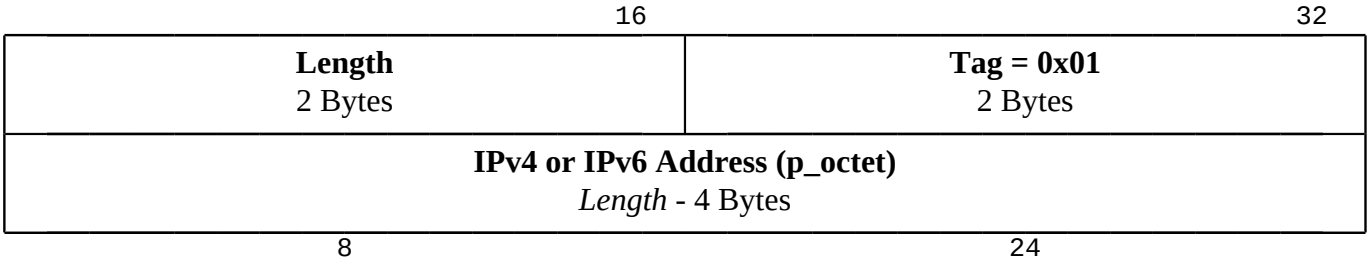
Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

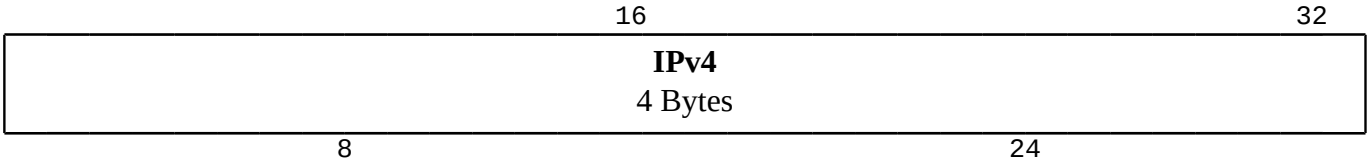
© Copyright 2001-2021 Bosch Security Systems

Tag 0x01: IPv4 or IPv6 Address

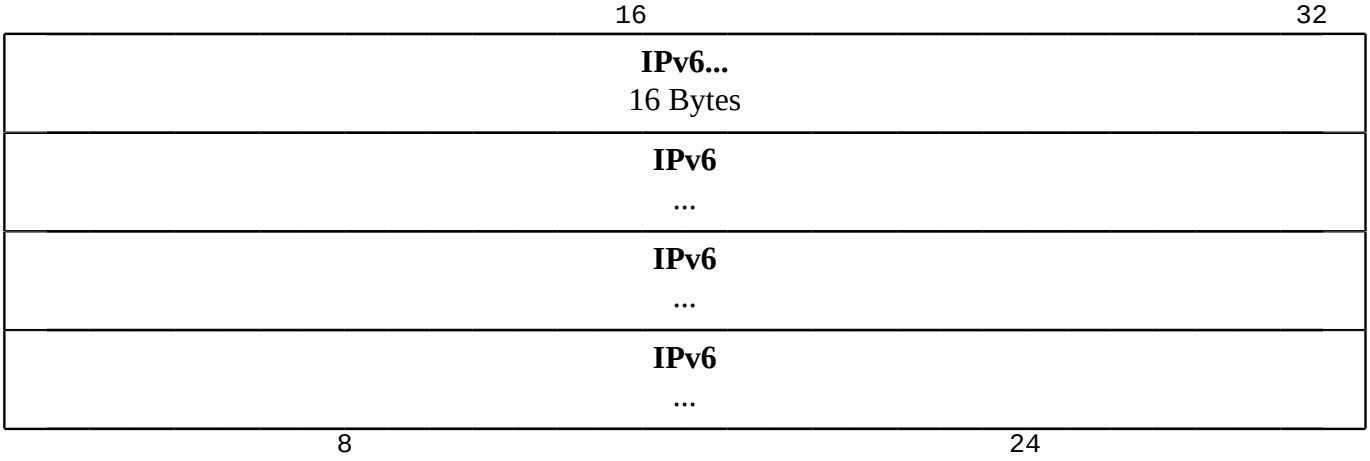


IPv4 or IPv6 Address

Payload for 'Length' = '8'

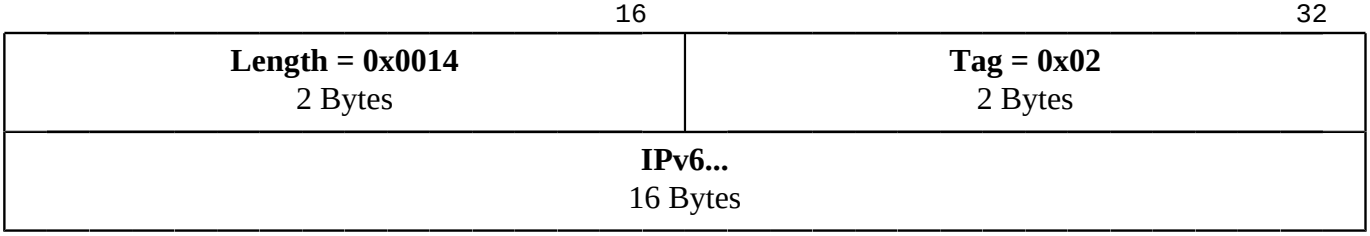


Payload for 'Length' = '20'



Tag 0x02: IPv6 Address

The payload contains an IPv6 address in network byte order which is to be flushed in the discovery cache.



© Copyright 2001-2021 Bosch Security Systems

IPv6 ...
IPv6 ...
IPv6 ...

8 24

Tag 0x03: Target Index

The payload of this tag contains a target index. If there are multiple targets on one IP you can flush the cache for individual target entries by passing the target index in network byte order.

16 32

Length = 0x0024 2 Bytes	**Tag = 0x03** 2 Bytes
Target Index (t_dword) 4 Bytes	

8 24

© Copyright 2001-2021 Bosch Security Systems

2.332 CONF_ISCSI_INITIATOR_NAME

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Returns the used initiator name; only applicable when iSCSI is connected	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.333 CONF_ISCSI_INITIATOR_NAME_EXTENTION

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Returns the used initiator name extention; used for identification only	
Write	p_string	service	Set the used initiator name extention; used for identification only	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.334 CONF_ISCSI_IP

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09aa	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Deprecated in fw > 4.00	
Write	t_dword	service	Deprecated in fw > 4.00	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.335 CONF_ISCSI_LOCK_OVERRIDE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal		
Write	p_string	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Target
N Bytes

Format of Payload String

Target_Id:Target_Idx:Lun

	Max. Length	Limits	Description
Target_Id			IPv4 address of the target as string like x.x.x.x
Target_Idx		>= 1	Target index of the target at the given IP
Lun		>= 1	Number of the LUN the lock should be overwritten

Note: The datatype of this command was FLAG and changed to STRING with fw 2.50. The target_id needs o be resolved by rules configured by CONF_TARGET_ID_RESOLVE_RULES. In older version the target id was the ipv4 address, which is now the default rule for target id resolving. The string format of the target id is the same as of the ipv4.

© Copyright 2001-2021 Bosch Security Systems

2.336 CONF_ISCSI_LOCK_RELEASE_ON_LEAVE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Returns wheather the next iSCSI logout should use the release action	
Write	f_flag	service	Advices the iSCSI initiator to release a lock at the target lun when the storage medium transitions away from iSCSI; this option clears its value after a successful release	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.337 CONF_ISCSI_LOWERDATARATE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b47	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	KBit/s the ISCSI lower Limit for the Iscsi Data rate Throttling	
Write	t_dword	service	KBit/s the ISCSI lower Limit for the Iscsi Data rate Throttling	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.338 CONF_ISCSI_LUN

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ac	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Deprecated in fw > 4.00	
Write	t_dword	service	Deprecated in fw > 4.00	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.339 CONF_ISCSI_MNI

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aa0	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Monitor iscsi targets.	
Write	p_octet	service	Monitor iscsi targets.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Use this command to monitor iscsi targets.

If you write this command, all the targets of the request packets are added/delete to the current monitoring list.

If you read this command, you will get the description of all currently monitored iscsi targets in the response packet.

Every time the status of a monitored target changes, a message is sent out with the current status of this target.

Request Payload Structure

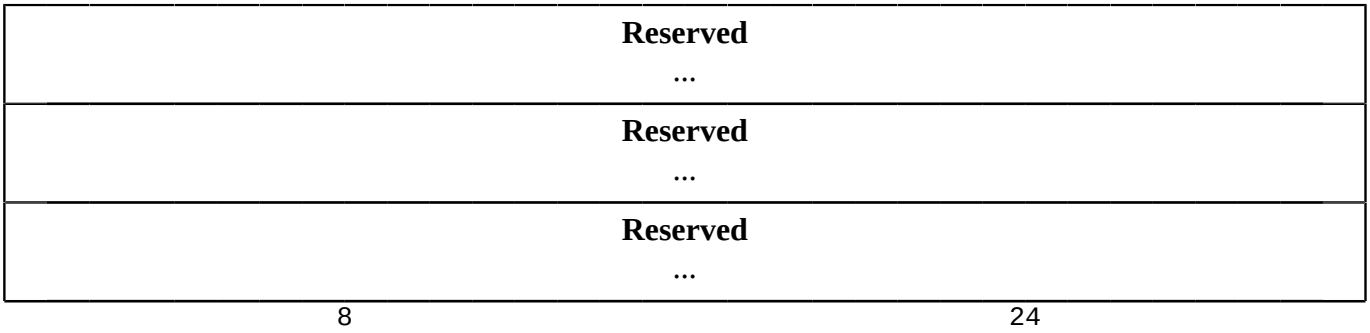
Target Descriptor [0] (see description)
...
Target Descriptor [N] (see description)

Target Descriptor

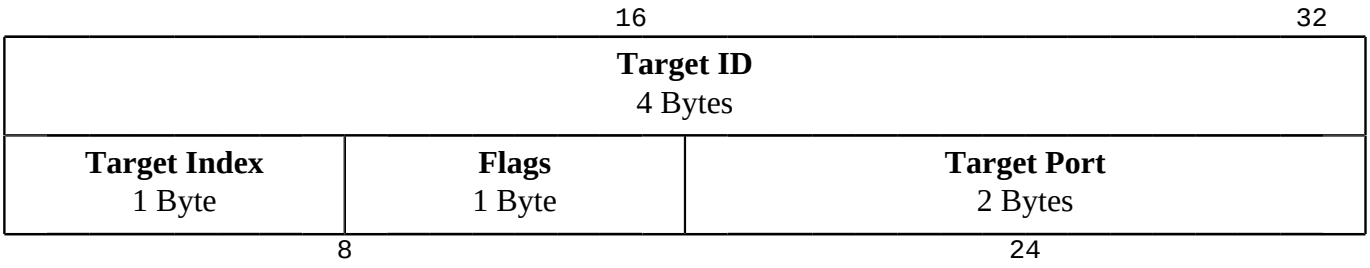
16 32

Target Address... (see description)	
Target Address ...	
Action 1 Byte	**Reserved...** 19 Bytes
Reserved ...	

© Copyright 2001-2021 Bosch Security Systems



Target Address



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES). If this field is set to -1 (0xFFFFFFFF), this descriptor is used for all remaining targets, for that no explicit descriptor is provided (default).

Target Index

The target index of the iscsi target. If this field is set to -1 (0xFF), this descriptor is used for all remaining iscsi targets with the same ip, for that no explicit descriptor is provided (default).

Flags

	Mask	Name
Bit 1	0x02	Not For Target
Bit 0	0x01	HTTP Tunnel

Target Port

The port of the iscsi target. If this value is set to zero, the port configured in CONF_ISCSI_PORT is used.

Action

Delete	0x00
Add	0x01

Response Payload Structure

© Copyright 2001-2021 Bosch Security Systems

Target Descriptor [0] (see description)
...
Target Descriptor [N] (see description)

Target Descriptor

16			32
Target Address... 8 Bytes			
Target Address ...			
Action 1 Byte	**Status** 1 Byte	**Error** 1 Byte	**Reserved** 1 Byte
Starttime 4 Bytes			
Session Status 1 Byte	**Session Error** 1 Byte	**Reconnects** 2 Bytes	
Reserved 4 Bytes			
Uptime 4 Bytes			
8		24	

Target Address

16		32
Target ID 4 Bytes		
Target Index 1 Byte	**Flags** 1 Byte	**Target Port** 2 Bytes
8		24

Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES). If this field is set to -1 (0xFFFFFFFF), this descriptor is used for all remaining targets, for that no explicit descriptor is provided (default).

© Copyright 2001-2021 Bosch Security Systems

Target Index

The target index of the iscsi target. If this field is set to -1 (0xFF), this descriptor is used for all remaining iscsi targets with the same ip, for that no explicit descriptor is provided (default).

Flags

	Mask	Name
Bit 1	0x02	Not For Target
Bit 0	0x01	HTTP Tunnel

Target Port

The port of the iscsi target. If this value is set to zero, the port configured in CONF_ISCSI_PORT is used.

Action

The value of the request packet. Zero in messages.

Delete	0x00
Add	0x01

Status

Fail	0x00
Success	0x01

Error

ISCSI_NO_ERROR	0x00	
ISCSI_MNI_ERR_INV_TARG	0x01	if the assigned target address is invalid (e.g. zero).
ISCSI_MNI_ERR_FULL	0x02	if the maxium of 64 iscsi targets is reached and a request packet with the ADD action was sent.
ISCSI_MNI_ERR_TARG_PRES	0x03	if a request packet if the ADD action was sent and the target address is already monitored.
ISCSI_MNI_ERR_NOT_FOUND	0x04	if a request packet if the DELETE action was sent and the target address is not currently monitored.
ISCSI_MNI_ERR_INTERN	0x05	

Starttime

The time when the monitoring process was started (in seconds since 2000).

© Copyright 2001-2021 Bosch Security Systems

Session Status

OFFLINE	0x00
ONLINE	0x01
ERROR	0x02

Session Error

ISCSI_ERR_CONNECT	0x31
ISCSI_ERR_LOGIN	0x34
ISCSI_ERR_INV_TARG_IDX	0x35
ISCSI_ERR_PWD	0x36
ISCSI_ERR_PROTO	0x37
ISCSI_ERR_TARG_NOT_REACH	0x38
ISCSI_ERR_NO_MEM	0x3a
ISCSI_ERR_SESS_CREATE	0x3b
ISCSI_ERR_INV_PARAMS	0x3c
ISCSI_ERR_SESS_NOT_FOUND	0x3d
ISCSI_ERR_DISCONN	0x3e
ISCSI_ERR_TIMEOUT	0x3f
ISCSI_ERR_SOCK	0x5f
ISCSI_SOCK_CLOSED	0x7f
ISCSI_ERR_TCP_CONN_RST	0x8f
ISCSI_ERR_IP_ZERO	0xa0

Reconnects

The number of times the session was reconnected since the begin of the monitoring process.

Uptime

The number of seconds the session is online.

© Copyright 2001-2021 Bosch Security Systems

2.340 CONF_ISCSI_MP_DISCOVER

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c32	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Multipath discover	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

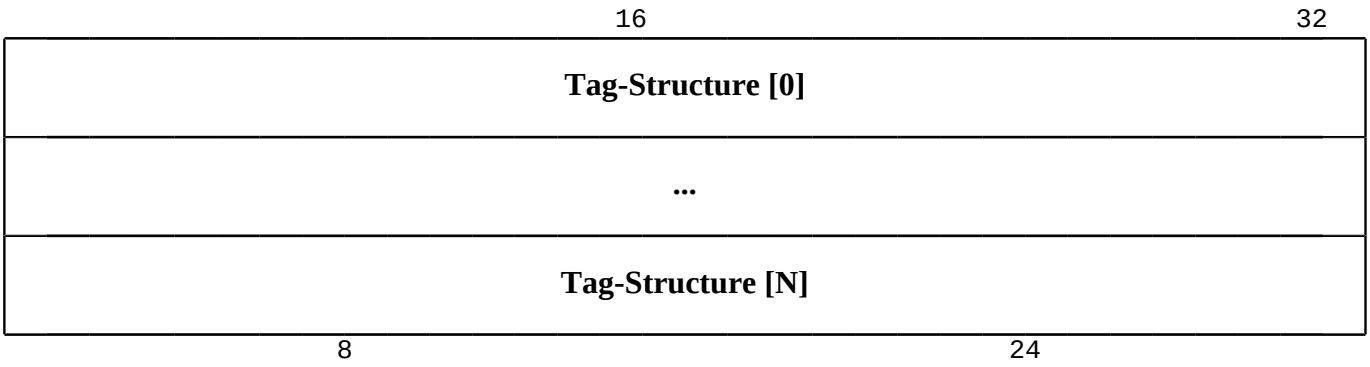
Description

This command initiates a multipath discovery of a target. This means iSCSI stack tries to detect all known alternative pathes for all LUNs for the given main path.

The user can select alternative pathes to be scanned with the 0x0003 tag. If no 0x0003 tag is given all pathes found at discovery are scanned.

ATTENTION: The command can take up to the amount of scanned pathes multiplied with 10 seconds to complete! This scan does not create any persistent discovery cache entries. All cached data is cleared after the scan.

Request Payload Structure



The payload must contain the tag 0x0001 and the tag 0x0002. Optionally you can pass one or more tag 0x0003 to tell the command to only scan the given pathes. The payload of the tag 0x0001 and tag 0x0003 can either be IPv4 or IPv6. Which IP is contained is indiacted by the tag length. 0x0008 is IPv4 0x0014 is IPv6. The tag 0x0002 indicates a target index.

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

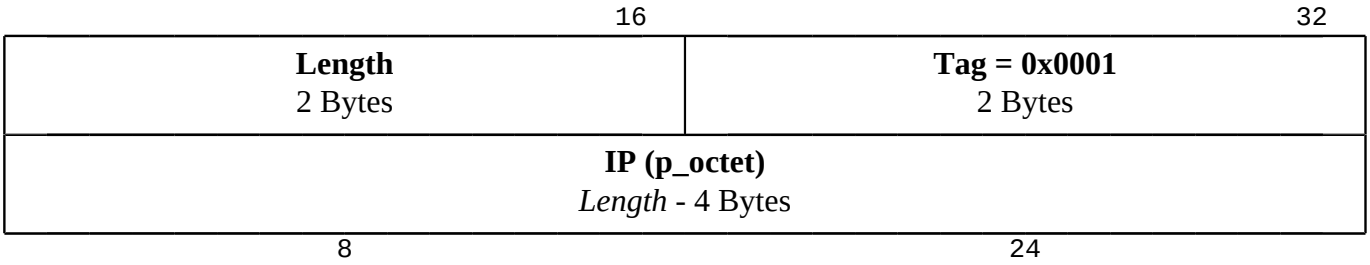
© Copyright 2001-2021 Bosch Security Systems

Tag

Tag specifying the encoding and meaning of the value

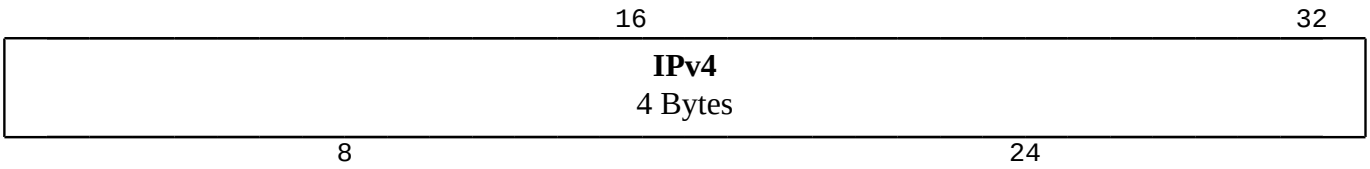
Tag 0x0001: IP

Indicates the main path to the target to scan

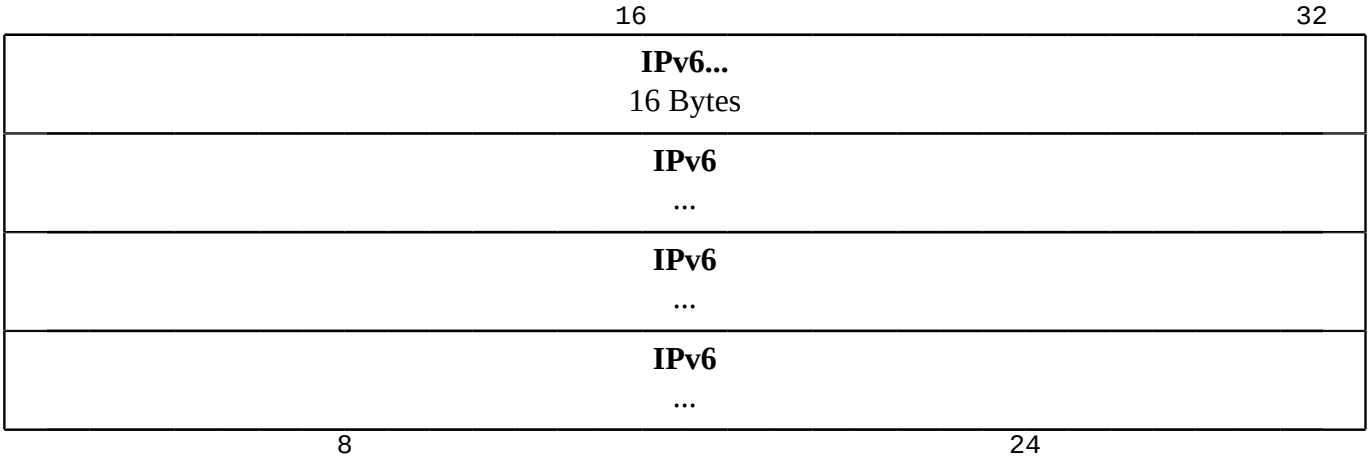


IP

Payload for 'Length' = '8'



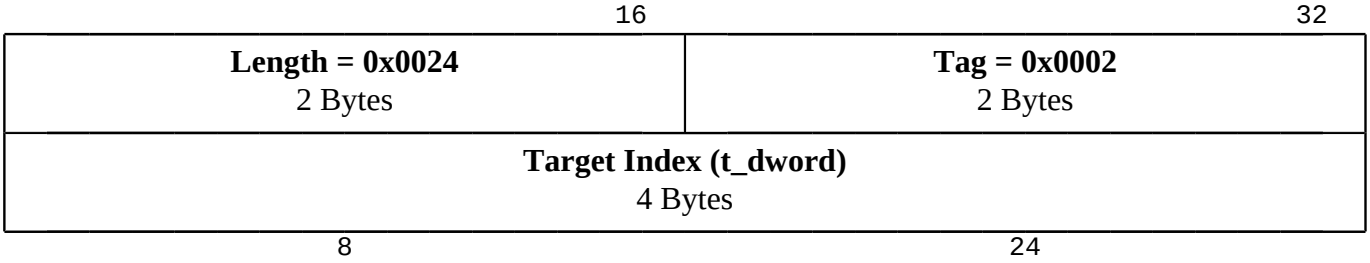
Payload for 'Length' = '20'



© Copyright 2001-2021 Bosch Security Systems

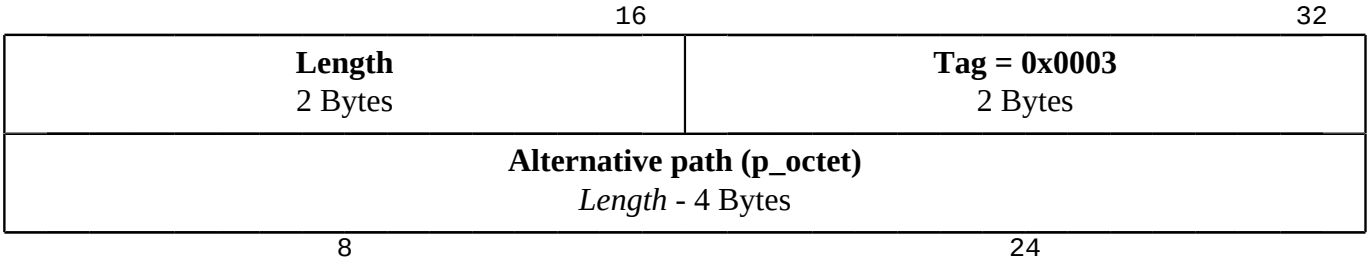
Tag 0x0002: Target Index

The target index (4 byte number) on network byte order



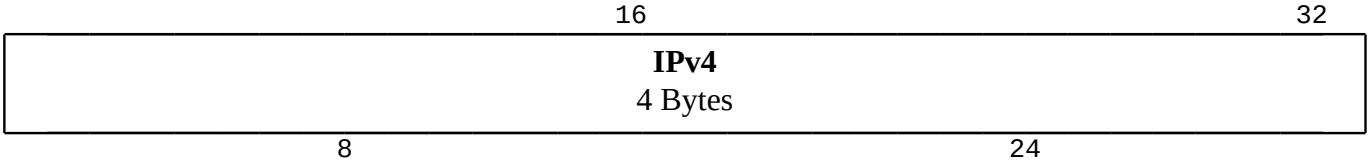
Tag 0x0003: Alternative path

Gives the IP of an alternative path to scan. If not given all found pathes are scanned

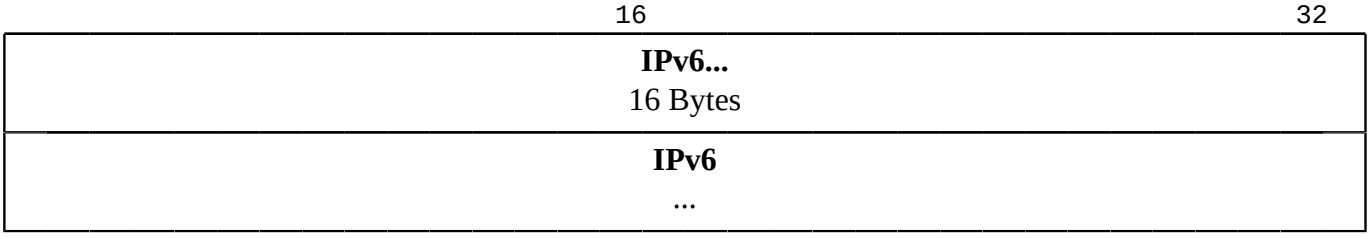


Alternative path

Payload for 'Length' = '8'



Payload for 'Length' = '20'



© Copyright 2001-2021 Bosch Security Systems

IPv6 ...
IPv6 ...

8 24

Response Payload Structure

This payload equals the payload of the command 'CONF_ISCSI_DISV_CACHE'

Command Specific Errors

The following specific erros might be returned on RCP level. If such an error occurs the command was not executed.

iSCSI specific errors

ISCSI_ERR_INV_PARAMS	0x3c	Error parsing the command (error in tag structure)
ISCSI_ERR_TARGET_SESSION_LIMIT	0x41	Too many scan active (only one to the same IP is allowed). Retry later.
ISCSI_ERR_NO_MEM	0x3a	No free job to do the scan. Retry later.
No cache entry	0x0003	The command can also return the 'No cache entry' tag followed by an error code. With this the command returns errors which occur while discovering the target. All known iSCSI error codes can occur there (see CONF_DISV_CACHE for a list of error codes).

© Copyright 2001-2021 Bosch Security Systems

2.341 CONF_ISCSI_MULTIPATH

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bee	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the behaviour of iSCSI multipathing (0=off, 1=only use prefered pathes, 2=use all pathes, 3=use NetApp specific path selection)	
Write	t_dword	service	Get the behaviour of iSCSI multipathing (0=off, 1=only use prefered pathes, 2=use all pathes, 3=use NetApp specific path selection)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.342 CONF_ISCSI_MULTIPATH_STATE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c14	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the multipathing state of all active iSCSI connections	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The message normally contains one or more 0x00 tags or one 0x04 tag if there is no active iSCSI connection. Inside the 0x00 tag there should appear one 0x01, 0x05, 0x02, 0x03 and 0x06 tag.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 0x8000: Group tag

(Surrounds information for one connection)

One group is marked by the 0x8000 starting tag. Each group gives information about one active iSCSI connection. It normally contains one 0x0001, one 0x0005, one 0x0002 and one 0x0003 tag to describe the multipath state of the connection.

© Copyright 2001-2021 Bosch Security Systems

The length field contains the length of all included subtags plus the group header itself.

The group tag is present to simplify parsing and to join subtags to a unique connection description.

16	32
Length = 0x0020 2 Bytes	**Tag = 0x8000** 2 Bytes
8	24

Tag 0x0001: Main path IP

Original or main IP address of the iSCSI target. Through this IP the session was initialy created. If the length field is 8 the payload is a IPv4 address (network byte order). If the length field is 20 the payload contains IPv6.

16	32
Length 2 Bytes	**Tag = 0x0001** 2 Bytes
Main path IP (p_octet) *Length* - 4 Bytes	
8	24

Main path IP

Payload for 'Length' = '8'

16	32
IPv4 4 Bytes	
8	24

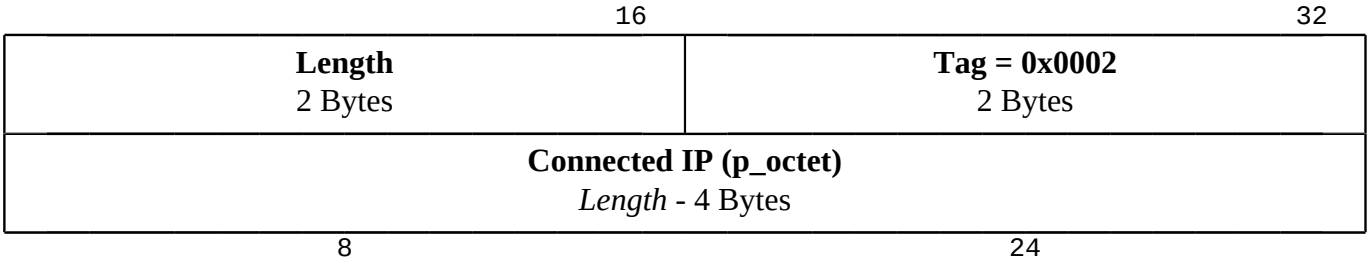
Payload for 'Length' = '20'

16	32
IPv6... 16 Bytes	
IPv6 ...	
IPv6 ...	
IPv6 ...	
8	24

© Copyright 2001-2021 Bosch Security Systems

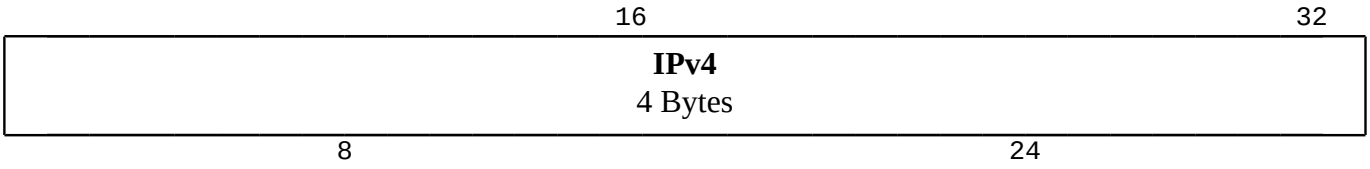
Tag 0x0002: Connected IP

The IP address the connection uses at the moment. This may be the IP of an alternative path. If the length field is 8 the payload is a IPv4 address (network byte order). If the length field is 20 the payload contains IPv6.

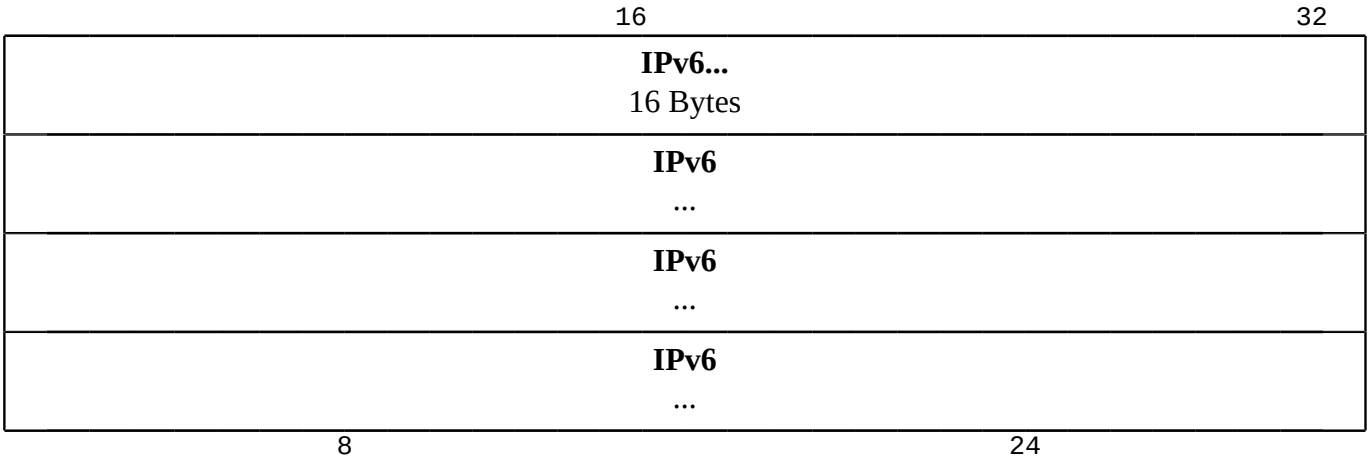


Connected IP

Payload for 'Length' = '8'



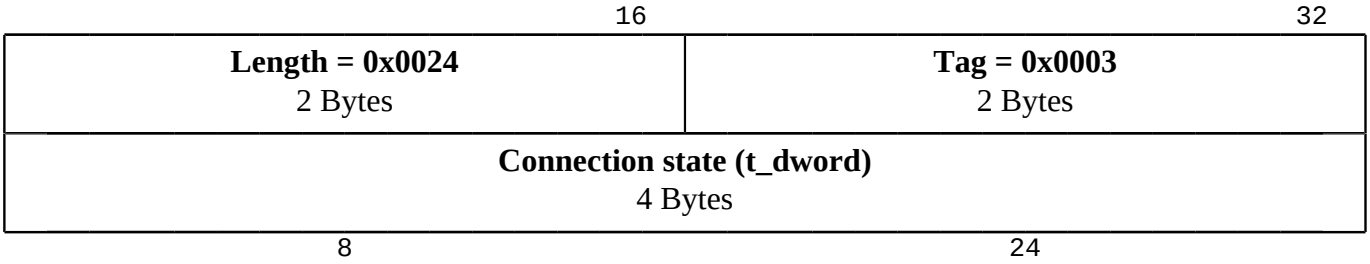
Payload for 'Length' = '20'



© Copyright 2001-2021 Bosch Security Systems

Tag 0x0003: Connection state

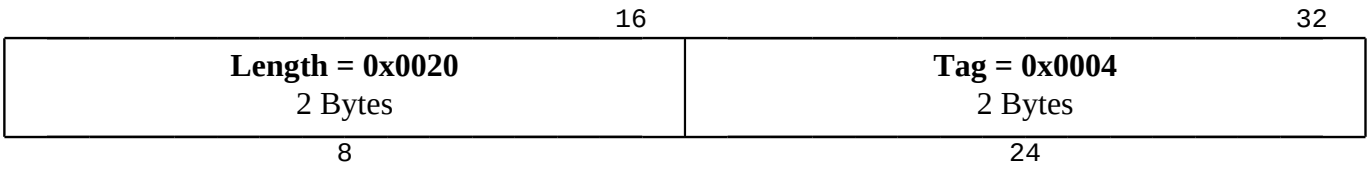
The payload is 4 byte containing a 32 bit value (DWORD) in network byte order. The value must be interpreted as a bit field with the following meaning:



	Mask	Name	Description
Bit 6	0x00000040	Check Failed	The connection is connected via the main IP, but the regularly check of the availability of the alternative paths failed. So at least one alternative path might not be reachable.
Bit 5	0x00000020	Closing	The session is about to be closed gracefully. There are no more users on this session and it will be closed.
Bit 4	0x00000010	NonOptimized	Connection runs on a non optimized path (lower performance) and will try to go back to an main path as soon as possible.
Bit 3	0x00000008	Alternate Path Unoptimized	The connection runs through an alternative path which is not declared as being optimized or not optimized. Connection will go back to the main path when possible.
Bit 2	0x00000004	Alternate Path Optimized	The connection is connected via an alternative path. So The connection runs optimized but not through the main path.
Bit 1	0x00000002	MainIP	The connection is connected via the main IP (main path or prefered path) and working optimized.
Bit 0	0x00000001	Offline	The connection is offline and not in a working state.

Tag 0x0004: No active connection

If this tag is present as first tag, no other tags will be present in the message. It tells the receiver, that no active connection exists at the moment. So the iSCSI stack is idle.



© Copyright 2001-2021 Bosch Security Systems

Tag 0x0005: Target index

The target index of the target which is addressed over the Target IP if multiple targets are available through this IP.

16	32
Length = 0x0024 2 Bytes	**Tag = 0x0005** 2 Bytes
Target index (t_dword) 4 Bytes	
8	24

Tag 0x0006: Multipath support

16	32
Length = 0x0024 2 Bytes	**Tag = 0x0006** 2 Bytes
Multipath support (t_dword) 4 Bytes	
8	24

No Multipath	0	The target has no multipath support and only one of the states Bit 0 (offline) and Bit 1 (connected optimized) is reported, as the target can only be on- or offline. It has no alternative pathes.
Multipath	1	The target supports multipathing and there are alternative pathes available. All states from above can be reported for the target.

Example

This is the payload when a target supports multipathing. The IP of the origin connection is 10.1.1.10. The target with index 2 on this IP is used. The connection runs over the IP 10.1.1.12 at the moment. This is a path which is marked as non optimized.

16	32
Length 0x001c	**Group Tag** 0x8000
Length 0x0008	**Subtag** 0x0001
IPv4 0x0a 0x01 0x01 0x0a	
Length 0x001c	**Subtag** 0x0005

© Copyright 2001-2021 Bosch Security Systems

Target Index 0x00 0x00 0x00 0x02	
Length 0x001c	**Subtag** 0x0002
IPv4 0x0a 0x01 0x01 0x0c	
Length 0x001c	**Subtag** 0x0003
Flags 0x00 0x00 0x1 0x00 (uint32 0x00000010)	
Length 0x001c	**Subtag** 0x0006
Multipath support 0x00 0x00 0x00 0x01 (uint32 0x00000001)	

8 24

When no iSCSI connection is requested/active at the moment (e.g. no recording running), the following payload is returned

16 32

Length 0x001c	**No active connection Tag** 0x0004

8 24

© Copyright 2001-2021 Bosch Security Systems

2.343 CONF_ISCSI_PORT

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ab	0: connect port, 1: tunnel port, 2: server port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Iscsi port	
Write	t_word	service	Iscsi port	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.344 CONF_ISCSI_READDATARATE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b3a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	KBit/s the ISCSI should read as maximum (0:=no limit send all data at once)	
Write	t_dword	service	KBit/s the ISCSI should read as maximum (0:=no limit send all data at once)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.345 CONF_ISCSI_SEG_SIZE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aff	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Maximum segment size for ISCSI transfers (should be a 2^n value like 8192, 16384 or 65536)	
Write	t_dword	service	Maximum segment size for ISCSI transfers (should be a 2^n value like 8192, 16384 or 65536)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.346 CONF_ISCSI_SERVER_STATE

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a2b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the state of the iscsi server (0: server down, 1: server running)	
Write	t_dword	service	Set the iscsi server state (0: shutdown server, 1: start server)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Note: The iscsi server is also started or stopped when you write the command CONF_STORAGE_LIST. If the storage list is empty or only contains entries with the 'iSCSI export' field set to 'For local use only' (0x00), the server is stopped if it is running cause no targets and luns would be provided. If the list contains at least one entry with the 'iSCSI export' field set to 'Make storage available through iSCSI' (0x01), the server is started, if not already running, to provide the iscsi service.

© Copyright 2001-2021 Bosch Security Systems

2.347 CONF_ISCSI_TARGET

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ad	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Gets the iscsi target name string	
Write	p_string	service	Deprecated in fw > 4.00	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.348 CONF_ISCSI_TARGET_IDX

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09f9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Deprecated in fw > 4.00	
Write	t_dword	service	Deprecated in fw > 4.00	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.349 CONF_ISCSI_TARGET_PWD

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ce	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	user	Get the password to authenticate at the iSCSI server. FW version >= 4.00: This command returns the password of the first entry of the CONF_ISCSI_AUTH list.	
Write	p_string	service	Set the password to authenticate at the iSCSI server. FW version >= 4.00: This command sets the password of the first entry of the CONF_ISCSI_AUTH list. If all characters of the string are the '*', the old stored value is not replaced.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.350 CONF_ISCSI_TCP_CONNECTIONS

[API: iscsi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09ae	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Ger the number of concurrent tcp connections to iscsi target	
Write	t_dword	service	Set the number of concurrent tcp connections to iscsi target	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.351 CONF_IVA_COUNTER_VALUES

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b4a	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	iva	Reset all counters	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command, which is at the same time a message, is used to retrieve the list of configured counters with the their current values. The output has the following structure: Initially the number of configured counters is set. Afterwards for each counter the data consisting of counter id, counter type, counter name in unicode and counter value is provided.

Payload Structure

Number of Counters 1 Byte	**Counter Data [0] ...** (see description)
Counter Data [0] (see description)	...

Number of Counters

This values gives the number of counters available in the payload. In the subsequent the data of each counter are sequentially provided.

Counter Data

	16	32
Id 1 Byte	**Type** 1 Byte	**Name...** 64 Bytes
Name [...]		
Name ...		

© Copyright 2001-2021 Bosch Security Systems

Name ...	**Value...** 4 Bytes
Value ...	
8	24

Id

Each counter has a unique id.

Type

Not supported yet.

Name

A unique name can be assigned to a counter, which is stored in unicode (maximum: 32 characters).

Value

The current counter value (DWORD).

© Copyright 2001-2021 Bosch Security Systems

2.352 CONF_JPEG

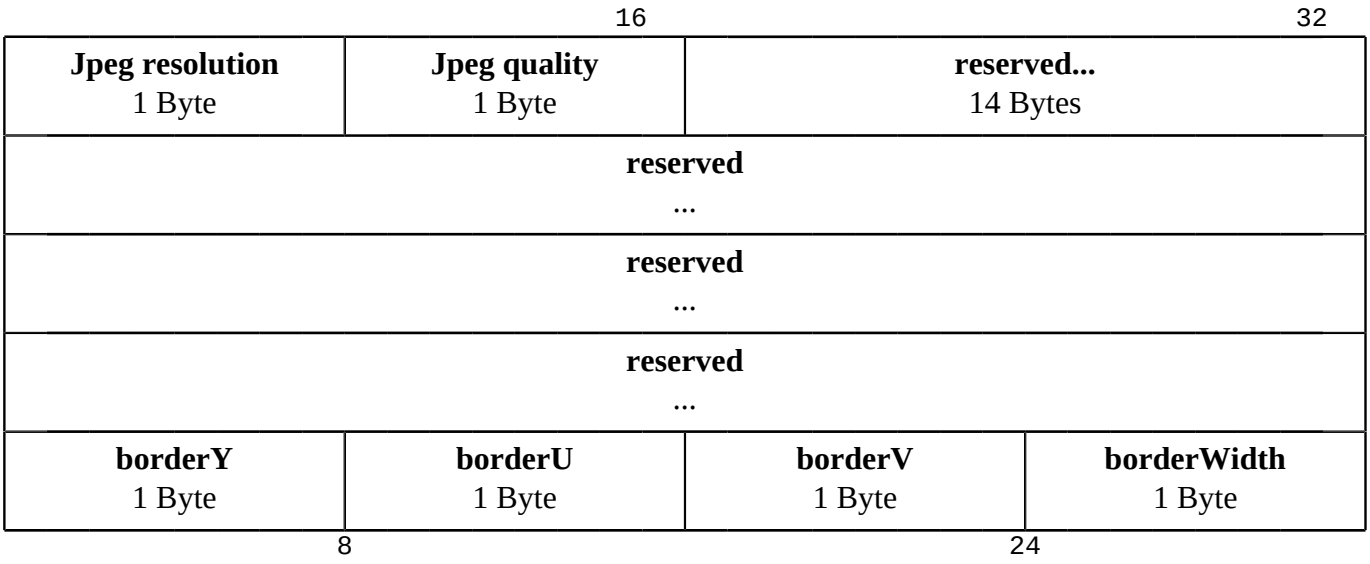
[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x099e	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Reads a jpeg that can be configured by the optional input payload.

Request Payload Structure



Jpeg resolution

QCIF	0
CIF	1
CIF	2
4CIF	3

Jpeg quality

Quality	0 - 100	0 is best.

Response Payload Structure

© Copyright 2001-2021 Bosch Security Systems

JPEG picture N Bytes

© Copyright 2001-2021 Bosch Security Systems

2.353 CONF_JPEG_BANDWIDTH_KBPS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x061d	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the jpeg bandwidth (in KBPS) of selected preset	
Write	t_dword	service	Sets the jpeg bandwidth (in KBPS) of selected preset	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.354 CONF_JPEG_STREAM_FRAME_RATES

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c81	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

Only read is supported.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: Global frame rates, independent from the resolution

A proposed list of max. frame rates to be used as selection for the max. frame rate of the JPEG stream.

16	32
Length 2 Bytes	**Tag = 0x0001** 2 Bytes
Global frame rates, independent from the resolution (p_octet) *Length* - 4 Bytes	
8	24

Global frame rates, independent from the resolution

Frame Rate [0] 4 Bytes
...
Frame Rate [N] 4 Bytes

Frame Rate

Frame rate in mHz

© Copyright 2001-2021 Bosch Security Systems

2.355 CONF_JPEG_STREAM_SETUP

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad5	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the configuration of the JPEG encoder; resolution defined by the id may vary please use CONF_JPEG_STREAM_SETUP_OPTIONS_V to query which ids are supported and there properties	
Write	p_octet	service	Write the configuration of the JPEG encoder; resolution defined by the id may vary please use CONF_JPEG_STREAM_SETUP_OPTIONS_V to query which ids are supported.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16	32
Id 4 Bytes	
FPS in mHz 4 Bytes	
Quality 4 Bytes	
8	24

Quality

Quality	0 - 100
Auto	0
Worst	1
Best	100

© Copyright 2001-2021 Bosch Security Systems

2.356 CONF_JPEG_STREAM_SETUP_OPTIONS_VERBOSE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c00	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

Allows to query information about the ids and the allowed ids configurable via CONF_JPEG_STREAM_SETUP. Only read is supported.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0: Setup id with width and height

Map CONF_JPEG_STREAM_SETUP ids to jpeg width and height.

16	32
Length = 0x0014 2 Bytes	**Tag = 0x0000** 2 Bytes
Setup id with width and height (p_octet) 16 Bytes	
8	24

Setup id with width and height

16	32
Id 4 Bytes	
Width 4 Bytes	
Height 4 Bytes	
Reserved 4 Bytes	
8	24

Id

Id supported in CONF_JPEG_STREAM_SETUP command

Width

Jpeg width, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

Height

Jpeg height, 0 -> device automaticaly decides which resolution delivers the best performance in conjuction with the current encoder configuration

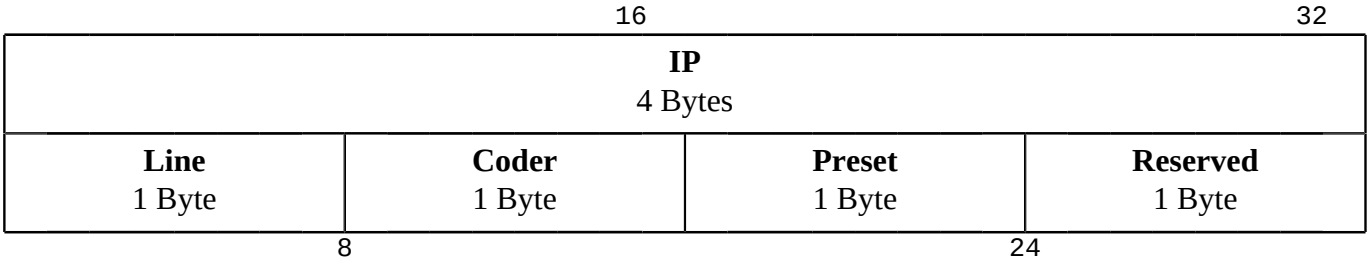
© Copyright 2001-2021 Bosch Security Systems

2.357 CONF_KBD_CONFIG_CAMERA

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a31	camera number on keyboard	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read configuration of camera numbers on keyboard	
Write	p_octet	service	Configure camera numbers on keyboard; assign parameters to a number (given by num)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure



IP

Encoder/Camera IP

Line

Video input line

Coder

Relative coder number (relative to line)

Preset

(Dome-) Preset Position (optional). (Every Preset of a Dome can be treated as a seperate camera). Set to '0' for don't care.

© Copyright 2001-2021 Bosch Security Systems

2.358 CONF_KBD_CONFIG_CAMERA_STR

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba3	camera number on keyboard	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read configuration of camera numbers on keyboard	
Write	p_octet	service	Configure camera numbers on keyboard; assign parameters to a number (given by num)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16			32
Line 1 Byte	**Coder** 1 Byte	**Preset** 1 Byte	**URL Length** 1 Byte
URL *URL Length* Bytes			
8		24	

Line

Video input line

Coder

Relative coder number (relative to line)

Preset

(Dome-) Preset Position (optional). (Every Preset of a Dome can be treated as a seperate camera). Set to '0' for don't care.

URL Length

Length of the following url (actual limit is 80).

URL

Encoder/Camera URL

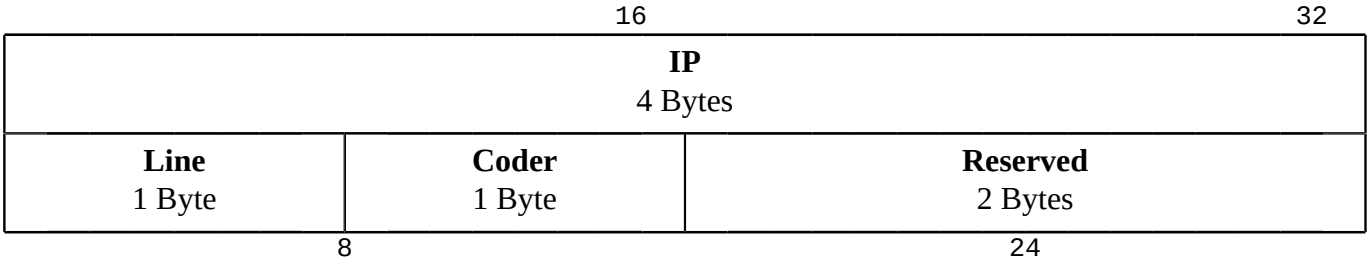
© Copyright 2001-2021 Bosch Security Systems

2.359 CONF_KBD_CONFIG_MONITOR

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a32	monitor number on keyboard	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read configuration of monitor numbers on keyboard	
Write	p_octet	service	Configure monitor numbers on keyboard; assign parameters to a number (given by num)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure



IP

Decoder/Monitor IP

Line

Video output line

Coder

Relative coder number (relative to line)

© Copyright 2001-2021 Bosch Security Systems

2.360 CONF_KBD_CONFIG_MONITOR_STR

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba4	monitor number on keyboard	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read configuration of monitor numbers on keyboard	
Write	p_octet	service	Configure monitor numbers on keyboard; assign parameters to a number (given by num)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16			32
Line 1 Byte	**Coder** 1 Byte	**URL Length** 1 Byte	**Reserved** 1 Byte
URL *URL Length* Bytes			
8		24	

Line

Video output line

Coder

Relative coder number (relative to line)

URL Length

Length of the following url (actual limit is 80).

URL

Decoder/Monitor URL

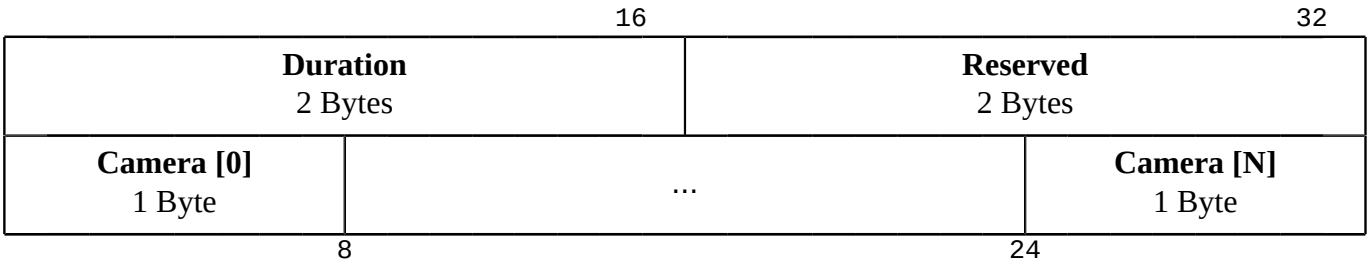
© Copyright 2001-2021 Bosch Security Systems

2.361 CONF_KBD_CONFIG_SALVO

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3e	salvo number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read configuration of salvos for keyboard	
Write	p_octet	service	Configure salvos used for keyboard; assign parameters to a salvo number (given by num)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure



Duration

Duration of one salvo position in seconds

Camera

Camera number for salvo

Note: Camera number has to be specified using CONF_KBD_CONFIG_CAMERA

© Copyright 2001-2021 Bosch Security Systems

2.362 CONF_KBD_CONNECT_PARAMS

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a33	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read parameters for connections established via keyboard	
Write	p_octet	service	Set parameters for connections established via keyboard	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16	32
Flags 2 Bytes	**Reserved** 2 Bytes
8	24

Flags

	Mask	Name	Description
Bit 1	0x0002	Request Audio RX	Speak at decoder side
Bit 0	0x0001	Request Audio TX	Listen at decoder side

© Copyright 2001-2021 Bosch Security Systems

2.363 CONF_KBD_KEY_LABEL

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a44	key number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read a label of a key used for the alarm task editor. The label is shown on the keyboard display.	
Write	p_string	service	Deposit a label for a key used for the alarm task editor. The label is shown on the keyboard display.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.364 CONF_KBD_PASSWORD

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a69	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Get the password (scrambled) of the keyboard	
Write	p_string	service	Deposit a password for the keyboard	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.365 CONF_KBD_PASSWORD_CAMERA

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a34	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Get the password (scrambled) for all cameras used on keyboard	
Write	p_string	service	Deposit a password for all cameras used on keyboard	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.366 CONF_KBD_SET_ALARM

[API: Keyboard]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a68	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	minimal	Send alarm to keyboard	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.367 CONF_LED_BLINKING

[API: debug]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x013d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Always 1	
Write	f_flag	minimal	1=power LED will flash for 7 sec	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.368 CONF_LED_CAPS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read power/traffic LED capabilities: 0=no capabilities, 1=disengageable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.369 CONF_LIST_APPS

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

Description

Command CONF_LIST_APPS returns all installed Apps on a SAST system.

Payload Structure

App Entry

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 1: App Entry

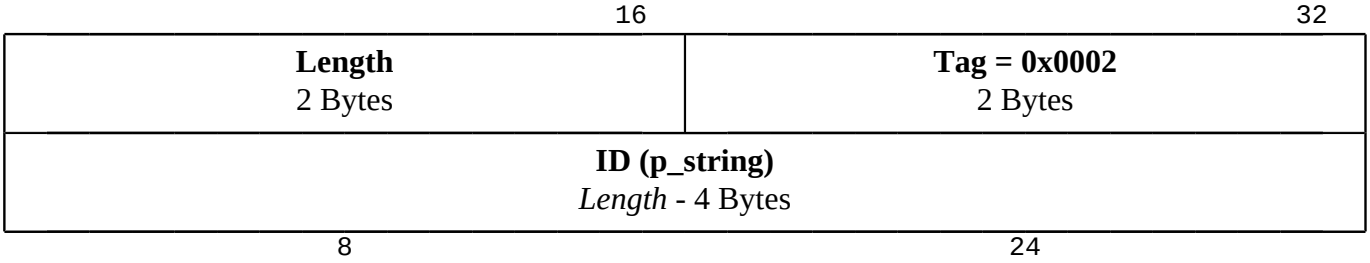
The payload of this tag entry contains further tagged values described below.

16	32
Length = 0x0020 2 Bytes	**Tag = 0x0001** 2 Bytes
8	24

© Copyright 2001-2021 Bosch Security Systems

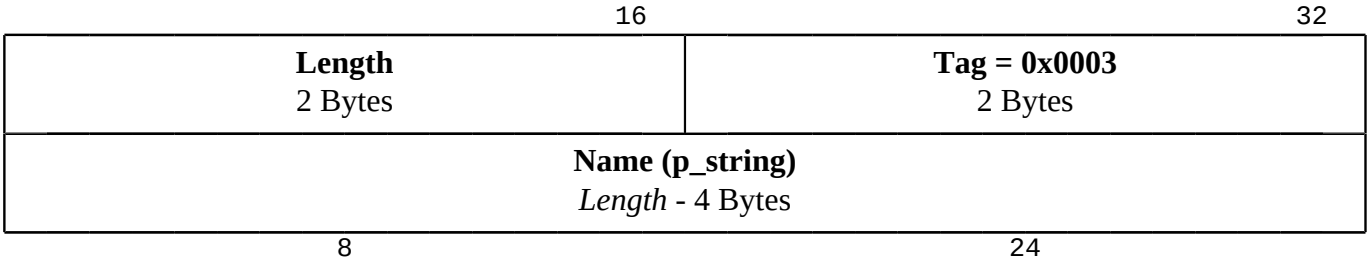
Tag 2: ID (Required)

app id



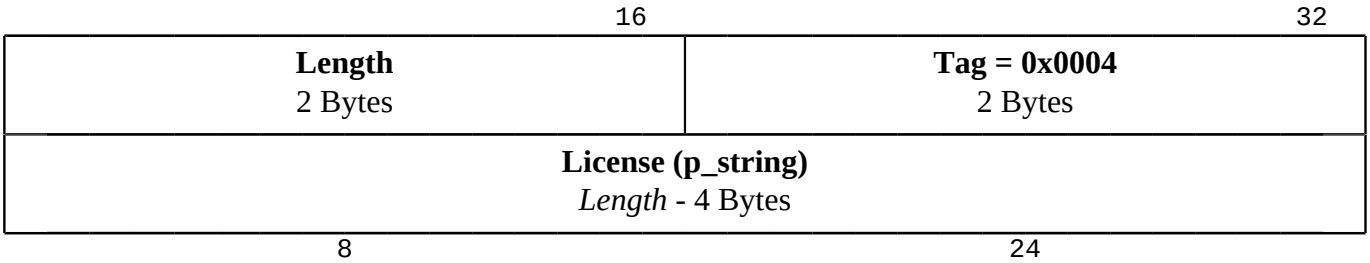
Tag 3: Name

app name



Tag 4: License

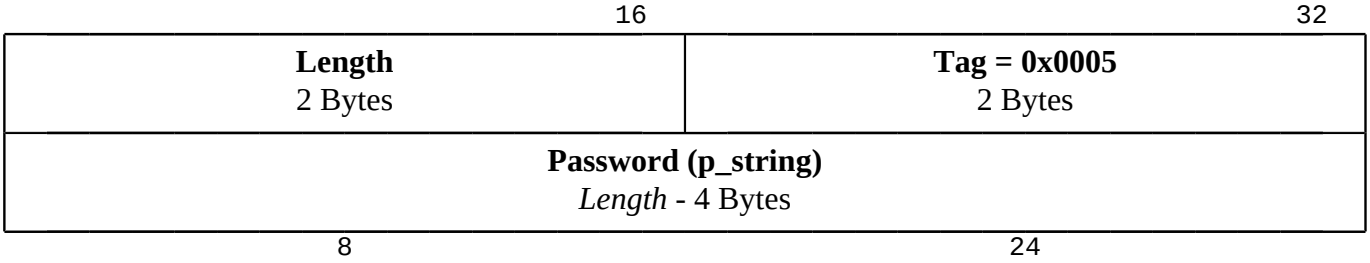
app license



© Copyright 2001-2021 Bosch Security Systems

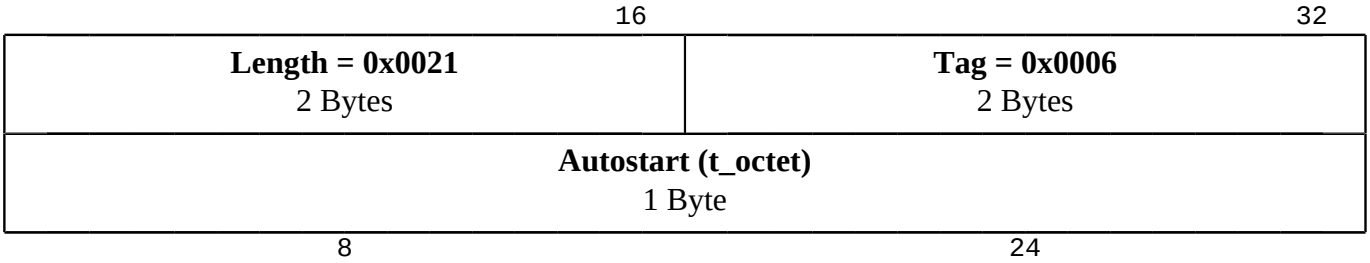
Tag 5: Password

app version



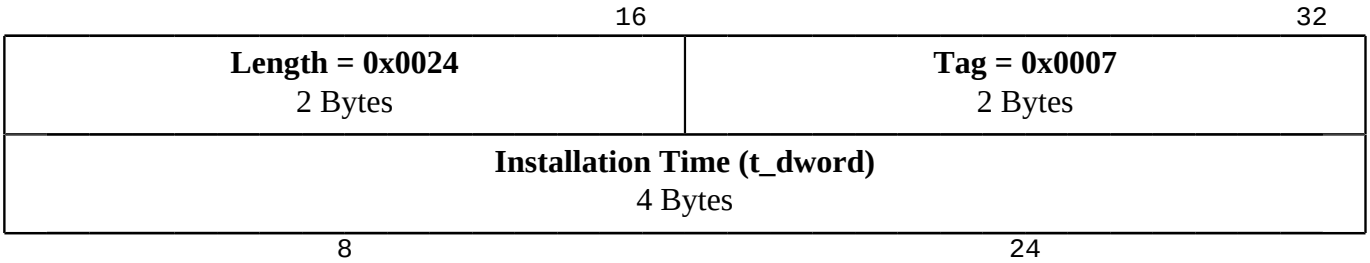
Tag 6: Autostart

autostart enabled or not



Tag 7: Installation Time

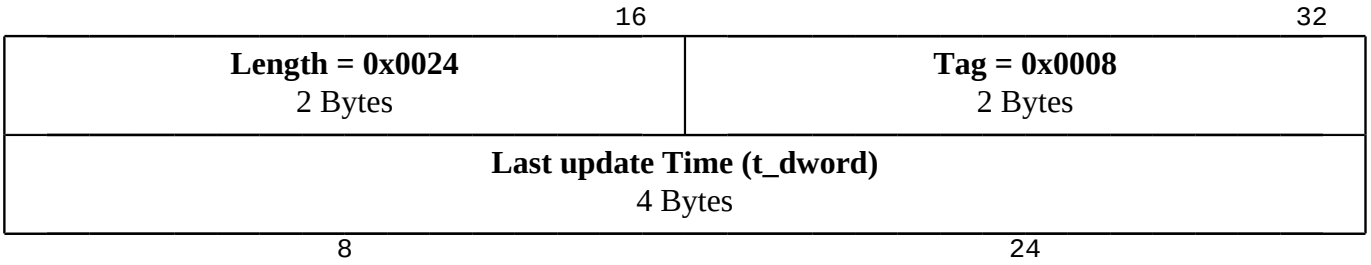
installation time in utc



© Copyright 2001-2021 Bosch Security Systems

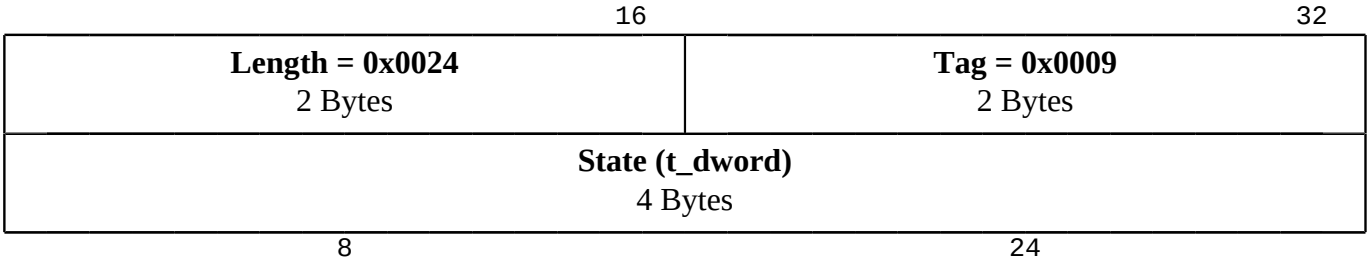
Tag 8: Last update Time

last update time time in utc



Tag 9: State

state of the app (0: inactive, 1: active)



© Copyright 2001-2021 Bosch Security Systems

2.370 CONF_LIST_OF_VIPROC_SCENES

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a41	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	This command works only for Gen4 autodomes. It returns the list of defined scenes together with their names. Thereby the first byte specifies the scene number, followed by the length L of the scene name. The next L bytes belong to the scene name in unicode without 0 termination. This structure repeats until either the payload ends or the scene number is zero.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.371 CONF_LLDP_ALLOCATED_POWER_TOTAL

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c90	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the poe power allocated for the device in total (adder+device itself) in tenths of Watts	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.372 CONF_LLDP_POWER_ADDER

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c92	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the power volume that the camera should request via LLDP in addition to its own requirements; value is in tenths of watts; e.g. 9.9 watts results in a value of 99;	
Write	t_word	service	Set the power volume that the camera should request via LLDP in addition to its own requirements; value is in tenths of watts; e.g. 9.9 watts results in a value of 99; when the adder value is different from 0, the device identifies as type 2 PD of class 4; when adder value is 0 and the device specific requirement is below 13 W, the device signals type 1 PD of the respective class; currently, the total power requested by a camera can not exceed the 25.5W specified as maximum in IEEE802.3at-2009	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.373 CONF_LLDP_REQUESTED_POWER_CAM

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c8f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the poe power requested by device (for the device alone) in tenths of Watts	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.374 CONF_LLDP_REQUESTED_POWER_TOTAL

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c8e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the poe power requested by device in total (adder+device itself) in tenths of Watts	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.375 CONF_LOADED_VIPROC_CONFIG

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3d	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the currently loaded config id.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.376 CONF_LOCAL_HTTP_PORT

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0954	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	always_legacy	Get the local HTTP port for browser access	
Write	t_word	service	Set the local HTTP port for browser access (NOTE: it is not allowed to turn off both, HTTP and HTTPS, at the same time)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

apply with APPLY_NETWORK_SETTINGS	0x8000

© Copyright 2001-2021 Bosch Security Systems

2.377 CONF_LOCAL_HTTPS_PORT

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a0e	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	always_legacy	Get the local HTTPS port for browser access	
Write	t_word	service	Set the local HTTPS port for browser access (NOTE: it is not allowed to turn off both, HTTP and HTTPS, at the same time)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

apply with APPLY_NETWORK_SETTINGS	0x8000

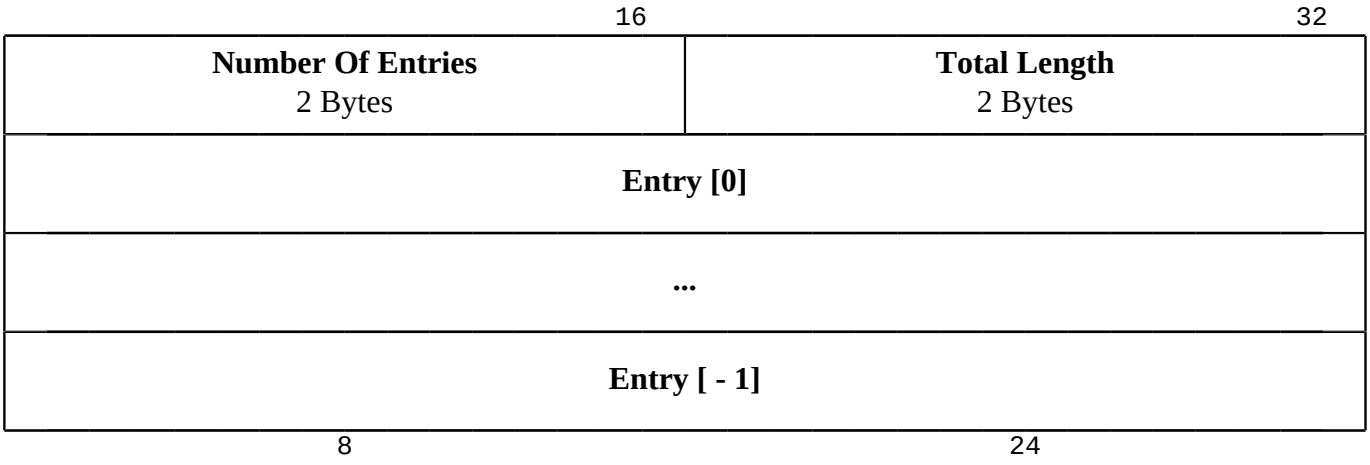
© Copyright 2001-2021 Bosch Security Systems

2.378 CONF_LOGIN_LIMITER_MESSAGE

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb3	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Number Of Entries

Number of Entries following

Total Length

Total Length in Bytes, including all entries and this header

Entry

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

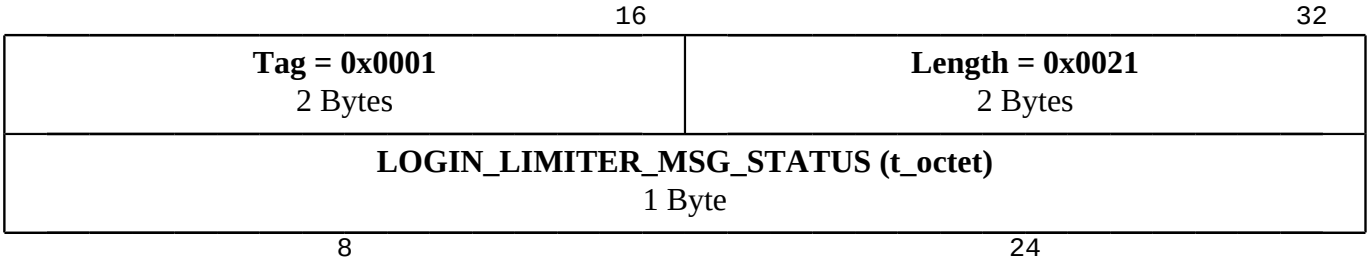
Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

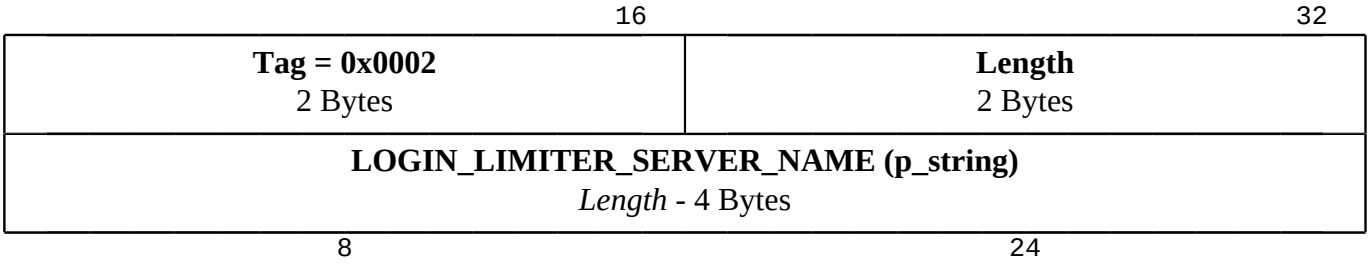
Tag 1: LOGIN_LIMITER_MSG_STATUS



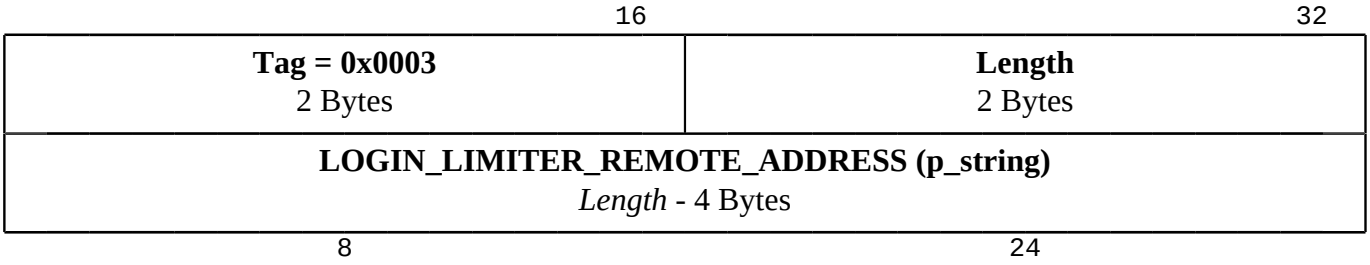
Values:

Fail Msg	0
Success Msg	1

Tag 2: LOGIN_LIMITER_SERVER_NAME

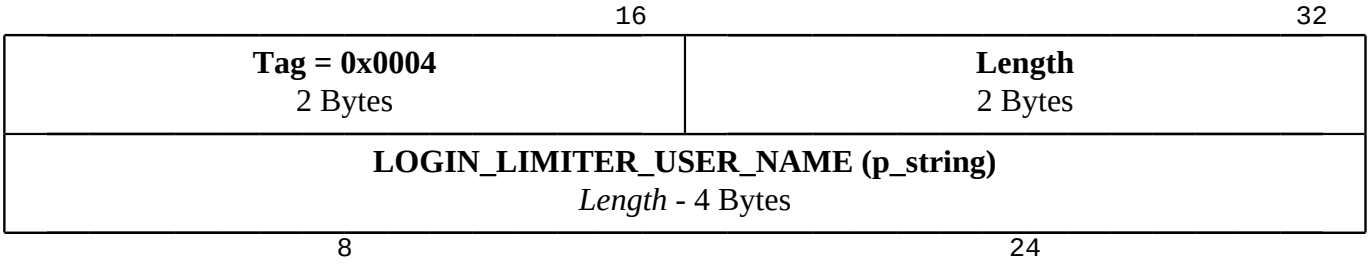


Tag 3: LOGIN_LIMITER_REMOTE_ADDRESS



© Copyright 2001-2021 Bosch Security Systems

Tag 4: LOGIN_LIMITER_USER_NAME



© Copyright 2001-2021 Bosch Security Systems

2.379 CONF_LOGIN_PAGE_TEXT

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d2a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	always	Read the (optional) text which is shown on the login page	
Write	p_unicode	service	Set an (optional) text which is shown on the login page (max 1000 unicode characters + null delimiter)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.380 CONF_LOGO

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0939	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	0 := show \logo.h263\" from toolkit 1:= no logo 2:= screen saver	
Write	t_octet	service	0 := show \logo.h263\" from toolkit 1:= no logo 2:= screen saver	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.381 CONF_LOGO_STAMP_VAL

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c10	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	t_octet	user	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.382 CONF_LOW_AMBIENT_LIGHT_THRESHOLD

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c46	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Get the normalized low ambient light level threshold (value range: 0-1000; typical 50); Only available on certain devices; To obtain the current level, see CONF_AMBIENT_LIGHT_LEVEL	
Write	t_dword	service	Set the normalized low ambient light threshold (value range: 0-1000; typical 50); Only available on certain devices; To obtain the current level, see CONF_AMBIENT_LIGHT_LEVEL	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.383 CONF_MAC_ADDRESS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00bc	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy	Read out the systems MAC address.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Ethernet	1
WLAN	2
Bluetooth	3

© Copyright 2001-2021 Bosch Security Systems

2.384 CONF_MANAGE_APPS

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf7	action: 1: activate, 2: deactivate, 3: uninstall	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	AppID	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.385 CONF_MANAGING_VRM

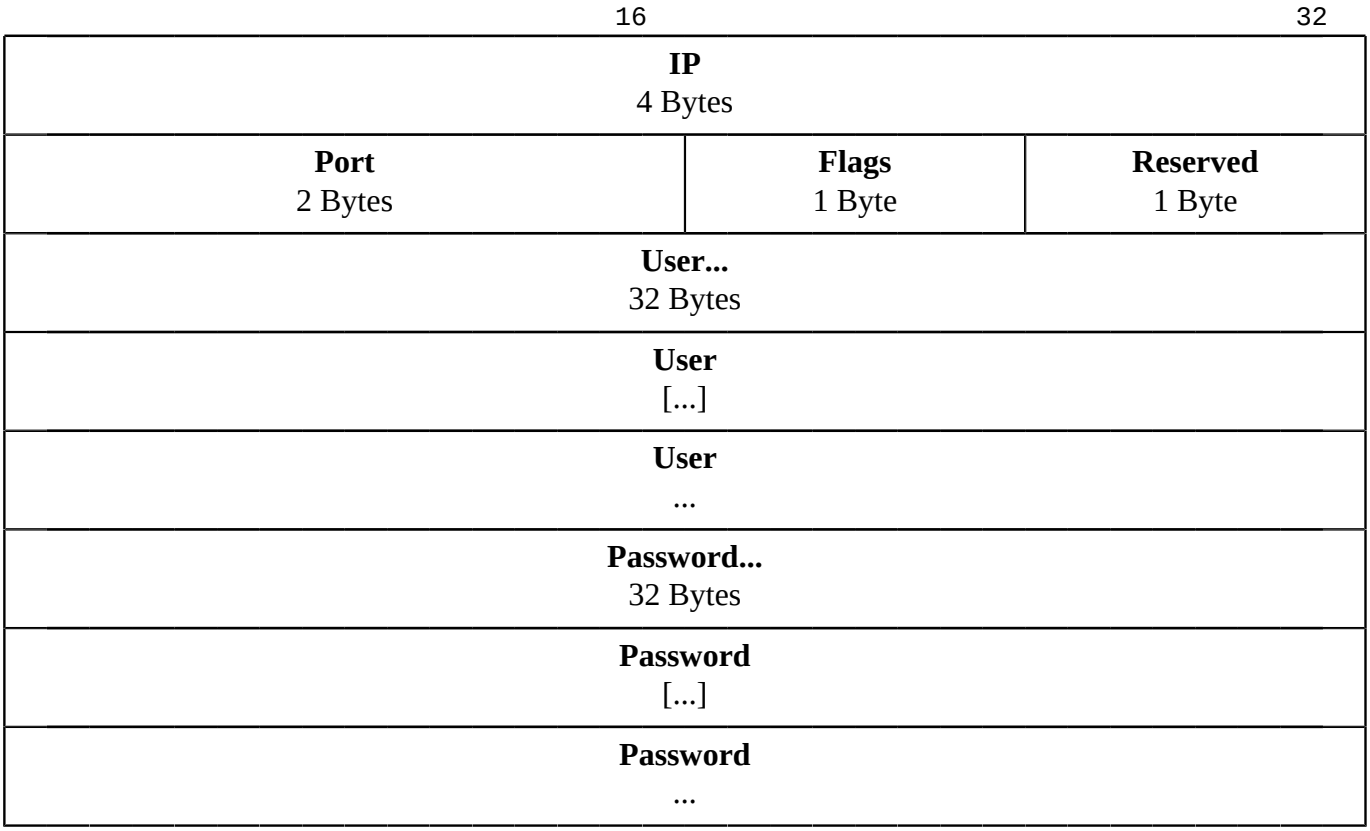
[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aeb	1 - primary recording, 2 - secondary recording	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to set or read the IP address, port and user and to set the password of the managing VRM and the backup VRM. The command cannot be used for reading the VRM password, it will return the string "********" instead.

Payload Structure



© Copyright 2001-2021 Bosch Security Systems

Backup IP 4 Bytes	
Backup Port 2 Bytes	**Reserved** 2 Bytes
8	24

IP

IP address of the managing VRM

Port

Port of the managing VRM

Flags

	Mask	Name
Bit 0	0x01	USE_SSL

User

VRM user: Max 31 ASCII character string with zero termination. If the user is shorter than 31 characters, the remaining bytes need to be filled up with 0x00 values.

Password

VRM password: Max 31 ASCII character string with zero termination. If the password is shorter than 31 characters, the remaining bytes need to be filled up with 0x00 values.

Backup IP

IP address of the backup VRM

Backup Port

Port of the backup VRM

© Copyright 2001-2021 Bosch Security Systems

2.386 CONF_MANIPULATION_ALARM_STATE

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af0	manipulation alarm nbr	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=manipulation alarm off; 1=manipulation alarm on (notice: for supervised alarm inputs the according CONF_INPUT_PIN_STATE also indicates an alarm in this case)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.387 CONF_MANUFACTURER_NAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d0b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Returns the manufacturer name	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.388 CONF_MASTERPWD_CHALLENGE

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x013f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	always	Generates an password challenge which has to be passed to the operator for password reset	
Write	p_string	always	Provide the generated unlock key from operator to reset passwords. The key has to be in hex string repesentation without 0x in front.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.389 CONF_MASTERPWD_CHALLENGE_STATE

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c21	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always	Retrieves status information about the masterpwd challenge (4 bytes validity in seconds, 4 bytes lock state)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.390 CONF_MAX_GOP_LENGTH_VALUE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b9d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the max gop length for recording	
Write	t_dword	service	Set the max gop length for recording, 0 means back to default	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.391 CONF_MAX_NBR_OF_ENC_STREAMS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x029e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of encoder streams	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.392 CONF_MAX_RECORDING_RETENTION_TIME

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b5b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the max recording retention time for a camera (in seconds), see detailed description	
Write	p_octet	service	Set the max recording retention time for a camera (in seconds), see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command sets the max retention time for recording identified by the rec idx and the camera in the num parameter starting from 1. For read direction a inpayload is required with same format as described above, except the max retention time field can be ignored. The max retention time will be stored as absolute local time in seconds since 2000 in each span unit header. It will be used to clear a recording span when this time expires. If a max retention time is configured unequal 0 and less than maximum, the recording time on a span is limited to a one day time span. If this limit is reached by the recording, a span switch will be triggered.

Payload Structure

16	32
Max retention time 4 Bytes	
Rec Idx 2 Bytes	**Reserved** 2 Bytes
8	24

Max retention time

Max retention in seconds, 0 or values bigger than about 32 years (value >= 1009152000) means maximum, in case of in payload for read direction, this field can be ignored and treaded as reserved field

Rec Idx

Recording Index

Primary Recording	1	
Secondary Recording	2	Secondary recording (active low)

© Copyright 2001-2021 Bosch Security Systems

2.393 CONF_MEDIA_SOCKETS_COMPLETE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xffc7	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.394 CONF_MIN_TLS_VERSION

[API: cert store]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c56	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the TLS version minimal required to connect to the device. Word coding: bit 15-8 mayor version, bit 7-0 minor version. E.g. 0x0102 is TLSv1.2	
Write	t_word	service	Set the TLS version minimal required to connect to the device. Word coding: bit 15-8 mayor version, bit 7-0 minor version. E.g. 0x0102 is TLSv1.2	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.395 CONF_MODE_AUTO_TRACKER

[API: autotracker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b40	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the autotracker mode (Modi: OFF=0, AUTO=1, CLICK=2).	
Write	t_octet	user	Sets the autotracker mode (Modi: OFF=0, AUTO=1, CLICK=2). The CLICK mode has to be rewritten after 10sec; otherwise, the mode shall switch back to AUTO	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.396 CONF_MONITOR_NAME

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x028a	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Get the video monitor name	
Write	p_unicode	service	Set the video monitor name (max 32 unicode characters)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.397 CONF_MOTION_ALARM_STATE

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01c3	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=motion alarm off; 1=motion alarm on (at elast one of the CONF_VIPROC_ALARM alarmbit (except the video loss alarm) is set	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.398 CONF_MPEG_AUDIO_SAMPLING_FREQ

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0932	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Mpeg audio sampling frequency (Hz)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.399 CONF_MPEG4_AVC_BITRATE_OPTIMIZATION

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c37	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] 0 = OFF, 1 = normally no visible effects (tuned for low amount of artefacts), 2 = very low amount of visable artefacts 3 = no artefacts in most scenes (best balance between bitrate and quality), 4 = some visible artefacts, but very low bitrate 5 = artefacts in many scenes but very low bitrate, Byte [1-7] reserved; if activated (byte 1 not 0) CONF_MPEG4_BANDWIDTH_KBPS will only have an influence if CONF_VIDEO_BITRATE_AVERAGING_PE is not 0 otherwise device will be in a varibale bitrate mode (only a max bitrate can be configured via CONF_MPEG4_BANDWIDTH_KBPS_SOFT_	
Write	p_octet	service	Byte[0] 0 = OFF, 1 = normally no visible effects (tuned for low amount of artefacts), 2 = very low amount of visable artefacts 3 = no artefacts in most scenes (best balance between bitrate and quality), 4 = some visible artefacts, but very low bitrate 5 = artefacts in many scenes but very low bitrate, Byte [1-7] reserved; if activated (byte 1 not 0) CONF_MPEG4_BANDWIDTH_KBPS will only have an influence if CONF_VIDEO_BITRATE_AVERAGING_PE is not 0 otherwise device will be in a varibale bitrate mode (only a max bitrate can be configured via CONF_MPEG4_BANDWIDTH_KBPS_SOFT_	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.400
CONF_MPEG4_AVC_BITRATE_OPTIMIZATION_OPTIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c49	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] basicSupport; Byte[1] reserved; Byte[2-7] reserved	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.401 CONF_MPEG4_AVC_CABAC

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aa6	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets cabac for the selected preset (0=off; 1=on)	
Write	t_octet	service	Sets cabac for the selected preset (0=off; 1=on)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.402 CONF_MPEG4_AVC_CHROMA_QUANT_OFF

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x061a	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Get the chroma quantisation offset (-12...12)	
Write	t_int	service	Set the chroma quantisation offset (-12...12)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.403 CONF_MPEG4_AVC_CODING_MODE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a45	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the coding mode for the selected preset (0=frame; 1=field; 2=makro block adaptive ff; 3=picture adaptive ff)	
Write	t_octet	service	Sets the coding mode for the selected preset (0=frame; 1=field; 2=makro block adaptive ff; 3=picture adaptive ff)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.404 CONF_MPEG4_AVC_DEBLOCKING_ALPHA

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0618	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Get the alpha H264 deblocking coefficent (-5...5)	
Write	t_int	service	Set the alpha H264 deblocking coefficent (-5...5)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.405 CONF_MPEG4_AVC_DEBLOCKING_BETA

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0619	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Get the beta H264 deblocking coefficent (-5...5)	
Write	t_int	service	Set the beta H264 deblocking coefficent (-5...5)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.406 CONF_MPEG4_AVC_DEBLOCKING_ENABLE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0617	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Enables/disables the H264 deblocking filter	
Write	f_flag	service	Enables/disables the H264 deblocking filter	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.407 CONF_MPEG4_AVC_DELTA_IPQUANT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0621	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Gets the difference (-10..+10) between I and P-Frame quantization for selected preset	
Write	t_int	service	Sets the difference (-10.. +10) between I- and P-Frame quantization for selected preset	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.408 CONF_MPEG4_AVC_GOP_STRUCTURE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a94	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the GOP structure of the selected preset (0=IP; 1=IBP; 2=IBBP; 3=IBBRBP)	
Write	t_octet	service	Sets the GOP structure of the selected preset (0=IP; 1=IBP; 2=IBBP; 3=IBBRBP)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.409 CONF_MPEG4_AVC_I_FRAME_QUANT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0615	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Obsolete	
Write	t_dword	service	Obsolete	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.410 CONF_MPEG4_AVC_P_FRAME_QUANT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0616	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Obsolete	
Write	t_dword	service	Obsolete	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.411 CONF_MPEG4_AVC_P_FRAME_QUANT_MIN

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0620	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the P frame min quantizer (9-51, 0=auto) for selected preset	
Write	t_dword	service	Sets the P frame min quantizer (0=auto, 9...51) for selected preset, sessionID to address session based video params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.412 CONF_MPEG4_AVC_QUANT_ADJ_REGION_1

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0624	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Gets the quality difference (H.264 QP Units; -51..+51) between normal and background regions; use CONF_ROI_OPTIONS to query options if only positiv offsets are allowed offsets between 0..+51 can be read	
Write	t_int	service	Sets the quality difference (H.264 QP units; -51..+51) between normal and background regions; use CONF_ROI_OPTIONS to query options if only positiv offsets are allowed offsets between 0..+51 can be set	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.413 CONF_MPEG4_AVC_QUANT_ADJ_REGION_2

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0625	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Gets the quality difference (H.264 QP Units; -51..+51) between normal and background regions; use CONF_ROI_OPTIONS to query options if only positiv offsets are allowed offsets between 0..+51 can be read	
Write	t_int	service	Sets the quality difference (H.264 QP units; -51..+51) between normal and background regions; use CONF_ROI_OPTIONS to query options if only positiv offsets are allowed offsets between 0..+51 can be set	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.414 CONF_MPEG4_BANDWIDTH_KBPS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0607	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Reads the bandwidth of selected preset	
Write	t_dword	service	Sets the bandwidth of selected preset, sessionID to address session based video params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.415
CONF_MPEG4_BANDWIDTH_KBPS_SOFT_LIMIT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0612	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Reads the bandwidth soft limit (in KBPS) of selected preset	
Write	t_dword	service	Sets the bandwith soft limit (in KBPS) of selected preset, sessionID to address session based video params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.416 CONF_MPEG4_CURRENT_PARAMS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0600	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the current profile preset number for the specified encoder (NumDesc)	
Write	t_dword	service	Sets the video encoder (given by NumDesc) to a preset (payload). If recording is running and the iframe distance is 0, the iframe distance of the preset will be patched to a recording suitable value. For transcoder: if the request contains the replay session id and no num descriptor, the device will assign the preset to the transcoder of the replay session. If the preset is empty the transcoder will be disabled. For exclusive (dual) stream instances: if the request contains the session id and the num descriptor is 0, the previously configured preset with its session based (non-permanent) modifications will be applied to the encoder.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.417
CONF_MPEG4_CURRENT_PARAMS_REL_CODER

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x061c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the current profile preset number for the specified encoder (relative to line).	
Write	p_octet	service	Sets the video encoder (relative to line) to a preset.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16	32
Line 1 Byte	**Coder** 1 Byte	**Coding Capabilities** 2 Bytes
Preset 1 Byte	**Reserved** 3 Bytes
8	24

Line

Video input line

Coder

Relative coder number (relative to line)

Coding Capabilities

Coding capabilities of encoder. All coding capabilities are one or multiple of:

	Mask	Name
Bit 6	0x0040	H.264
Bit 3	0x0008	Mpeg 2
Bit 2	0x0004	Mpeg 4
Bit 1	0x0002	H.263

Preset

Number of the encoder preset profile

© Copyright 2001-2021 Bosch Security Systems

Note: When reading the preset profile, the according byte in the payload will be set.

© Copyright 2001-2021 Bosch Security Systems

2.418
CONF_MPEG4_CURRENT_PARAMS_TRANSCODER

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b4e	has to be zero	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	SessionID references the replay session; 0:= no transcoding; 1-8: video preset of the transcoder	
Write	t_dword	user	Assign preset n (1, ..N) to transcoder session; preset 0 means no transcoding	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.419 CONF_MPEG4_DEFAULTS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0601	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	service	Set the selected preset params to the default values (if the recording is running and the preset in on the recording schedule, the preset will keep its old iframe distance)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.420 CONF_MPEG4_FRAME_SKIP_RATIO

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0606	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets Mpeg4 frame skip ratio (1=all frames, ...)	
Write	t_dword	service	Set Mpeg4 frame skip ratio (1=all frames, ...), sessionID to address session based video params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.421 CONF_MPEG4_INTRA_FRAME_DISTANCE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0604	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the intra frame distance of selected preset	
Write	t_dword	service	Sets the intra frame distance of selected preset (not supported while recording is running or configured to active and the preset is used on the recording schedule), sessionID to address session based video params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.422 CONF_MPEG4_INTRA_FRAME_REQUEST

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0605	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag t_dword	minimal	Request a video intra frame, SessionID is mandatory	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.423 CONF_MPEG4_NAME

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0602	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the name of the preset given by the Numeric Descriptor	
Write	p_string	service	Sets the name of the preset number given by the Numeric Descriptor	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.424 CONF_MPEG4_PARAMS_MAX_NUM

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0614	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of MPEG4 presets	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.425 CONF_MPEG4_RESOLUTION

[API: encoder]

© Copyright 2001-2021 Bosch Security Systems

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0608	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the spatial resolution for selected preset. Possible resolutions depending on the device platform (CPPx). They can be queried with the command CONF_ENC_PROFILE_RESOLUTION_OPTI Info: for devices where CONF_ENC_BASE_OPERATION_MODE_TY is '1' this command only applies for SD resolutions (0=QCIF, 1=CIF, 2=2CIF, 3=4CIF, 4=(1/2 D1), 5=(2/3D1), 6=QVGA, 7=VGA, 8=WD144 (256x144), 9=WD288 (512x288), 10=WD432 (768x432), 18=WD1 (960H)). HD resolutions then need to be set via CONF_VIDEO_H264_ENC_BASE_OPERATI For devices where CONF_ENC_BASE_OPERATION_MODE_TY is '0' (no base operation modes) or '2' (upper limit) then this command also applies for HD resolutions. The t_dword payload is then as follows: upper 2 Bytes of DWORD = width, lower 2 Bytes = height. (e.g. 1280x720 -> t_dword = 0x050002D0 = 83886800dec); HD resolutions on transcoder: 720 (up to 720x1280), 1080 (up to 1080x1920), 2160(up to 2160x3840); SessionID to address session based video params	
Write	t_dword	service	Sets the spatial resolution for selected preset. Possible resolutions depending on the device platform (CPPx). They can be queried with the command CONF_ENC_PROFILE_RESOLUTION_OPTI Info: for devices where CONF_ENC_BASE_OPERATION_MODE_TY is '1' this command only applies for SD resolutions (0=QCIF, 1=CIF, 2=2CIF, 3=4CIF, 4=(1/2 D1), 5=(2/3D1), 6=QVGA, 7=VGA, 8=WD144 (256x144), 9=WD288 (512x288), 10=WD432 (768x432), 18=WD1 (960H)). HD resolutions then need to be set via	

© Copyright 2001-2021 Bosch Security Systems

	CPP6/CPP7/CPP7.3	CPP13
Available	yes	yes

© Copyright 2001-2021 Bosch Security Systems

2.426 CONF_MQTT_SETTINGS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d0a	MQTT instance	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	MQTT Settings in a tagged format	
Write	p_octet	service	MQTT Settings in a tagged format	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

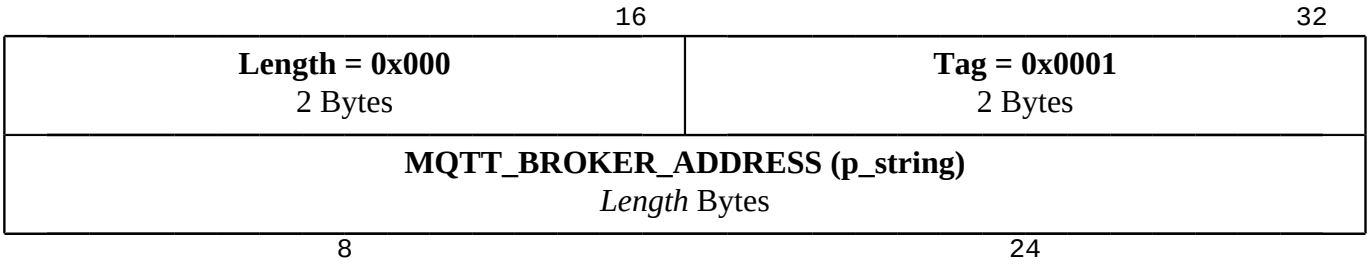
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: MQTT_BROKER_ADDRESS

Address of the remote broker. Example 'mqtt



© Copyright 2001-2021 Bosch Security Systems

2.427 CONF_MTU_SIZE

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b3d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Mtu size (0 means default)	
Write	t_dword	service	Mtu size (0 means default)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.428 CONF_MULTICAST_AUDIO_GROUP_IP

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd1	audio coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the multicast audio group address	
Write	t_dword	service	Set the multicast audio group address (range: 224.0.0.10 .. 239.255.255.255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.429 CONF_MULTICAST_AUDIO_GROUP_IP_STR

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd2	audio coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the multicast audio group address	
Write	p_string	service	Set the multicast audio group address (range: 224.0.0.10 .. 239.255.255.255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.430 CONF_MULTICAST_AUDIO_PORT

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01b2	audio coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the multicast audio UDP port number	
Write	t_word	service	Set the multicast audio UDP port (even port numbers only)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.431 CONF_MULTICAST_AUDIO_PORT_STR

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01b5	audio coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the multicast audio UDP port number	
Write	p_string	service	Set the multicast audio UDP port (even port numbers only)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.432 CONF_MULTICAST_TTL

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0267	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the TTL (time to live) in IP-Header for multicast packets	
Write	t_octet	service	Set the TTL (time to live) in IP-Header for multicast packets	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.433 CONF_MULTICAST_VIDEO_GROUP_IP

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01b1	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the multicast video group address	
Write	t_dword	service	Set the multicast video group address (range: 224.0.0.10 .. 239.255.255.255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.434 CONF_MULTICAST_VIDEO_GROUP_IP_STR

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01b4	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the multicast video group address	
Write	p_string	service	Set the multicast video group address (range: 224.0.0.10 .. 239.255.255.255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.435 CONF_MULTICAST_VIDEO_PORT

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0286	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the multicast video UDP port number	
Write	t_word	service	Set the multicast video UDP port (even port numbers only)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.436 CONF_MULTICAST_VIDEO_PORT_STR

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0288	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the multicast video UDP port number	
Write	p_string	service	Set the multicast video UDP port (even port numbers only)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.437 CONF_MUTE_MEDIA_CHANNEL

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff14	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	live	Mutes/unmutes a media channel: Bit 1: video, Bit2: audio, Bit3: meta; session id is needed, don't mute if media channel is in multicast	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.438 CONF_NAME_STAMP_VAL

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0084	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	t_octet	user	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.439 CONF_NBR_OF_ACCOUNTS

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b62	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the max number of accounts	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.440 CONF_NBR_OF_ALARM_IN

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01db	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed alarm input contacts (possible masterswitch included)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.441 CONF_NBR_OF_ALARM_OUT

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01dc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed alarm output contacts	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.442 CONF_NBR_OF_ALTERNATIVE_ALARM_IPS

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0303	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the number of available alarm ip addresses (total presets)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.443 CONF_NBR_OF_AUDIO_IN

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01d8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of audio inputs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.444 CONF_NBR_OF_AUDIO_OUT

[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01d9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of audio outputs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.445 CONF_NBR_OF_ENC_STREAMS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb1	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the number of supported individual h26x streams	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.446 CONF_NBR_OF_EXT_ETH_COPPER_PORTS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a29	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of external copper ethernet ports of a device	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.447 CONF_NBR_OF_EXT_ETH_FIBER_PORTS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a2a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of external fiber ethernet ports of a device	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.448 CONF_NBR_OF_EXT_ETH_PORTS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a28	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of external ethernet ports of a device	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.449 CONF_NBR_OF_HUMIDITY_SENSORS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c60	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	The number of humidity sensors	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.450 CONF_NBR_OF_IMU

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd5	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of available IMU sensors (IMU: Inertial Measurement Unit; accelerometer, gyroscope, ..)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.451 CONF_NBR_OF_MANIPULATION_ALARMS

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af1	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of manipulation alarms	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.452 CONF_NBR_OF_MOTION_DETECTORS

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09af	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed motion detectors	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.453 CONF_NBR_OF_PIR

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c1a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed PIR sensors	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.454 CONF_NBR_OF_SD_CARD_SLOTS

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get number of sd card slots	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.455 CONF_NBR_OF_TEMP_SENS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09c4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	The number of temperature sensors	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.456 CONF_NBR_OF_TRAPS_HOSTS

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x029d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of hosts snmp traps can be sent to	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.457 CONF_NBR_OF_VIDEO_IN

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01d6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed video inputs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.458 CONF_NBR_OF_VIDEO_OUT

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01d7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of installed video outputs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.459 CONF_NBR_OF_VIRTUAL_ALARMS

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aed	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Number of virtual alarm inputs	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.460 CONF_NETWORK_SERVICES

[API: network services]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c62	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns all running network services	
Write	p_octet	service	Returns all running network services	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Reading the CONF_NETWORK_SERVICES command returns a list of TCP/UDP services which are available on the device. Each entry represents one service. The entry show if the service is enabled at the moment and what TCP/UDP port it is assigned to if a port is available. If the port cannot be adjusted the 0x0004 tag is simply missing. Client software can show this list to give the user the opportunity to check and configure all runiing network services.

Writing this command allows to configure the services returned by a preceding read. In write direction the command payload has the same structure. The client does not need to include all services returned by the read. It can only add the services to change to the write payload. The tag 0x0002 (label) must not be included in write payload. The service ID must provided and identitfies the service to change. The ID is not fixed across firmware versions and must be retrieved by a preceeding read.

Writing this command is only a request to the device to shutdown or start the services. That's why the device cannot answer with with the new state of the services. The device replies to a write operation with the actual state of the network services (the complete list). This does eventually not reflect the changes which will be done by the requests in this write operation.

Note: Most network services need a reboot of the device to be stopped or started. After writing this command the device should be rebootet!

Payload Structure

Service Entry [0]
...
Service Entry [N]

© Copyright 2001-2021 Bosch Security Systems

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

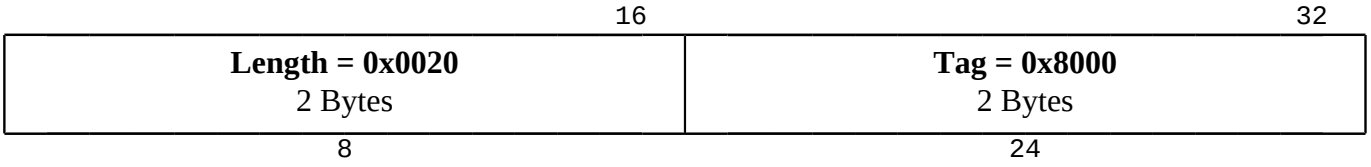
Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

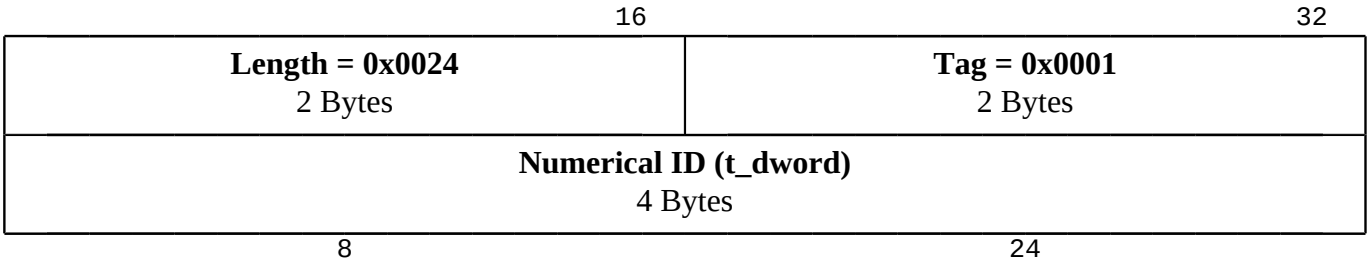
Tag 0x8000: Service Entry

Surrounding tag for each entry in the list. With help of the length field client software is the change to skip complete entries.



Tag 1: Numerical ID

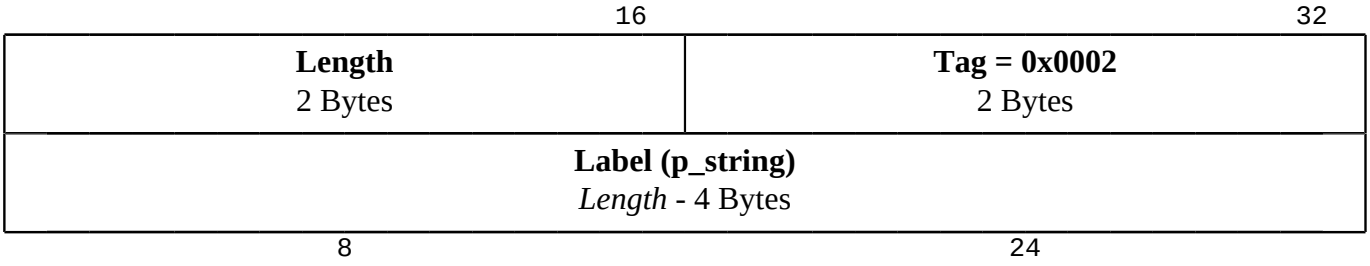
This tag gives each service a numerical ID. The ID should be retrieved via reading this command first. You can assume, that the ID for a special service keeps constant in newer firmware versions. So this ID can be treated as a unique identifier for a specific service. When using this command in write direction to configure a service this ID must be presented to identify the service.



© Copyright 2001-2021 Bosch Security Systems

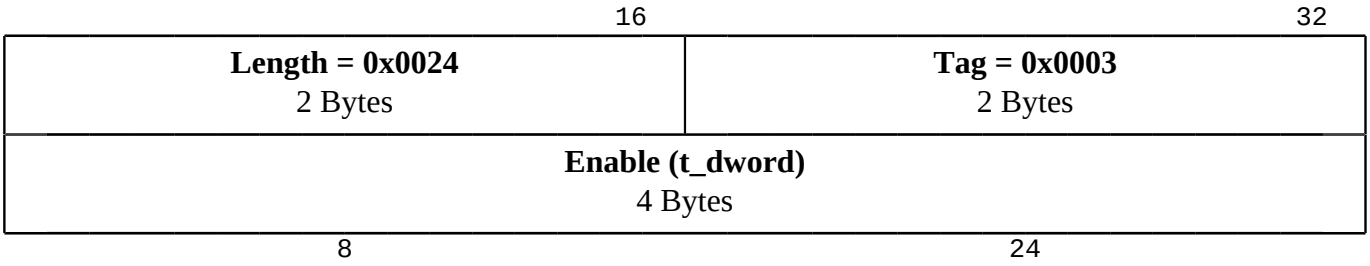
Tag 2: Label

A printable label for this service. If the client does not know how to translate this label it can show this label as default. The label is not used to identify the service and can be omitted when writing this command.



Tag 3: Enable

A value != 0 indicates that this service is enabled and the port is open at UDP/TCP level. A value == 0 indeicates that the service is disabled.

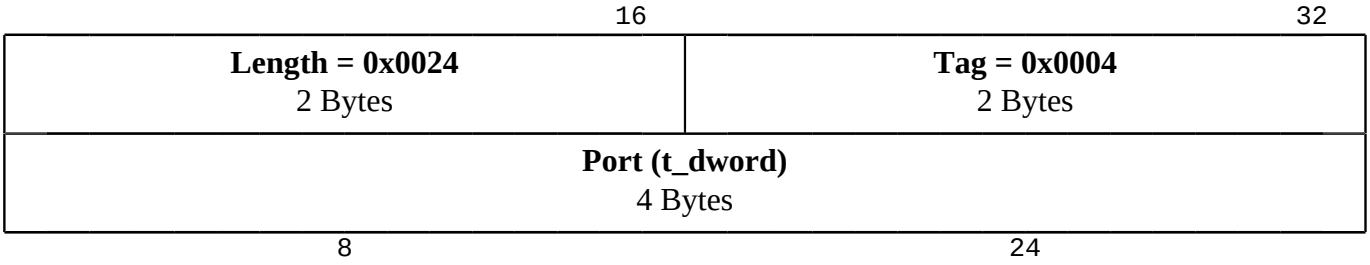


Values:

Enable	1
Disable	0

Tag 4: Port

This tag is only present if the UDP/TCP port is adjustable for this service. Else this tag is simply omitted. For historical reasons most services set their port to 0 to show that they are disabled. If you set the 'Enable' tag to 0 at write and set the port != 0 the command will automaticly set the port to 0. Setting 'Enabled' tag to != 0 and the port to 0 will use default port. Else the given port will be used.



© Copyright 2001-2021 Bosch Security Systems

2.461 CONF_NIGHT_MODE_STATE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aa2	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=night mode off; 1=night mode on (only for devices supporting night mode)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.462 CONF_NTP_SERVER_IP_STR

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x024f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Reads the time servers ip address (\xxx.xxx.xxx.xxx\	
Write	p_string	service	Or URL) used to sync time from	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.463 CONF_NTP_START_SERVER

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c63	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Reads if the NTP server of the camera should be started	
Write	f_flag	service	Set if the NTP server of the camera should be started (0 = do not start; 1 = start server)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.464 CONF_NTP_SYNC_MODE

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x031e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns mode which is used by time server, time server(0), SNTP server(1), off(2)	
Write	t_dword	service	Selects mode which is used to sync time on this machine, time server(0) SNTP server(1), off(2), tls_date(3)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.465 CONF_NUMBER_OF_VIPROC_CONFIGS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3c	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the number of configurations for the specified line.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.466 CONF_OBEY_ICMP_REDICRECTS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c13	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	service	Obey or ignore ICMP redirect messages,	
Write	t_octet	service	Obey or ignore ICMP redirect messages,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

ignore	0
obey	1

© Copyright 2001-2021 Bosch Security Systems

2.467 CONF_OEM_DEVICE_DOMAIN

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the oem device domain	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.468 CONF_OEM_DEVICE_NAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x097c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the oem device name	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.469 CONF_OEM_EXT_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x097d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	DEPRECATED: please use CONF_DEVICE_TYPE_IDS instead. Get the oem extension id	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.470 CONF_ONVIF_STREAM_MODE

[API: onvif]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bcb	coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	#ValueList: 1=stream is configured by onvif; 0=stream is not configured by onvif;	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.471 CONF_ONVIF_STREAM_URI_EX

[API: onvif]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Extension to the onvif stream uri that will be appended in the GetStreamUri command	
Write	p_string	service	Extension to the onvif stream uri that will be appended in the GetStreamUri command	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.472 CONF_OPTIONAL_TIME_SERVER_PORT

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c40	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the optional/alternative time server port. If not set (0), the according standard port will be used (depending on the configured time server mode)	
Write	t_word	service	Set an optional/alternative time server port. If not set (0), the according standard port will be used (depending on the configured time server mode).	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.473 CONF_OSD_ACCESS

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x098a	0: print to all coders, else, print only to selected absolute coder	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Access to the OSD via RCP command	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Up to version 4.50, this command only applies to decoders to display text onto the video output. The num field carries the video output number. A value of 0 indicates that the text should be displayed on all screens. Remind that the num value numbering scheme takes quad mode into account for the lines where quad is possible, even if quad mode is not active. (E.g. to have an OSD stamped into the fourth line of a VIP X1600 XFMD, you need to use num value of 10). To delete previously submitted strings, send an empty string (not space characters!).

From version 4.50 on this command applies to encoders as well. In that case, the encoder will embed the text information into the RTP data stream to be overlayed by the receiving decoder. The attribute and flag fields are not applicable to the decoder's video output. The num field applies to the video input number.

Payload Structure

16 32

ID 2 Bytes		**Len** 2 Bytes	
x 1 Byte	**y** 1 Byte	**bg_luma** 1 Byte	**bg_chroma** 1 Byte
attributes 4 Bytes			
font_luma 1 Byte	**font_chroma** 1 Byte	**flags** 2 Bytes	
Text *Len - 16* Bytes			

8 24

© Copyright 2001-2021 Bosch Security Systems

ID

Tag ID, for addressing different strings. Values 0x04-0x08 are possible.

Len

Length of the complete payload in bytes. Maximum length is 64 bytes.

x

X-Position (0...255) where the string should be displayed (0 is left)

y

Y-Position (0...255) where the string should be displayed (0 is up)

bg_luma

Background color: 8 bit luma (Y)

bg_chroma

Background color: 2 x 4 bit chroma (U/V); 'lower 4 bit' = U, 'upper 4 bit' = V

attributes

	Mask	Name	Description
Bit 20	0x00100000	ShowBorder	Draw a box around the text or transparent text background (0=box, 1=transparent)
Bit 11	0x00000800	Center	Centered display (0=off, 1=on)
Bit 10	0x00000400	BackgroundBox	Big text background box (0=off, 1=on)
Bit 8	0x00000100		Use custom font and background color given in bg_luma, bg_chroma, font_luma and font_chroma (0=off, 1=on)

font_luma

Font color: 8 bit luma (Y)

font_chroma

Font color: 2 x 4 bit chroma (U/V); 'lower 4 bit' = U, 'upper 4 bit' = V

flags

	Mask	Name	Description
Bit 0	0x0001	Unicode	Interpret text as unicode characters (UTF-16)

© Copyright 2001-2021 Bosch Security Systems

Text

Text to be displayed

© Copyright 2001-2021 Bosch Security Systems

2.474 CONF_OSD_POS

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae0	0: print to all coders, else, print only to selected absolute coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Access to the OSD via RCP command	
Write	p_octet	service	Access to the OSD via RCP command	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

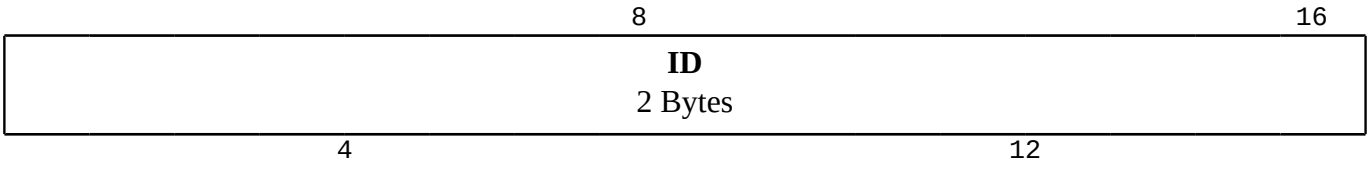
Description

Set or get the position of OSD objects (see CONF_OSD_ACCESS). For "write" the num field carries the video number; 0 indicates that the string positions on all video lines will be set.

Up to version 4.50, this command only applies to decoders to display text onto the video output. The num field carries the video output number. A value of 0 indicates that the text should be displayed on all screens. Remind that the num value numbering scheme takes quad mode into account for the lines where quad is possible, even if quad mode is not active. (E.g. to have an OSD stamped into the fourth line of a VIP X1600 XFMD, you need to use num value of 10). To delete previously submitted strings, send an empty string (not space characters!).

From version 4.50 on this command applies to encoders as well. In that case, the encoder will embed the text information into the RTP data stream to be overlayed by the receiving decoder. The attribute and flag fields are not applicable to the decoder's video output. The num field applies to the video input number.

Request Payload Structure

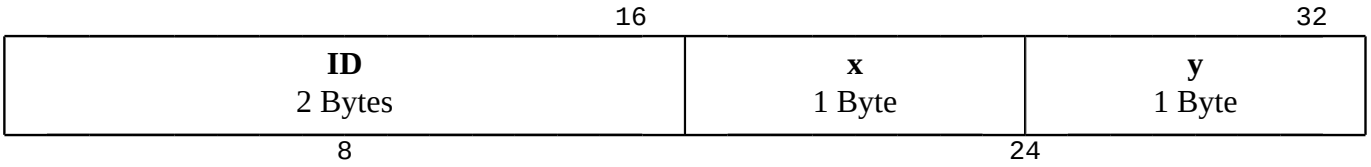


ID

Tag ID, for addressing different strings. Values 0x04-0x08 are possible.

Response/Write Payload Structure

© Copyright 2001-2021 Bosch Security Systems



ID

Tag ID, for addressing different strings. Values 0x04-0x08 are possible.

x

X-Position (0...255) of the string (0 is left)

y

Y-Position (0...255) of the string (0 is top)

© Copyright 2001-2021 Bosch Security Systems

2.475 CONF_PASSWORD_SETTINGS

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x028b	password level	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the unit password (scrambled). num parameter sets the password levels; 1=user, 2=service, 3=live	
Write	p_string	no_pwd	Set the unit password. num parameter sets the password levels; 1=user, 2=service, 3=live; Notice: only standard ASCII characters allowed!	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.476 CONF_PIC_INFO

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b8e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Enables/disables the generation of pic info packets (payload type 97)	
Write	t_octet	service	Enables/disables the generation of pic info packets (payload type 97)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.477 CONF_PIR_ALARM_STATE

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c1b	PIR sensor nbr	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=PIR alarm off; 1=PIR alarm on	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.478 CONF_PIR_SENSITIVITY

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bbb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Get sensitivity of PIR sensor.	
Write	t_octet	service	Set sensitivity of PIR sensor.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

disabled	0
lowest	1
highest	10

© Copyright 2001-2021 Bosch Security Systems

2.479 CONF_PIR_STATUS

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bbc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Obsolete - Raw ADC value of the PIR sensor.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.480 CONF_PMPP_ADDRESS

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ab9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the Pmpp Address (allowed values: 1-63;)	
Write	t_octet	service	Sets the Pmpp Address (allowed values: 1-63;)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.481 CONF_PMPP_PORT

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aba	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the Pmpp Port of the pmpp server,	
Write	t_word	service	Sets the Pmpp Port of the pmpp server,	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

off	0

© Copyright 2001-2021 Bosch Security Systems

2.482 CONF_POE_GRANTED_POWER

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae6	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the current poe power granted by switch / midspan in tenths of Watts	
Write	t_word	service	Write/limit the max power budget to be used by the device in tenths of Watts	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.483 CONF_PORT_FC_MODE

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0abb	num=0 is the flow control mode of the external state on single ethernet port units or the internal fc mode on multiple ethernet port units; num>=1 is the fc mode of the corresponding port on multiple ethenet port units (SFP Fiber port is always the last counted port).	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	On module based units, only the master module is capable of setting the external port fc modes.	
Write	t_octet	service	On module based units, only the master module is capable of setting the external port fc modes.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

off	0
on	1

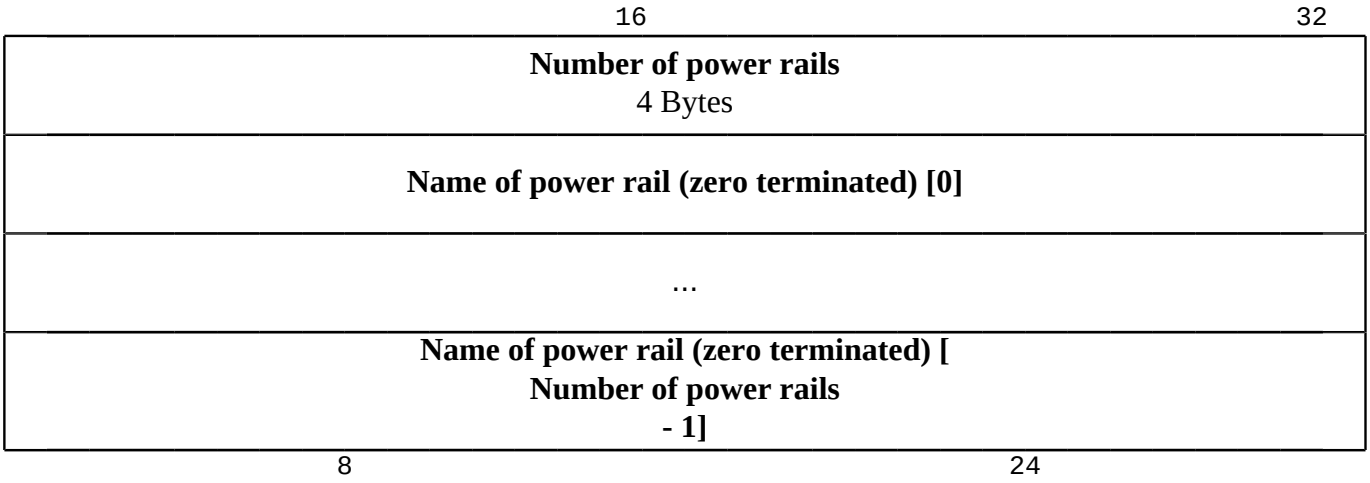
© Copyright 2001-2021 Bosch Security Systems

2.484 CONF_POWER_MONITOR_NAMES

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c54	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns names from all monitored power rails, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure



Number of power rails

Number of monitored power rails.

© Copyright 2001-2021 Bosch Security Systems

2.485 CONF_POWER_MONITOR_VALUES

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c53	Instance of Power Monitor	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns values from all monitored power rails, see detailed description	
Write	p_octet	service	Writing one Byte with 0x01 resets min/max/avg	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16	32
Number of rails 4 Bytes	
Rail value [0] (see description)	
...	
Rail value [Number of rails - 1] (see description)	

8	24

Number of rails

Number of monitored rails. Payload of complete command is 4 + (Number Of Rails)*40 Bytes

Rail value

16	32
Options 4 Bytes	
Actual Voltage 4 Bytes	
Min Voltage 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Max Voltage 4 Bytes
Actual Current 4 Bytes
Min Current 4 Bytes
Max Current 4 Bytes
Actual Power 4 Bytes
Min Power 4 Bytes
Max Power 4 Bytes

8 24

Options

	Mask	Name
Bit 2	0x00000004	Power Measurement Valid
Bit 1	0x00000002	Voltage Measurement Valid
Bit 0	0x00000001	Current Measurement Valid

Actual Voltage

Actual measured voltage, which is measured last, in uV

Min Voltage

Minimum measured voltage in uV

Max Voltage

Maximum measured voltage in uV

Actual Current

Actual measured current, which is measured last in uA

Min Current

Minimum measured current in uA

Max Current

Maximum measured current in uA

© Copyright 2001-2021 Bosch Security Systems

Actual Power

Actual measured power, which is measured last in uW

Min Power

Minimum measured power in uW

Max Power

Maximum measured power in uW

© Copyright 2001-2021 Bosch Security Systems

2.486 CONF_PREDEFINED_MOUNTING_LIST

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c76	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	user	Contains predefined mounting positions separated by semicolon, last three characters defines the associated tilt angle	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.487 CONF_PREPARE_FOR_RECORDING

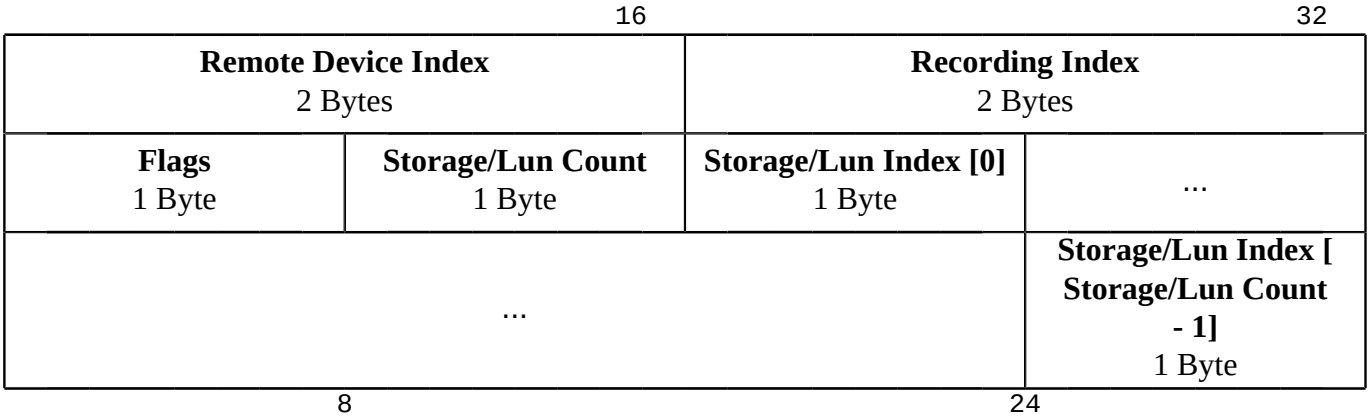
[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b51	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used to prepare a remote device for recording via a another device (e.g remote recording on a transcoder). Normally the remote device will be configured to record on one or more (upto 8) local managed storage, which is already configured by the command CONF_STORAGE_LIST. The Storage can be a local storage on the remote device or any storage on the local device. This is choosen by the flag "remote managed storage". If this flag is set, the storage/lun idx refers the lun index of a local storage on the remote device. Otherwise it is the index from the storage list of the local device. The preparation will be the configuration of the micro vrm on the recording device and in case of remote recording constellation, it will allow the remote recording on the storage managing device.

Payload Structure



Remote Device Index

Index of a remote device entry configured by CONF_ADD_REMOTE_DEVICE from 1 to n. Value 0 means the local device.

Recording Index

Primary Recording	1
Secondary Recording	2

© Copyright 2001-2021 Bosch Security Systems

Flags

	Mask	Name
Bit 0	0x01	Remote Managed Storage

Storage/Lun Count

Number of following index enties for storages/luns (max. 8)

Storage/Lun Index

Index of an as managed configured storage or the lun index of local storage of the remote device.

© Copyright 2001-2021 Bosch Security Systems

2.488 CONF_PRIV_MASK_DOME_PTZ_POS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c48	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 0: Privacy mask dome PTZ

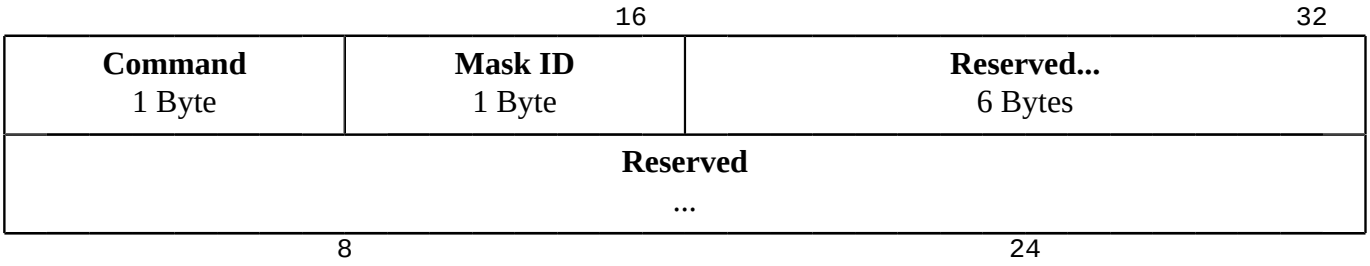
16 32

Length 2 Bytes	**Tag = 0x0000** 2 Bytes
Privacy mask dome PTZ (p_octet) *Length* - 4 Bytes	

8 24

© Copyright 2001-2021 Bosch Security Systems

Privacy mask dome PTZ



Command

Move To Defined Position	0	Move dome to position stored at maskId

Mask ID

Define mask ID which is used in conjunction with the command. Use CONF_PRIV_MSK_OPTIONS to query number of supported masks. (CONF_PRIV_MSK_OPTIONS / Max Number Of Priv Masks Per Line).

© Copyright 2001-2021 Bosch Security Systems

2.489 CONF_PRIV_MSK

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ab7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Legacy command please use CONF_PRIV_MSK_POLY	
Write	p_octet	service	Legacy command please use CONF_PRIV_MSK_POLY	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.490 CONF_PRIV_MSK_OPTIONS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bd7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16			32
Color Support 1 Byte	**Rectangle Support** 1 Byte	**Polygon Support** 1 Byte	**Max Number Of Polygon Vertices** 1 Byte
Max Number Of Priv Masks Per Line 1 Byte	**Only Legacy Cmd Support** 1 Byte	**Only Legacy Dome Cmd Support Via Bicom** 1 Byte	**Dome rcp priv mask support** 1 Byte
Dome zoom threshold support 1 Byte	**Number Of Priv Masks Lines** 1 Byte	**Dome Priv Mask Enlargement Support** 1 Byte	**IVA Behind Priv Mask Support** 1 Byte
Supported feature bits 1 Byte	**Reserved** 3 Bytes		
8			24

Color Support

Definition of color via (R,G,B) is supported

No	0
Yes	1

Rectangle Support

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

Polygon Support

No	0	
Yes	1	May not be self intersecting

Max Number Of Polygon Vertices

Maximum points for polygon definition

Max Number Of Priv Masks Per Line

Maximum Number Of Priv Masks Per Line

Only Legacy Cmd Support

No	0
Yes	1

Only Legacy Dome Cmd Support Via Bicom

No	0
Yes	1

Dome rcp priv mask support

No	0
Yes	1

Dome zoom threshold support

No	0
Yes	1

Dome Priv Mask Enlargement Support

No	0
Yes	1

IVA Behind Priv Mask Support

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

Supported feature bits

	Mask	Name
Bit 2	0x04	Pixelized
Bit 1	0x02	Privacy Mask Off
Bit 0	0x01	Auto Color Mode

© Copyright 2001-2021 Bosch Security Systems

2.491 CONF_PRIV_MSK_POLY

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bd8	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

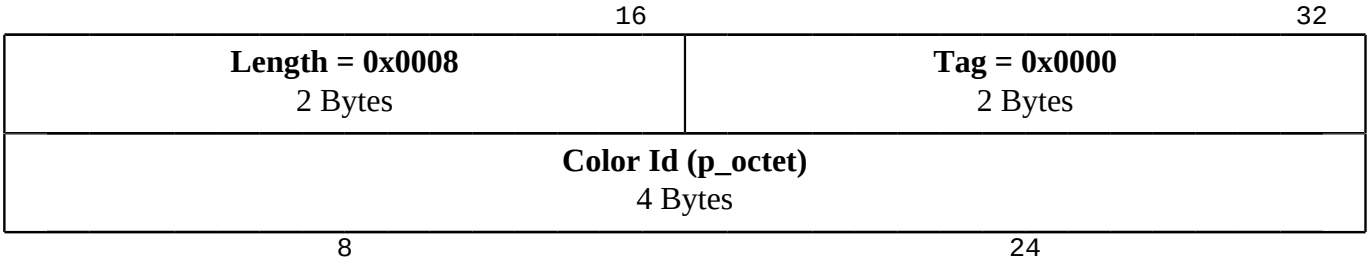
Tag

Tag specifying the encoding and meaning of the value

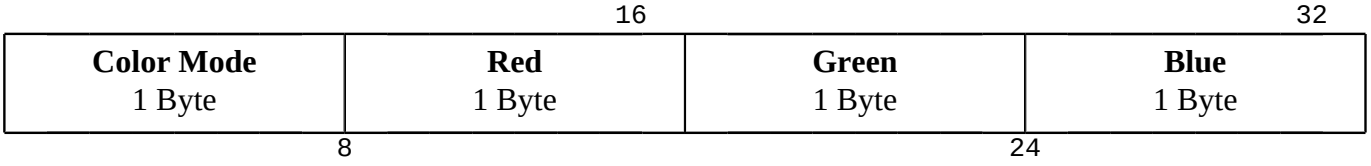
© Copyright 2001-2021 Bosch Security Systems

Tag 0: Color Id

Configure the priv mask color only supported if CONF_PRIV_MSK_OPTIONS allows the color configuration



Color Id



Color Mode

RGB	0	Red, Green, Blue values are used.
Auto Color Mode	1	Privacy mask color is auto selected from the surrounding. Red, Green, Blue values are ignored (use CONF_PRIV_MSK_OPTIONS to query support)
Off	2	Use CONF_PRIV_MSK_OPTIONS to query support.
Pixelized	3	Use CONF_PRIV_MSK_OPTIONS to query support.

Red

Value	0 - 255

Green

Value	0 - 255

Blue

Value	0 - 255

© Copyright 2001-2021 Bosch Security Systems

Tag 0: Color Mode

Use as an alternative to the Color Id tag above to change only the color mode without affecting the configured RGB values or any other tags. Color Mode supports the same values as the field with the same name in the Color Id tag

16	32
Length = 0x0005 2 Bytes	**Tag = 0x0000** 2 Bytes
Color Mode (t_octet) 1 Byte	
8	24

Tag 1: Polygon (fixed camera) description id

Configure up to n polygons query n via CONF_PRIV_MSK_OPTIONS.

CONF_PRIV_MSK_OPTIONS also defines if only rectangles are supported then only rectangles are allow to be send with this command.

Currently only a block configuration is supported. All polygons have to be sent as separate taged items within the same command.

Currently only non self intersecting polygons are supported refer to CONF_PRIV_MSK_OPTIONS.

16	32
Length 2 Bytes	**Tag = 0x0001** 2 Bytes
Polygon (fixed camera) description id (p_octet) *Length* - 4 Bytes	
8	24

Polygon (fixed camera) description id

16			32
Enable 1 Byte	**Id** 1 Byte	**Number Of Vertices** 1 Byte	**Reserved** 1 Byte
Reserved 4 Bytes			
Vertex [0] (see description)			

© Copyright 2001-2021 Bosch Security Systems

...
Vertex [Number Of Vertices - 1] (see description)

8 24

Enable

Disabled	0	Privacy mask is not shown.
Enabled	1	Privacy mask is shown.

Id

Mask number in range [0... max number privacy masks] (defined via CONF_PRIV_MSK_OPTIONS)

Number Of Vertices

Number of points for this polygon, must be in range [0... max number points] (defined via CONF_PRIV_MSK_OPTIONS)

Vertex

Definition of the coordinate system for privacy masking (stick to sei information)

Upper left edge : (0,0)

Upper right edge: (32768,0)

Lower left edge : (0,32768)

Lower right edge: (32768,32768)

16 32

X 2 Bytes	**Y** 2 Bytes

8 24

X

X Coordinate	0 - 32678

Y

Y Coordinate	0 - 32678

© Copyright 2001-2021 Bosch Security Systems

Tag 2: Polygon Config @ moving camera

use CONF_PRIV_MSK_OPTIONS to query if supported (CONF_PRIV_MSK_OPTIONS: Byte[7] rcp priv mask support @ dome): 2

Configure up to n polygons, query n via CONF_PRIV_MSK_OPTIONS.

CONF_PRIV_MSK_OPTIONS also defines if only rectangles are supported then only rectangles are allow to be send with this command.

Only one polygon is allowed to be send at once. Current dome position is automatically associated with the polygon

Currently only convex polygons are supported refer to CONF_PRIV_MSK_OPTIONS.

Use CONF_PRIV_MSK_OPTIONS to query if supported (CONF_PRIV_MSK_OPTIONS: Byte[7] rcp priv mask support @ dome)

16	32
Length 2 Bytes	**Tag = 0x0002** 2 Bytes
Polygon Config @ moving camera (p_octet) *Length* - 4 Bytes	

8 24

Polygon Config @ moving camera

16			32
Enable 1 Byte	**Id** 1 Byte	**Number Of Vertices** 1 Byte	**Zoom Threshold** 1 Byte
Reserved 4 Bytes			
Vertex [0] (see description)			
...			
Vertex [Number Of Vertices - 1] (see description)			

8 24

Enable

Disabled	0	Privacy mask is not shown.
Enabled	1	Privacy mask is shown.

© Copyright 2001-2021 Bosch Security Systems

Id

Mask number in range [0... max number privacy masks] (defined via CONF_PRIV_MSK_OPTIONS)

Number Of Vertices

Number of points for this polygon, must be in range [0... max number points] (defined via CONF_PRIV_MSK_OPTIONS)

Zoom Threshold

Availability defined via CONF_PRIV_MSK_OPTIONS

Disabled	0
Enabled	1

Vertex

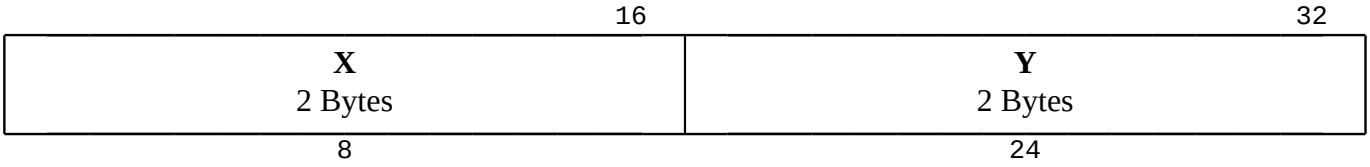
Definition of the coordinate system for privacy masking (stick to sei information)

Upper left edge : (0,0)

Upper right edge: (32768,0)

Lower left edge : (0,32768)

Lower right edge: (32768,32768)



X

X Coordinate	0 - 32678

Y

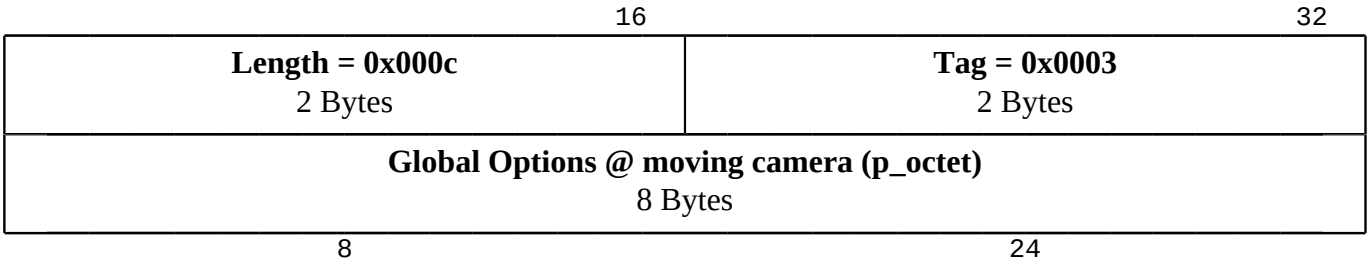
Y Coordinate	0 - 32678

Tag 3: Global Options @ moving camera

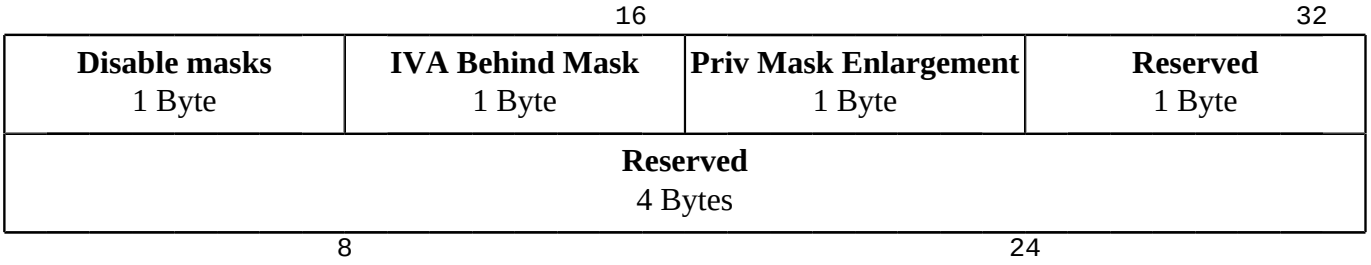
Global priv mask options at dome

© Copyright 2001-2021 Bosch Security Systems

Use CONF_PRIV_MSK_OPTIONS to query if supported (CONF_PRIV_MSK_OPTIONS: Byte[7] rcp priv mask support @ dome)



Global Options @ moving camera



Disable masks

Disable all privacy masks

IVA Behind Mask

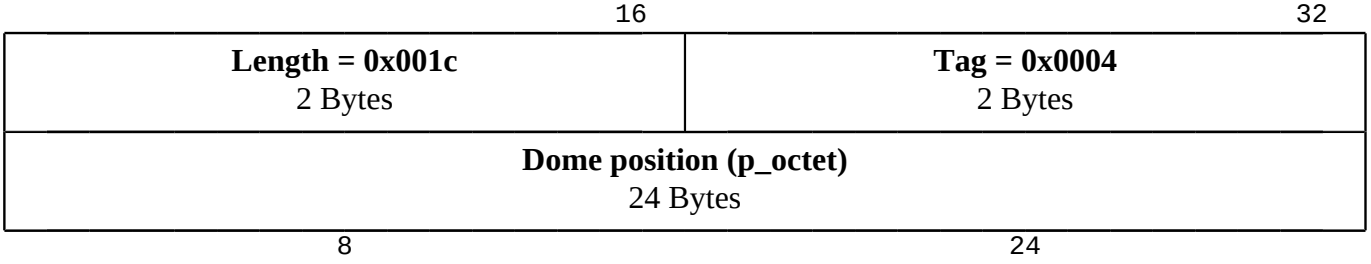
Enable IVA behind privacy masks

Priv Mask Enlargement

Automatically enlarge priv masks while moving

Tag 4: Dome position

Defines on moving cameras the view for a certain priv mask. Only for internal use! Automatically generated when a polygon is written on a dome.



© Copyright 2001-2021 Bosch Security Systems

Dome position

16		32
enable 1 Byte	**id** 1 Byte	**Reserved** 2 Bytes
panPosition 4 Bytes		
tiltPosition 4 Bytes		
zoomPosition 4 Bytes		
reserved 4 Bytes		
8		24

© Copyright 2001-2021 Bosch Security Systems

2.492 CONF_PRIVACY_MODE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c33	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the privacy mode status	
Write	t_dword	user	Set the privacy mode status	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Read Payload Structure

16	32
Status 1 Byte	**Source of Privacy Mode** 1 Byte	**opt. Timeout in min** 2 Bytes
8	24

Status

off	0
on	1

Source of Privacy Mode

RCP cmd	0
Tap	1

Write Payload Structure

16	32
Status 1 Byte	**reserved** 1 Byte	**opt. Timeout in minutes** 2 Bytes
8	24

Status

off	0
on	1

reserved

Set to '0'

© Copyright 2001-2021 Bosch Security Systems

opt. Timeout in minutes

Only applicable if 'Status' is 'On'. Then, after the timeout, privacy mode is automatically switched 'off' again

© Copyright 2001-2021 Bosch Security Systems

2.493 CONF_PRODUCT_NAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aea	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string p_octet	always	Read the product name of the device. This command was previously named \CTN\".	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.494 CONF_PROTECT_HTTP_COOKIE

[API: http settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cdc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Protect default session cookie with HttpOnly (legacy behaviour)	
Write	f_flag	service	Protect default session cookie with HttpOnly (legacy behaviour)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

Disabled	0
Enabled	1

© Copyright 2001-2021 Bosch Security Systems

2.495 CONF_PTZ_AUTO_ROTATE_MODE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c3f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Residential indoor ptz auto-rotate (auto-track) mode. 0=off, 1=std function, 2=production test mode	
Write	t_octet	service	Residential indoor ptz auto-rotate (auto-track) mode. 0=off, 1=std function, 2=production test mode	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.496 CONF_PTZ_CONTROLLER_AVAILABLE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a51	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Returns true if for the specified line a ptz controller is available which can be used by the VCA module.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.497 CONF_PUBLISH_MEDIA_KEYS

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09fc	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Switches whether the media key of a media source are readable and transferred during the hw-hw connection setup. Obsolete - keys are always forwarded on ssl control connections	
Write	f_flag	service	Switches whether the media key of a media source are readable and transferred during the hw-hw connection setup. Obsolete - keys are always forwarded on ssl control connections	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

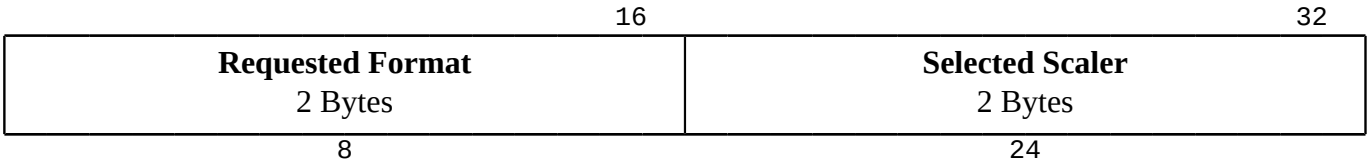
© Copyright 2001-2021 Bosch Security Systems

2.498 CONF_RAW_IMAGE

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c98	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure



Requested Format

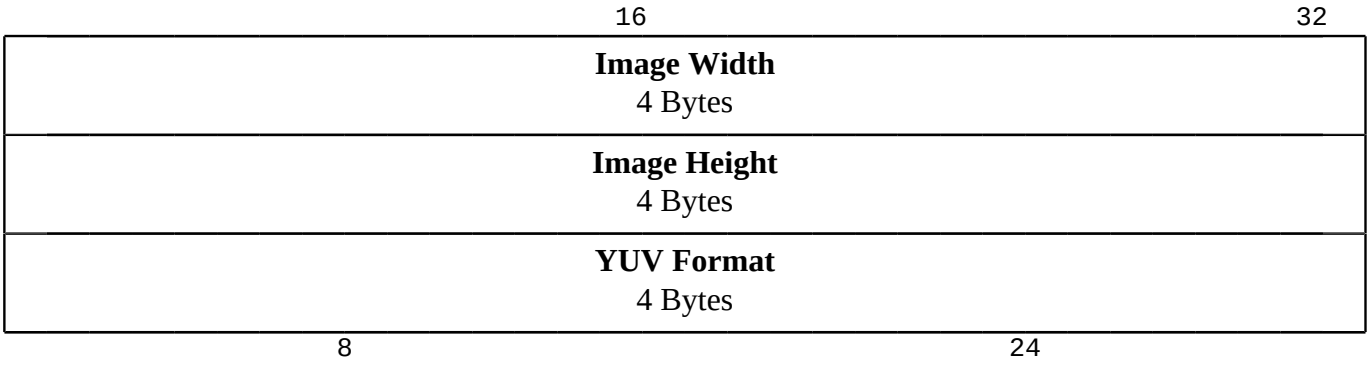
Raw YUV	0x0000	(default)
TIFF encoded raw	0x0001	(default)

Selected Scaler

VCS Scaler	0x0001	(default)

Response Payload Structure

Payload for 'Requested Format' = 'Raw YUV' (0x0000)



© Copyright 2001-2021 Bosch Security Systems

Image Width

Image Height

YUV Format

Planar420	1	420 planar, YYYY...UUUU...VVVV
Chroma420	-1	420 chroma interleaved, YYYY....UVUVUV
Planar422	2	422 planar, YYYY...UUUU...VVVV
Chroma422	-2	422 chroma interleaved, YYYY....UVUVUV

© Copyright 2001-2021 Bosch Security Systems

2.499 CONF_RCP_CLIENT_REGISTRATION

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff00	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always	Not supported; will generate message if client idle timeout will happen in 1 minute	
Write	p_octet	always		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The registration is a standard RCP packet with the command code 0xFF00 with data type P_OCTET (0x0c). The Numeric Descriptor is not used. The Client ID in the header section is ignored.

Request Payload Structure

16	32
Registration Type 1 Byte	**Reserved** 1 Byte	**Client ID** 2 Bytes
PE 1 Byte	**PL** 1 Byte	**Number Of Tags** 2 Bytes
Tag [0] 2 Bytes	…
Tag [Number Of Tags - 1] 2 Bytes	**Password...** N Bytes
Password ...	
8	24

Registration Type

Modify Registration	0x00
Normal Registration	0x01
Hook Back Registration	0x03

© Copyright 2001-2021 Bosch Security Systems

Client ID

Unused

PE

Password and User Delivery Encryption.

Plain text	0x00
MD5 hash	0x01

PL

Password and User String Length. Number of bytes to follow for the password string.

Number Of Tags

Number of message tags inside this packet.

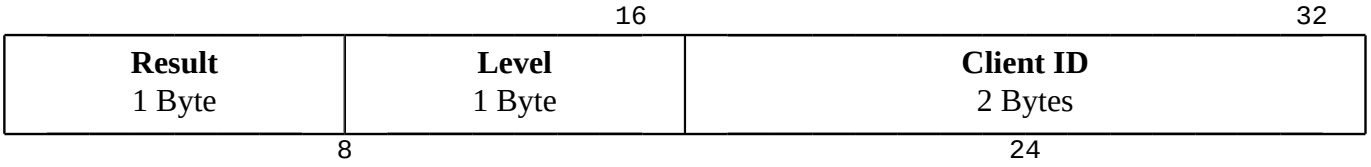
Tag

Tag codes for the messages which should be passed to the RCP client (see tag code table for capability to generate messages) - A Tag code of 0xFFFF will force the VideoJet to generate a message on all possible events.

Password

Password string itself. NOTE: No \0 termination!

Response Payload Structure



Result

Registration failed	0x00
Registration successful	0x01

Level

The level after this registration.

© Copyright 2001-2021 Bosch Security Systems

no	0
live	3
user	1
service	2

Client ID

Identification number under which the client is now registered at the VideoJet.

- The client opens up a TCP connection to the VideoJet.
- The client sends an RCP registration as mentioned above.
- The RCP server (the VideoJet) will respond with the result code and the assigned Client ID.
- The client is now allowed to send RCP Requests provided with the Client ID.

An already registered client can change the message list.

This mechanism is used during connection establishment only. A called host is requested to do a back register to the calling host. To achieve proper client allocation on the calling host, the Client ID of the server session on the called host must be provided in the 'Hook back' registration (Password section).

```
Password format: XXXX@PasswordString

 XXXX:         The Client ID in hexadecimal notation (4 digits)
 PasswordString: As defined below
```

There are currently three user levels on the videojet:

- live - For video access only; no control
- user - For video access and control over the video stream (e.g. caminput)
- service - for video and all administrative access

The identification string must have the following format: +Username:Password+

- A 16 character random string for MD5 hash calculation must be requested from the VideoJet (RCP command CONF_RCP_REG_MD5_RANDOM).
- A string +random_string+++username:password+ must be formed.
- The response MD5 hash over this string must be calculated.
- The identification string in the register packet must have the following format: +Username:random_string:response_string+

Note: No additional whitespaces are allowed.

© Copyright 2001-2021 Bosch Security Systems

2.500 CONF_RCP_CLIENT_REGISTRATION_V2

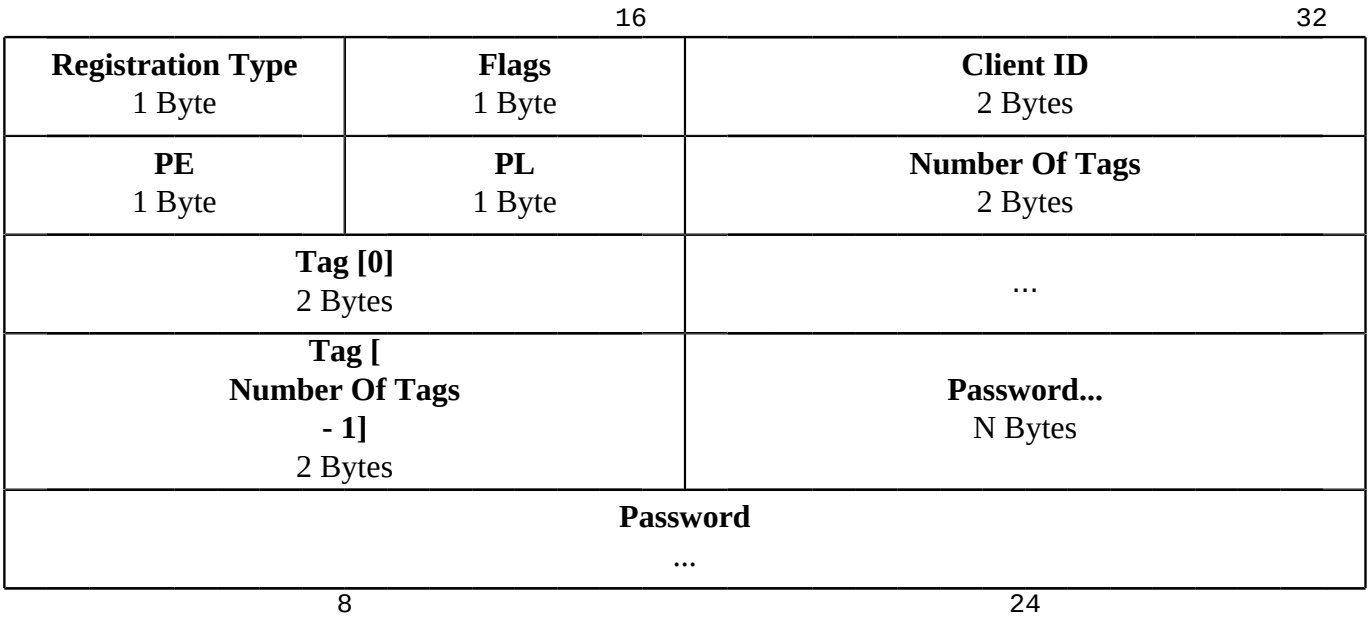
[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff04	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	always		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The registration is a standard RCP packet with the command code 0xFF00 with data type P_OCTET (0x0c). The Numeric Descriptor is not used. The Client ID in the header section is ignored.

Request Payload Structure



Registration Type

Modify Registration	0x00
Normal Registration	0x01
Hook Back Registration	0x03

© Copyright 2001-2021 Bosch Security Systems

Flags

	Mask	Name	Description
Bit 2	0x04	TAGGED_OPTION_HDR	Signal that the client will send extra tagged option list on each request. Internal only
Bit 1	0x02	MESSAGES_IN_V2_FORMAT	Register for messages with extended timestamp and sequence number header. If this flag is set, then all messages sent from the server to this client have an additional timestamp and sequence number header right after the RCP+ Header. This header is identical to the 'Timestamp Header Structure' returned by CONF_RCP_MSG_V2_HISTORY. Please see description of CONF_RCP_MSG_V2_HISTORY for details.

Client ID

Unused

PE

Password and User Delivery Encryption.

Plain text	0x00
MD5 hash	0x01

PL

Password and User String Length. Number of bytes to follow for the password string.

Number Of Tags

Number of message tags inside this packet.

Tag

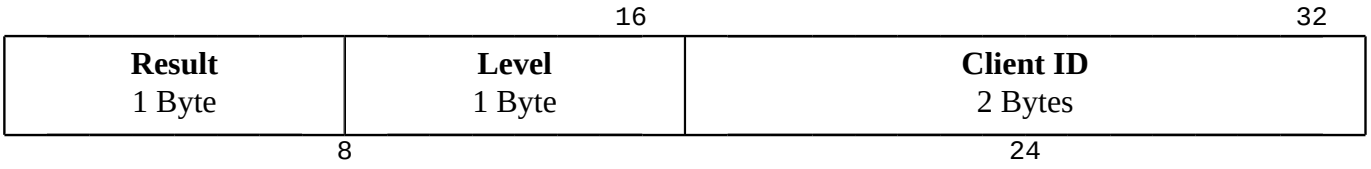
Tag codes for the messages which should be passed to the RCP client (see tag code table for capability to generate messages) - A Tag code of 0xFFFF will force the VideoJet to generate a message on all possible events.

Password

Password string itself. NOTE: No \0 termination!

Response Payload Structure

© Copyright 2001-2021 Bosch Security Systems



Result

Registration failed	0x00
Registration successful	0x01

Level

The level after this registration.

no	0
live	3
user	1
service	2

Client ID

Identification number under which the client is now registered at the VideoJet.

- The client opens up a TCP connection to the VideoJet.
- The client sends an RCP registration as mentioned above.
- The RCP server (the VideoJet) will respond with the result code and the assigned Client ID.
- The client is now allowed to send RCP Requests provided with the Client ID.

An already registered client can change the message list.

This mechanism is used during connection establishment only. A called host is requested to do a back register to the calling host. To achieve proper client allocation on the calling host, the Client ID of the server session on the called host must be provided in the 'Hook back' registration (Password section).

```
Password format: XXXX@PasswordString

 XXXX:         The Client ID in hexadecimal notation (4 digits)
 PasswordString: As defined below
```

There are currently three user levels on the videojet:

- live - For video access only; no control
- user - For video access and control over the video stream (e.g. caminput)
- service - for video and all administrative access

The identification string must have the following format: +Username:Password+

- A 16 character random string for MD5 hash calculation must be requested from the VideoJet (RCP command CONF_RCP_REG_MD5_RANDOM).
- A string +random_string+++username:password+ must be formed.
- The response MD5 hash over this string must be calculated.

© Copyright 2001-2021 Bosch Security Systems

- The identification string in the register packet must have the following format: +Username:random_string:response_string+

Note: No additional whitespaces are allowed.

© Copyright 2001-2021 Bosch Security Systems

2.501 CONF_RCP_CLIENT_TIMEOUT_WARNING

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff03	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The RCP server has to deal with lots of RCP connections at the same time. If an RCP client is going down without notification to the server, an orphan table entry will remain. To avoid this, a timeout mechanism is provided. Each RCP client must show any activity within 10 minutes. After this time the registration of a specific RCP client is invalid. For easy handling, the RCP client can register itself for an RCP_CLIENT_TIMEOUT_WARNIG message. The client will be notified 1 minute before the timeout occurs.

The timeout renewal can be achieved by any RCP read or write command. A good advise is to read the RCP_CLIENT_REGISTRATION.

© Copyright 2001-2021 Bosch Security Systems

2.502 CONF_RCP_CLIENT_UNREGISTER

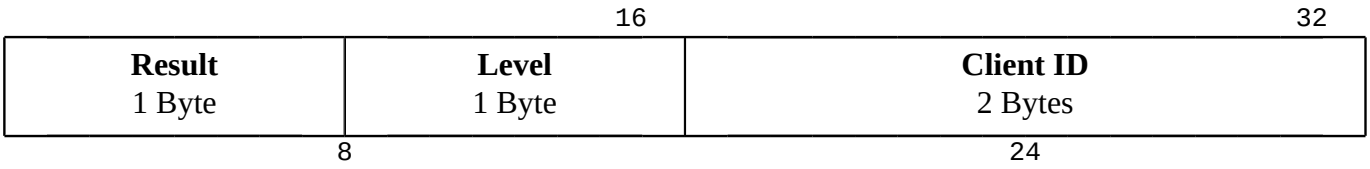
[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff01	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	always		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure

The unregister is a standard RCP packet with the command code 0xFF01 with no payload section. The needed information (Client ID) will be taken from the header section.

Response Payload Structure



Result

Unregister failed	0x00
Unregister successful	0x01

Level

The level before this unregister.

Client ID

Identification number under which the client was registered at the VideoJet.

Note: The command is not readable.

© Copyright 2001-2021 Bosch Security Systems

2.503 CONF_RCP_CODER_LIST

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff11	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used to retrieve the list of encoders and decoders from a VideoJet. There are two reply formats supported, the compact and the extended list view. The compact view only reports the absolute coder numbers available on a given line input or output. The extended view offers more information about coding capabilities and other parameters of a coder.

Payload Structure

	12	24
MediaType 1 Byte	**Direction** 1 Byte	**Flags** 1 Byte
6	18	

MediaType

Video	0x01
Audio	0x02
Data	0x03

Direction

Input	0x00
Output	0x01

Flags

	Mask	Name	Description
Bit 0	0x01	Extended Format	Request extended format instead of compact.

Response Payload Structure

© Copyright 2001-2021 Bosch Security Systems

Payload for 'Flags' = 'Compact Format Response Structure'

Coder Number [0] 1 Byte	...	**Coder Number [N]** 1 Byte

Coder Number

Absolute coder numbers.

Extended Format Video Response Structure

Payload for any other case

Coder Definition [0] (see description)
...
Coder Definition [N] (see description)

Coder Definition

16	32
Coder Number 2 Bytes	**Coding Capabilities** 2 Bytes
Current Coding 2 Bytes	**Resolution Capabilities** 2 Bytes
Resolution Current 2 Bytes	**Reserved...** 6 Bytes
Reserved ...	
8	24

Coder Number

Absolute coder numbers.

Coding Capabilities

All coding capabilities is one or multiple of:

	Mask	Name
Bit 15	0x8000	Mpeg2 PrgStr
Bit 14	0x4000	Recorded Media
Bit 9	0x0200	H.265

© Copyright 2001-2021 Bosch Security Systems

Bit 7	0x0080	Jpeg
Bit 6	0x0040	H.264
Bit 5	0x0020	H.263-1998
Bit 4	0x0010	Meta Data
Bit 3	0x0008	Mpeg2
Bit 2	0x0004	Mpeg4
Bit 1	0x0002	H.263

Current Coding

See 'Coding Capabilities' for details.

The current available coding capabilities.

Resolution Capabilities

All resolution capabilities is one or multiple of:

	Mask	**Name**
Bit 12	0x1000	HD5M
Bit 11	0x0800	WD432
Bit 10	0x0400	WD288
Bit 9	0x0200	WD144
Bit 8	0x0100	HD1080
Bit 7	0x0080	HD720
Bit 6	0x0040	VGA
Bit 5	0x0020	QVGA
Bit 4	0x0010	Custom
Bit 3	0x0008	4CIF
Bit 2	0x0004	2CIF
Bit 1	0x0002	CIF
Bit 0	0x0001	QCIF

Resolution Current

See 'Resolution Capabilities' for details.

The current available resolution capabilities.

Extended Format Audio Response Structure

© Copyright 2001-2021 Bosch Security Systems

Payload for any other case

Coder Definition [0] (see description)
...
Coder Definition [N] (see description)

Coder Definition

16	32
Coder Number 2 Bytes	**Coding Capabilities** 2 Bytes
Current Coding 2 Bytes	**CodParameter Capabilities** 2 Bytes
CodParameter Current 2 Bytes	**Reserved...** 6 Bytes
Reserved ...	
8	24

Coder Number

Absolute coder numbers.

Coding Capabilities

All coding capabilities is one or multiple of:

	Mask	Name
Bit 15	0x8000	Mpeg2 PrgStr
Bit 14	0x4000	Recorded Media
Bit 4	0x0010	L16
Bit 3	0x0008	L16 (16kHz timestamps)
Bit 2	0x0004	G.711 (8kHz timestamps)
Bit 1	0x0002	AAC
Bit 0	0x0001	G.711

Current Coding

See 'Coding Capabilities' for details.

© Copyright 2001-2021 Bosch Security Systems

The current available coding capabilities.

CodParameter Capabilities

Currently none available

CodParameter Current

Currently none available

Extended Format Data Response Structure

Payload for any other case

Coder Definition [0] (see description)
...
Coder Definition [N] (see description)

Coder Definition

16	32
Coder Number 2 Bytes	**Coding Capabilities** 2 Bytes
Current Coding 2 Bytes	**CodParameter Capabilities** 2 Bytes
CodParameter Current 2 Bytes	**Reserved...** 6 Bytes
Reserved ...	
8	24

Coder Number

Absolute coder numbers.

Coding Capabilities

All coding capabilities is one or multiple of:

	Mask	Name
Bit 0	0x0001	RCP

© Copyright 2001-2021 Bosch Security Systems

Current Coding

See 'Coding Capabilities' for details.

The current available coding capabilities.

CodParameter Capabilities

	Mask	Name
Bit 2	0x0004	RS422
Bit 1	0x0002	RS485
Bit 0	0x0001	RS232

CodParameter Current

See 'CodParameter Capabilities' for details.

The current available coding capabilities.

© Copyright 2001-2021 Bosch Security Systems

2.504 CONF_RCP_CONNECT_SALVO

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff13	yes(0 - nothing, 1 to 4 refering to entries of CONF_ADD_REMOTE_DI	no	no

	Datatype	Access Level	Description
Read	-	-	Unavailable
Write	p_octet	live	

	CPP6/CPP7/CPP7.3	CPP13
Available	yes	yes

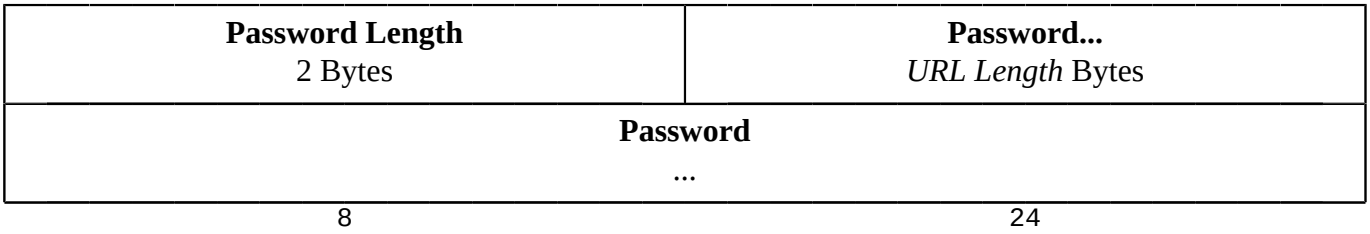
Description

A replay connection can be established in two steps: 1) build up a "dry" (replay) connection by setting the flag in the CONNECT_PRIMITIVE request. A Rcp connection is established and a session id to control the connection is returned. Data is not sent out as long as no source is connected. 2) connect a source to the connection (only implemented for replay) using this RCP_CONNECT_SALVO_COMMAND with the provided session id. The connection will retrieve the data from the specified device If a source is already connected a second call of this command will replace the first connection. The num parameter can be used to refere to an entry from CONF_ADD_REMOTE_DEVICE (num 1 to 4). In that case only a shorter payload (line and relative coder) is required. All other parameters will be taken from the CONF_ADD_REMOTE_DEVICE command. For port and ssl port the first two ports of a CONF_ADD_REMOTE_DEVICE entry will be used in that order. If the num parameter is set to zero, all other parameters are required and used if the connection is a remote device. If num is zero and the short payload (line and rel coder) is also used, it is asumed, that the local device is addressed.

Payload Structure

16			32
Flags 1 Byte	**Line** 1 Byte	**Decoder Instance** 1 Byte	**Relative Coder** 1 Byte
Port 2 Bytes		**SSL Port** 2 Bytes	
VRM Track Id 2 Bytes		**Reserved** 2 Bytes	
URL Length 2 Bytes		**URL...** *URL Length* Bytes	
URL ...			

© Copyright 2001-2021 Bosch Security Systems



Flags

	Mask	Name	Description
Bit 6	0x40	Delete cam	delete cam connected to dec inst (multi view only)
Bit 3	0x08	Replay live only	
Bit 1	0x02	Use TCP to get live video	
Bit 0	0x01	Live video	

Line

Line from 1 to max lines (line 0 is mapped to line 1)

Decoder Instance

Decoder instance from 1 to max ... (decoder instance 0 means any) for multiview capable de/transcoders

Relative Coder

Relative coder number from 1 to ... (coder number 0 is mapped to coder number 1)

VRM Track Id

Only relevant for VRM

URL Length

Length of the following url (actual limit is 64)

URL

URL for the connection salvo. ASCII string including zero termination. If the special url starting with 'trackid://' is used, the following decimal value is the VRM Track ID (e.g. 'tackid://15096')

Password Length

Length of the following password (actual limit is 64)

Password

Password for the salvo connection. ASCII string including zero termination.

© Copyright 2001-2021 Bosch Security Systems

2.505 CONF_RCP_CONNECTIONS_ALIVE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xffc2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live	Retriggers a running connection identified by its session_id; returns the current number of active connections	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

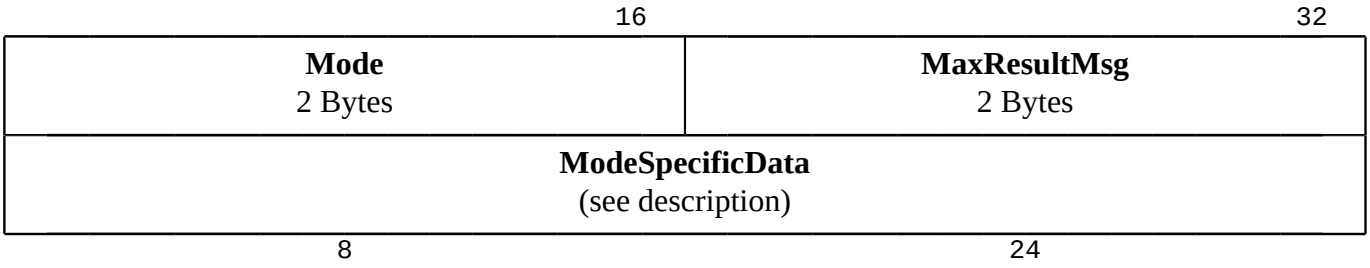
© Copyright 2001-2021 Bosch Security Systems

2.506 CONF_RCP_MSG_V2_HISTORY

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c57	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Read the list of RCPv2 Event-Messages pending in device history buffer, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure



Mode

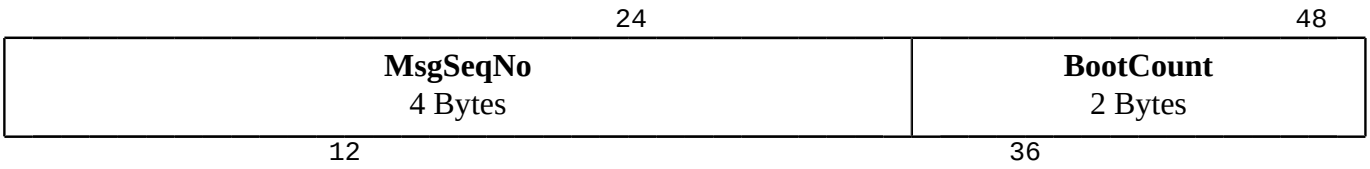
By Sequence Number	0x0000	Select the mode to read messages indicated by a initial sequence number.
By Timestamp	0x0001	Select the mode to read messages indicated by by relative timestamp

MaxResultMsg

Maximum number of result messages to be returned by one execution of this command.

ModeSpecificData

'ModeSpecificData' payload for 'Mode' = 'Sequence Number' (0x0000)



© Copyright 2001-2021 Bosch Security Systems

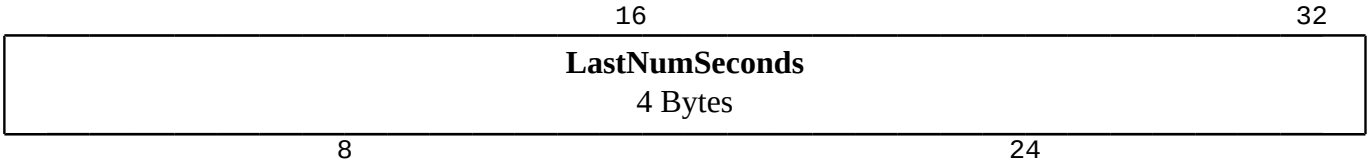
MsgSeqNo

The first returned message will be the next in device buffer after those indicated by this number. e.g. if MsgSeqNo is 7, then first result is 8 or even a higher number if there is no message 8 according to filters in this data path. If value 0 is supplied then the result starts with the oldest message which is still available on the device. To perform a chunked read the last MsgSeqNo returned by prior request to this command can be the argument of the next request to retreive the next chunk.

BootCount

Boot counter. This in combination with MsgSeqNo makes a unique ID for a message. If supplied BootCount is lower than actual BootCount then all messages in Buffer are returned up to MaxResultMsg

'ModeSpecificData' payload for 'Mode' = 'Timestamp' (0x0001)



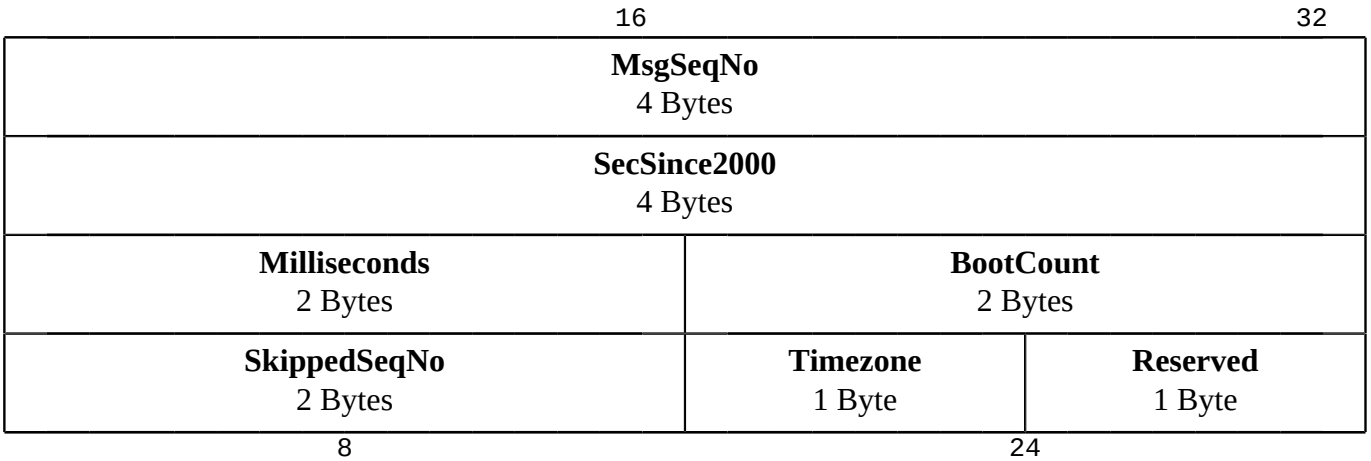
LastNumSeconds

The first returned message will be those indicated by this number of seconds back in the past. E.g. value of 30 will start returning the oldest messages which was triggered 30 seconds ago, and then continue with all subsequent up to MaxResultMsg.

Response Payload Structure

The result payload consists of a repeated sequence for each returned message. A RCP-Plus header (see chapter 'Transport Protocol'), a Timestamp Header, and then the usual RCP-Message payload.

Timestamp Header Structure



© Copyright 2001-2021 Bosch Security Systems

MsgSeqNo

Message counter starts with 0 on each bootup

SecSince2000

Time in seconds from RTC in UTC

Milliseconds

Millisecond extension to SecSince2000

BootCount

Number of reboots. Use in combination with MsgSeqNo to get a unique Message-ID

SkippedSeqNo

Number of skipped sequence numbers in this stream due to registering/filtering rules

Timezone

Timezone in 15 minute steps. Negative values have the leading bit 7 set. E.g. 0x03 = +45 minutes; 0x84 = -60 minutes

© Copyright 2001-2021 Bosch Security Systems

2.507 CONF_RCP_REG_MD5_RANDOM

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xff05	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string p_octet	always	Returns a 16 char. random string to be used in MD5 hash encrypted registration. see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Note: To avoid 'denial of service' attacks, the RCP server will only grant up to 5 random strings at the same time. Maximum time from the request of a random string to its use: 5 seconds. Multiple requests from the same host are ignored.

© Copyright 2001-2021 Bosch Security Systems

2.508 CONF_RCP_SERVER_PORT

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a17	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the local RCP server TCP port number	
Write	t_word	service	Set the local RCP server TCP port number, allowed: 0 or 1756 (Reboot necessary)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.509 CONF_RCP_TRANSFER_TRANSPARENT_DATA

[API: rcp.control]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xffdd	yes	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	user		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The transparent data from and to the serial interfaces is handled by RCP to achieve reliable transfer of information. To gather control over the remote serial interface a successful registration is necessary.

Once the VideoJet has received a TRANSFER_TRANSPARENT_DATA command, it checks whether the RCP client is in control or not. If the RCP client is allowed to send data to the serial interface, the reply will present an OK. If the RCP client is not allowed to send data, a FAIL will be returned. In this case another RCP client is controlling the serial out. The timeout and priority handling of the serial ports is beyond the scope of this document.

Note: This command is NOT readable in a sense to obtain data from a serial port. In can only be read in order to check if serial port access is currently granted. See further details below.

The reply to the Read command will be the same as the reply to the write command. The returned code will present the availability of the serial port.

NOTE: Despite a positive reply to a read command, the port may be locked by another RCP client in the time slice between the read and a following write command.

The data coming from the serial input is delivered using an RCP message. All RCP clients which want to receive this data must be registered for the message 0xFFDD. Data is posted if the corresponding RCP client is in control only.

If no client has a lease on the serial port, a message to all registered clients will be generated.

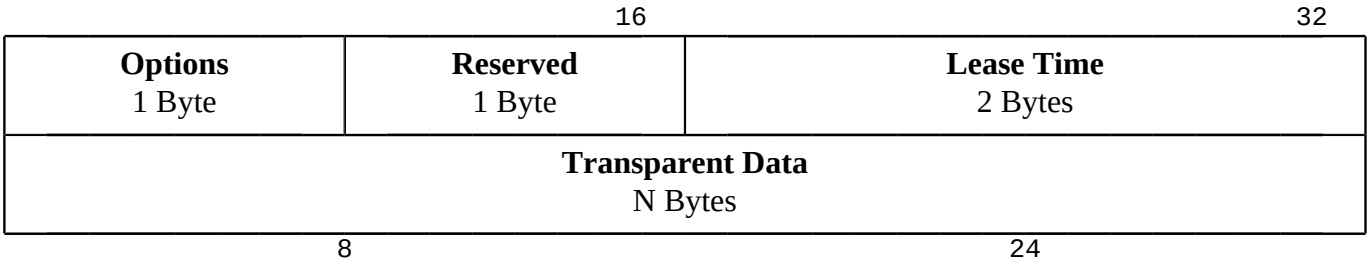
NOTE: The received message will carry NO header.

Num Descriptor Values

Any	Port to write to

Write Payload Structure

© Copyright 2001-2021 Bosch Security Systems



Options

Currently no options used

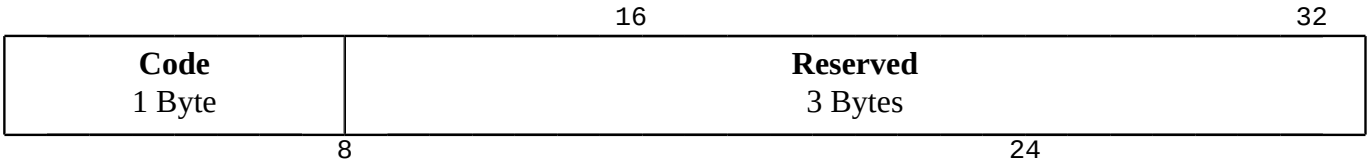
Lease Time

Time in seconds the lease is requested

	Any	Time in seconds the lease is requested
None	0x0000	Only this packet should be sent out; no further control is requested
Infitinite	0xFFFF	Indefinite lease time; request lease as long as the current registration is valid

Note: The lease time should be treated as a request; the VideoJet may switch leases before the requested time is over due to a higher prioritized RCP client

Response Payload Structure



Code

Access Denied	0x00	Access to the serial port denied
Access Granted	0x01	Access to the serial port granted

© Copyright 2001-2021 Bosch Security Systems

2.510 CONF_REC_MGNT

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a89	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read the type of the recording management (0 = LOCAL, 1 = VRM, 2 = VRM+ANR, 3 = ONLY LOCAL (obsolete), 4 = DUAL VRM, 5 = ONVIF)	
Write	t_octet	service	Set the type of the recording management (0 = LOCAL, 1 = VRM, 2 = VRM+ANR, 3 = ONLY LOCAL(obsolete), 4 = DUAL VRM, 5 = ONVIF) (not possible to change settings while recording)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.511 CONF_REC_MONITOR_STATUS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cfc	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the recorder monitor status, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to get the status of recording, whether it records all data as desired or if unintentional loss is occuring. It also gives a hint as utc time interval, when the data where lost. The status is also send as message on status changes from nominal to critical and vise versa.The status changes to critical, when loss occures and returns to status nominal after 2 minitus without further data loss

Request Payload Structure

16	32
Camera 2 Bytes	**Recording Index** 2 Bytes
8	24

Camera

The camera index 1 to n

Recording Index

Primary Recording	1
Secondary Recording	2

Response Payload Structure

16		32
Camera 2 Bytes		**Recording Index** 2 Bytes
Status 1 Byte	**Version** 1 Byte	**reserved** 2 Bytes
recorded high 4 Bytes		

© Copyright 2001-2021 Bosch Security Systems

recorded low 4 Bytes
lost high 4 Bytes
lost low 4 Bytes
issue start time utc 4 Bytes
issue end time utc 4 Bytes
latest issue time utc 4 Bytes
info time stamp 4 Bytes
State ID 4 Bytes

8 24

Camera

The camera index from 1 to n

Recording Index

Primary Recording	1
Secondary Recording	2

Status

NONE	0	no info
NOMINAL	1	no actual recording issue
CRITICAL	2	record data loss within the last 2 minutes

Version

Payload version (actual 1).

reserved

recorded high

Upper 32 bit of 64 bit counter for recorded packets.

© Copyright 2001-2021 Bosch Security Systems

recorded low

Lower 32 bit of 64 bit counter for recorded packets.

lost high

Upper 32 bit of 64 bit counter for recorded packets.

lost low

Lower 32 bit of 64 bit counter for recorded packets.

issue start time utc

Seconds since 2000 utc timestamp for latest issue time interval start

issue end time utc

Seconds since 2000 utc timestamp for latest issue time interval end

latest issue time utc

Seconds since 2000 utc timestamp for latest reported issue

info time stamp

Seconds since 2000 utc timestamp for this info

State ID

Changes every time the status changes

© Copyright 2001-2021 Bosch Security Systems

2.512 CONF_REC_SPAN_MGR

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a36	1 - primary recording, 2 - secondary recording	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Use this command to edit the recording span manager list. This list is consulted every time when a device is recording in VRM mode and its span list runs out resources. The entry with the highest priority is picked out of this list, the corresponding span manager is contacted and queried for spans. If the span manager is not accessable or will not return a span for some reason (maybe it has no more free spans to export), the entry with the next lower priority is picked out of the list. If at least one span manager for primary recording is configured, all entries of managed vrm (configured by CONF_MANAGING_VRM) will be deleted.

Payload Structure

Span Manager Config [0] (see description)
...
Span Manager Config [N] (see description)

Span Manager Config

Repeated 8 times at maximum

16	32
Span Mananger IP Address 4 Bytes	
Target ID 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Target Index 1 Byte	Lun 1 Byte	Reserved 2 Bytes
Priority 2 Bytes		Flags 2 Bytes

8 24

Span Mananger IP Address

The IP address of the device the span manager is running on.

Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

Target Index

The iSCSI target index.

Lun

The iSCSI lun of the storage.

Priority

Priority as recording storage, higher values mean lower priority, 0 - highest priority

Flags

	Mask	Name	Description
Bit 0	0x0001	OVERWRITE_OLDEST_RECORDINGS	(obsolete)

Note: If no free spans are available on all span manager of the devices recording span manager list, the recording with the retention time that expires next is overwritten.

© Copyright 2001-2021 Bosch Security Systems

2.513 CONF_REC_STORAGE_REQ_CFG

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b30	recording index: 1 - primary, 2 - secondary	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read error tollerance parameter for storage requests from recording (4 bytes, 2 words) (1st word retry in network order 0xffff for retry always, 2nd word timeout seconds in network order 0 for default timeout 0xffff for never timeout	
Write	p_octet	service	Set error tollerance parameter for storage requests from recording (4 bytes, 2 words) (1st word retry in network order 0xffff for retry always, 2nd word timeout seconds in network order 0 for default timeout 0xffff for never timeout	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.514 CONF_RECORDING_BUFFER_LEVEL

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b70	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the levels in percent of the recording rate control based on the recording buffer fill-level: 4 bytes: level off (if the fill level is below that value the rate control will be turned off), 4 bytes: level on (if the fill level is above that value the rate control will be turned on), (set both values to 0 to disable the rate control). This features is enabled per default on TI devices with 10% (off) and 30% (on)	
Write	p_octet	service	Returns the levels in percent of the recording rate control based on the recording buffer fill-level: 4 bytes: level off (if the fill level is below that value the rate control will be turned off), 4 bytes: level on (if the fill level is above that value the rate control will be turned on), (set both values to 0 to disable the rate control). This features is enabled per default on TI devices with 10% (off) and 30% (on)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.515 CONF_RECORDING_RETENTION_TIME

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a30	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the primary recording retention time for a camera (in seconds).	
Write	t_dword	service	Set the primary recording retention time for a camera (in seconds) (value >= 1009152000) means maximum, no influence on local recording, only for vrm managed recording. (effect takes place on next mounted span)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.516 CONF_RECORDING_RETENTION_TIME_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a48	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the secondary recording retention time for a camera (in seconds).	
Write	t_dword	service	Set the secondary recording retention time for a camera (in seconds) (value >= 1009152000) means maximum, no influence on local recording, only for vrm managed recording. (effect takes place on next mounted span)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.517 CONF_RECORDING_STATUS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a9b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the current recording status,	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure

16		32
Camera 1 Byte	**Recording Index** 1 Byte	**Reserved...** 50 Bytes
Reserved [...]		
Reserved ...		
Reserved ...		
8		24

Camera

The camera index

Recording Index

Primary Recording	1
Secondary Recording	2

Response Payload Structure

16			32
Camera 1 Byte	**Recording Index** 1 Byte	**Status** 1 Byte	**Error** 1 Byte
Current Recording Span... (see description)			

© Copyright 2001-2021 Bosch Security Systems

Current Recording Span ...	
Flags 1 Byte	**Reserved** 3 Bytes
Datarate 4 Bytes	
Packet alloc no wait 4 Bytes	
Packet alloc wait 4 Bytes	
Packet alloc failed 4 Bytes	
Recent time period 4 Bytes	
Packets recorded high 4 Bytes	
Packets recorded low 4 Bytes	
Packets lost high 4 Bytes	
Packets lost low 4 Bytes	

8 24

Camera

The camera index

Recording Index

Primary Recording	1
Secondary Recording	2

Status

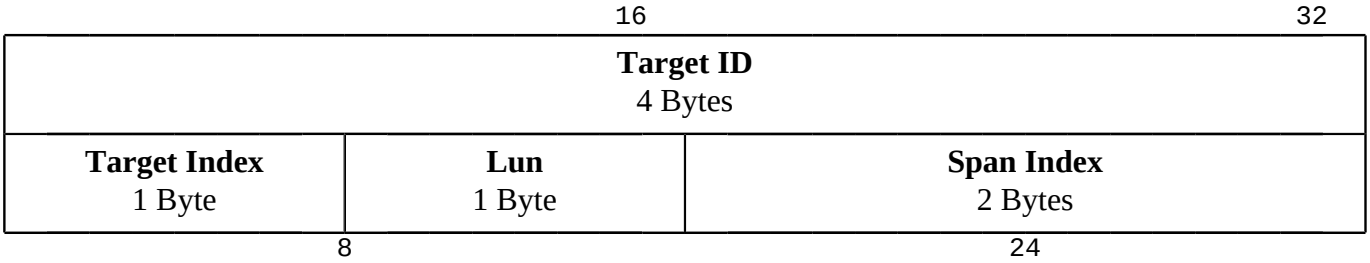
OFFLINE	1	configured for not recording
IDLE	2	(obsolete)
PEND	3	trying to connect storage or waiting for block list from vrm
RUNNING	4	storage connected and recording running or prepared for recording
ERROR	5	error on storage ocurred
REC_SRC_ERROR	6	error of recorder source(encoder)

© Copyright 2001-2021 Bosch Security Systems

Error

See error codes of CONF_SPAN_USE_STATUS.

Current Recording Span



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

Target Index

In index of the iscsi target.

Lun

The lun indentifier.

Span Index

The index of the span.

Flags

	Mask	Name	Description
Bit 0	0x01	ENCRYPTION	Encrypted recording

Datarate

The datarate that is written to the storage.

Packet alloc no wait

Recently alloc no wait vdp packets (1.5 kbyte) since recent time period

Packet alloc wait

Recently alloc wait vdp packets (1.5 kbyte) since recent time period (indicates problems of the recording get its data to the drive)

© Copyright 2001-2021 Bosch Security Systems

Packet alloc failed

Recently alloc failed vdp packets (1.5 kbyte) since recent time period (indicates problems of the recording get its data to the drive, corrupt data stream)

Recent time period

Time period in which the packet allock counter were counted

Packets recorded high

High part (32 bits network order) of a 64 bit counter for all recorded packets of this recording since device start up (packets counted by 'packet alloc no wait' and 'packet alloc wait' not included)

Packets recorded low

Low part (32 bits network order) of a 64 bit counter for all recorded packets of this recording since device start up (packets counted by 'packet alloc no wait' and 'packet alloc wait' not included)

Packets lost high

High part (32 bits network order) of a 64 bit counter for all packets, which couldn't be recorded by this recording since device start up (packets counted by 'packet alloc failed' not included)

Packets lost low

Low part (32 bits network order) of a 64 bit counter for all packets, which couldn't be recorded by this recording since device start up (packets counted by 'packet alloc failed' not included)

© Copyright 2001-2021 Bosch Security Systems

2.518
CONF_REFERENCE_IMAGE_CHECK_INFO_MESSAGE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b42	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

0			10				20				30				40
Flags 1 Byte			**Correlation** 1 Byte				**Threshold** 1 Byte				**SecondsToAlarm** 2 Bytes				

Flags

If no info available may have one of the following reason: reference image check was not enabled or reference image has not enough contours to perform reference image check

Values:

	Mask	Name
Bit 7	0x80	CheckFailed
Bit 6	0x40	NoInfoAvailable

Correlation

Current correlation value between recent image and reference image

Threshold

Threshold value

SecondsToAlarm

Seconds until alarm is triggered

© Copyright 2001-2021 Bosch Security Systems

2.519 CONF_RELAIS_NAME

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0109	relay output	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Read the description for a relay output	
Write	p_unicode	service	Set the description for a relay output (32 unicode characters	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.520 CONF_RELAIS_SWITCH

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0094	relay output	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Return the current relay state	
Write	f_flag	user	Toggle the logical level of a relay output (0->1; 1->0) (notice: a flag payload value is not necessary, i.e. ignored)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.521 CONF_RELAY_OUTPUT_MODE

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bdb	relay output	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	1: output pin is \normally open\"; 2: output pin is \"normally closed \"; (Notice: the relay output mode can also be set via the Alarm Task Script Language)	
Write	t_octet	service	1: set output pin to \normally open \"; 2: set output pin to \"normally closed\"; (Notice: the relay output mode can also be set via the Alarm Task Script Language; should not be used in parallel)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.522 CONF_RELAY_OUTPUT_STATE

[API: I/Os]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01c1	relay output	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Get the logical level of an relay output	
Write	f_flag	user	Set the logical level of an relay output	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.523 CONF_REMOTE_PASSWORD

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x010c	destination IP number	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal		
Write	p_string	service	Deposit the password of the called station	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.524 CONF_REMOTE_PORTAL_INFO

[API: Cbs]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d00	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get information about the camera's Remote Portal connection	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

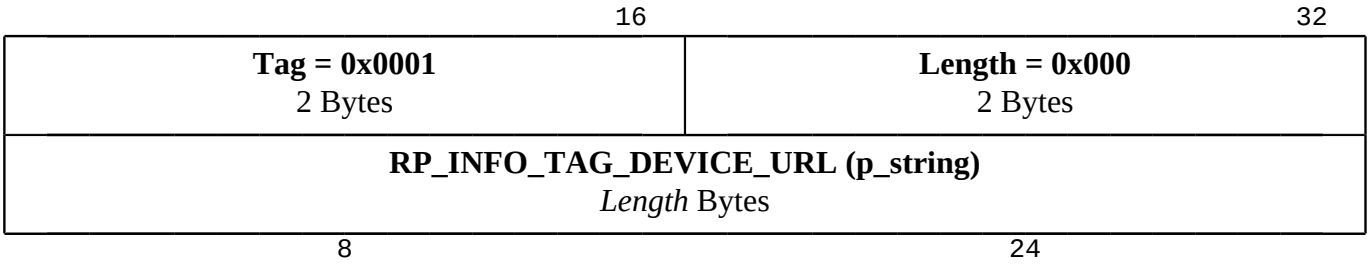
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: RP_INFO_TAG_DEVICE_URL

Direct link to the device in the Remote Portal



© Copyright 2001-2021 Bosch Security Systems

2.525 CONF_REMOTE_REC_DEVICE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a85	yes (1 to max 8) 0 - any free entry	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the ip address or iqn of the remote recording device (max. 50 characters).	
Write	p_string	service	Set the ip(ipv4 only) address or the iqn (starting with \iqn.\" or * for wildcard) of a remote recording device (max. 50 characters) on one entry ore use any free entry by num = 0	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.526 CONF_REQ_FAST_UPDATE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01d3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	minimal	Request a video intra frame, SessionID is mandatory	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.527 CONF_ROI

[API: roi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b48	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	live		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Select region of interest. The Session-ID identifies the video coder.

Payload Structure

16	32
hPos 2 Bytes	**vPos** 2 Bytes
size 2 Bytes	**reserved** 2 Bytes
8	24

hPos

Center position of the cropping window in relative coordinates (0..32768). E.g. (0, 0) is upper left; (16384, 16384) is the center of the original image.

vPos

Center position of the cropping window in relative coordinates (0..32768). E.g. (0, 0) is upper left; (16384, 16384) is the center of the original image.

size

Size of the cropping window in relative coordinates (0..32768). The aspect ratio will be preserved.

reserved

2 Bytes reserved

© Copyright 2001-2021 Bosch Security Systems

2.528 CONF_ROI_OPTIONS

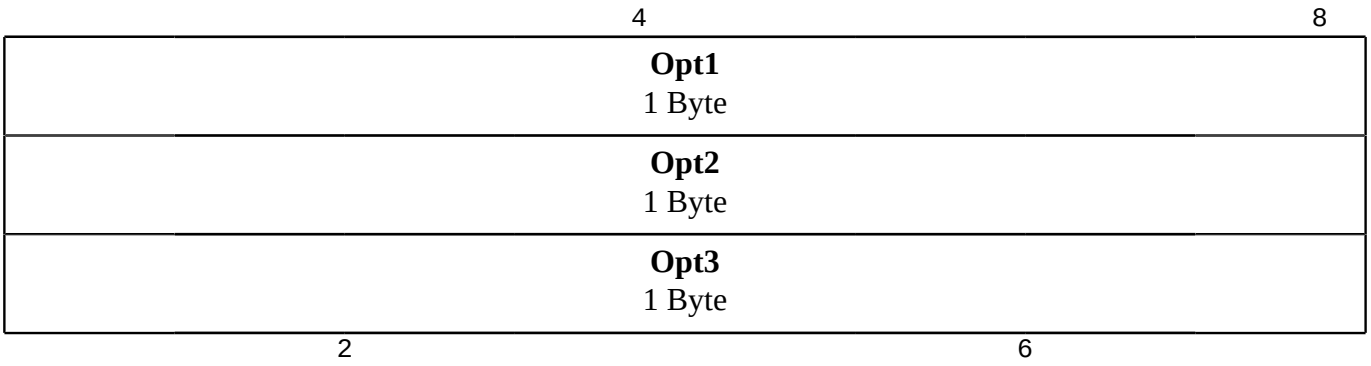
[API: roi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c7e	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Get CONF_VIDEO_STATIC_SCENE_REGIONS encoding options per line.

Payload Structure



Opt1

	Mask	Name	Description
Bit 1	0x02	h.265	Supports static scene options in h.265
Bit 0	0x01	h.264	Supports static scene options in h.264

Opt2

	Mask	Name	Description
Bit 0	0x01	POS	Supports preset pos inside CONF_VIDEO_STATIC_SCENE_REGIONS

Opt3

	Mask	Name	Description
Bit 0	0x01	QP+	Only positive QP offsets at encoder regions

© Copyright 2001-2021 Bosch Security Systems

2.529 CONF_RTSP_PORT

[API: rtsp settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a63	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the local Rtsp port	
Write	t_dword	service	Sets the local Rtsp port #ValueList: 0= Rtsp off	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.530 CONF_RTSPS_PORT

[API: rtsp settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ce6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the local Rtsps port	
Write	t_dword	service	Set the local Rtsps port #ValueList: 0= Rtsp off;	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.531 CONF_RULE_ENGINE_ID

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b94	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Reads the version id of the rule engine	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.532 CONF_RUN_QR_READER

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c3a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	service	Run the QR code reader for a certain time (in seconds). (notice: values 0 and 0xFFFFFFFF not allowed; reserved for future use)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.533 CONF_SAST_CLAIM_DEVICE_URI

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cfb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the url to claim the device	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.534 CONF_SAST_CLOUD_CONNECTION

[API: app management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cfa	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Connected to SAST cloud	
Write	f_flag	service	Connected to SAST cloud	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.535 CONF_SCAN_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c27	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	Store ScanID; used for residential cloud services APP to identify the camera in the Wifi network	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.536 CONF_SCHEDULED_PTZ_PROFILE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c20	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read one scheduled ptz profile, see detailed description	
Write	p_octet	service	Write one scheduled ptz profile, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read/Write one scheduled ptz profiles for a camera, payload upto 12 bytes total. The profiles will be written to the config. Based on the record schedule the profiles will become active and a PTZ/ROI Tour/Preset will be activated or deativated. The flag SCHEDULED_PTZ_PROFILE_FLAG_TOUR, will use a PTZ tour on PTZ domes. If the cam isn't a PTZ camera, ROI is used instead, if the line is capable of it. The preset/tour will be applied on changes only, except the SCHEDULED_PTZ_PROFILE_FLAG_RETRIGGER flag is set, this will repeat the apply periodically. An User will be always able to change the PTZ state on a camera after the scheduled apply. In read direction only the command header (4 bytes) is required. The response will have the full payload size of 12 bytes including the PTZ profile.

Payload Structure

16		32
Camera 2 Bytes	**Reserved** 1 Byte	**Profile Number** 1 Byte
PTZ Profile... (see description)		
PTZ Profile ...		
8		24

Camera

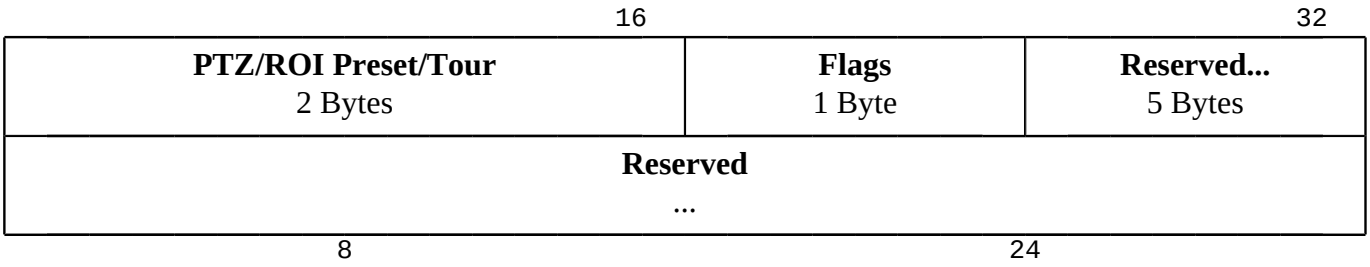
Camara line of the requested PTZ profile (1 to n)

Profile Number

Profile number of the requested PTZ profile (1 to 10)

© Copyright 2001-2021 Bosch Security Systems

PTZ Profile



PTZ/ROI Preset/Tour

Number of a preset or tour for PTZ or ROI depending on the flags settings, if the tourflag is set, following values are defined, 1 - Tour A, 2 - Tour B, 3 - Custom Tour

Flags

	Mask	Name	Description
Bit 1	0x02	RETRIGGER_LOCK	If set, the preset (not tour) setting will be reapplied about every 10th second and ptz from user will be locked for both tour and preset (except bicom aux)
Bit 0	0x01	FLAG_TOUR	If set, the preset number specifies a PTZ tour (only for PTZ domes)

© Copyright 2001-2021 Bosch Security Systems

2.537 CONF_SCHEDULED_PTZ_PROFILES

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c1f	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read scheduled ptz profiles of one cam, see detailed description	
Write	p_octet	service	Write scheduled ptz profiles of one cam, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Read/Write one scheduled ptz profiles for a camera, payload upto 12 bytes total. The profiles will be written to the config. Based on the record schedule the profiles will become active and a PTZ/ROI Tour/Preset will be activated or deativated. The flag SCHEDULED_PTZ_PROFILE_FLAG_TOUR, will use a PTZ tour on PTZ domes. If the cam isn't a PTZ camera, ROI is used instead, if the line is capable of it. The preset/tour will be applied on changes only, except the SCHEDULED_PTZ_PROFILE_FLAG_RETRIGGER flag is set, this will repeat the apply periodically. An User will be always able to change the PTZ state on a camera after the scheduled apply. In read direction only the command header (4 bytes) is required. The response will have the full payload size of 12 bytes including the PTZ profile.

Payload Structure

PTZ Profile [0] (see description)
…
PTZ Profile [9] (see description)

PTZ Profile

16		32
PTZ/ROI Preset/Tour 2 Bytes	**Flags** 1 Byte	**Reserved...** 5 Bytes
Reserved ...		
8		24

© Copyright 2001-2021 Bosch Security Systems

PTZ/ROI Preset/Tour

Number of a preset or tour for PTZ or ROI depending on the flags settings, if the tourflag is set, following values are defined, 1 - Tour A, 2 - Tour B, 3 - Custom Tour

Flags

	Mask	Name	Description
Bit 1	0x02	RETRIGGER_LOCK	If set, the preset (not tour) setting will be reapplied about every 10th second and ptz from user will be locked for both tour and preset (except bicom aux)
Bit 0	0x01	FLAG_TOUR	If set, the preset number specifies a PTZ tour (only for PTZ domes)

© Copyright 2001-2021 Bosch Security Systems

2.538
CONF_SD_CARD_LIFE_SPAN_ALARM_THRESHOLD

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c87	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Read the current alarm threshold when SD-Cards should trigger an EndOfLifespan alarm: 0..100%; 0 = off (Only for SD-Card which supports this feature)	
Write	t_octet	service	Write the current alarm threshold when SD-Cards should trigger an EndOfLifespan alarm: 0..100%; 0 = off (Only for SD-Card which supports this feature)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.539 CONF_SD_CARD_LIFE_SPAN_ENABLE

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c9c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	user	Read enable state for SD-Card live-span information (may be disabled for SD-Card which does not supports this feature)	
Write	f_flag	service	Write enable state for SD-Card live-span information (may be disabled for SD-Card which does not supports this feature)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.540 CONF_SD_CARD_LIFE_SPAN_MANF

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cde	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Read SD-Card manufacturer for SD Lifespan indicator. 0=Autodetect; 1=SANDISK; 2=SONY; 3=MICRON; 4=EMMC_GENERIC	
Write	t_octet	service	Write SD-Card manufacturer for SD Lifespan indicator. 0=Autodetect; 1=SANDISK; 2=SONY; 3=MICRON; 4=EMMC_GENERIC	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.541 CONF_SD_CARD_LIFE_SPAN_STATUS

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c5A	SD card slot ID. Starts with 1.	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: MANF_ID

SD Card manufacturer ID.

Length = 0x0021 2 Bytes	**Tag = 0x0001** 2 Bytes
MANF_ID (t_octet) 1 Byte	

Tag 2: PROD_ID

SD Card product ID.

Length = 0x0022 2 Bytes	**Tag = 0x0002** 2 Bytes
PROD_ID (t_word) 2 Bytes	

Tag 3: PROD_STRING

SD Card product string.

Length 2 Bytes	**Tag = 0x0003** 2 Bytes
PROD_STRING (p_string) *Length* - 4 Bytes	

Tag 4: NUM_BLOCKS

Number of SD Card 512 byte blocks.

Length = 0x0024 2 Bytes	**Tag = 0x0004** 2 Bytes
NUM_BLOCKS (t_dword) 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

Tag 5: ESTIMATED_LIFE_SPAN_PERCENT_DONE

Consumed value of estimated lifespan for SD-Card in percent (only for supported cards)

Length = 0x0021 2 Bytes	**Tag = 0x0005** 2 Bytes
ESTIMATED_LIFE_SPAN_PERCENT_DONE (t_octet) 1 Byte	

Tag 6: MANF_NAME

Manufacturer plain text name (only for supported cards)

Length 2 Bytes	**Tag = 0x0006** 2 Bytes
MANF_NAME (p_string) *Length* - 4 Bytes	

Tag 7: SERIAL_NO

SD-Card serial number

Length = 0x0024 2 Bytes	**Tag = 0x0007** 2 Bytes
SERIAL_NO (t_dword) 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

2.542 CONF_SD_CARD_STATUS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b89	SD card slot ID. Starts with 1.	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the current status of the SD card. BIT 0: 0: no card, 1: card detected. BIT 1: 0: SD, 1: CF. BIT 2..15: reserved.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.543 CONF_SECURITY_COPROC_AUTHENTICATE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b92	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read with incoming payload: send a challenge for device authentication (16 bytes: 8 random bytes + the 8 byte sequence 0x4E, 0x58, 0x50, 0x48, 0x41, 0x2E, 0x30, 0x31), read the 128 bytes challenge response	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.544 CONF_SECURITY_COPROC_CERTIFICATE

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b91	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the security coproc device certificate (X.509-certificate ASN.1-DER coded)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.545 CONF_SECURITY_COPROC_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b93	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the security coproc version	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.546 CONF_SEI_ENABLE

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b6d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	user	Enable/disable SEI: 0=off, 1=on	
Write	t_octet	user	Enable/disable SEI: 0=off, 1=on	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.547 CONF_SENSITIVITY_OBJECT_BASED_VCA

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b31	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Reads the sensitivity of the object based VCA algorithm (of active config profile), if not an object based VCA algorithm is running the command returns with an error	
Write	t_octet	iva	Sets the sensitivity of the object based VCA algorithm in the active config profile, if not an object based VCA algorithm is running the command returns with an error	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.548 CONF_SENSOR_ORIENTATION

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c39	IMU Sensor at image sensor: 0x0001-0x00fe; IMU Sensor at chassis: 0x0101 ... 0x01ff; legacy mapping: 0 -> 1, 0xff -> 0x0101, 0x00ff -> 0x0101	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

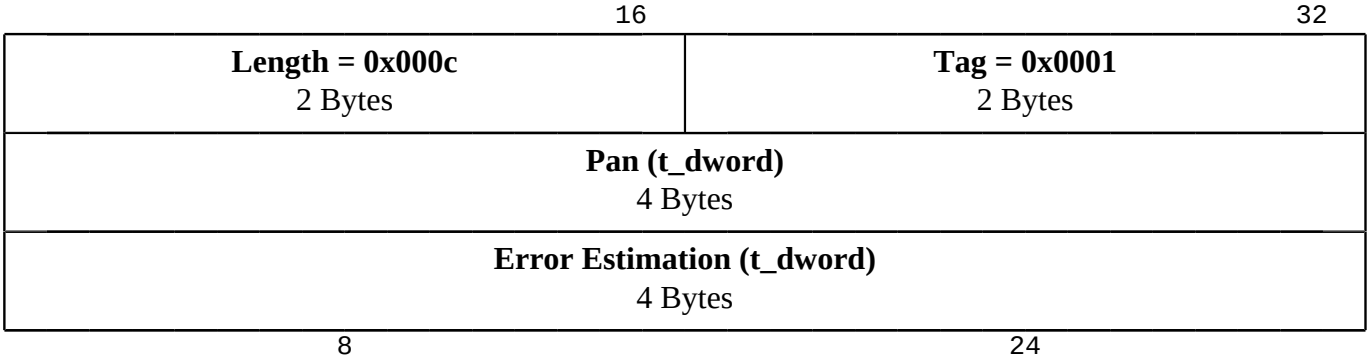
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: Pan

Pan angle (magnetic sensor present).



Pan

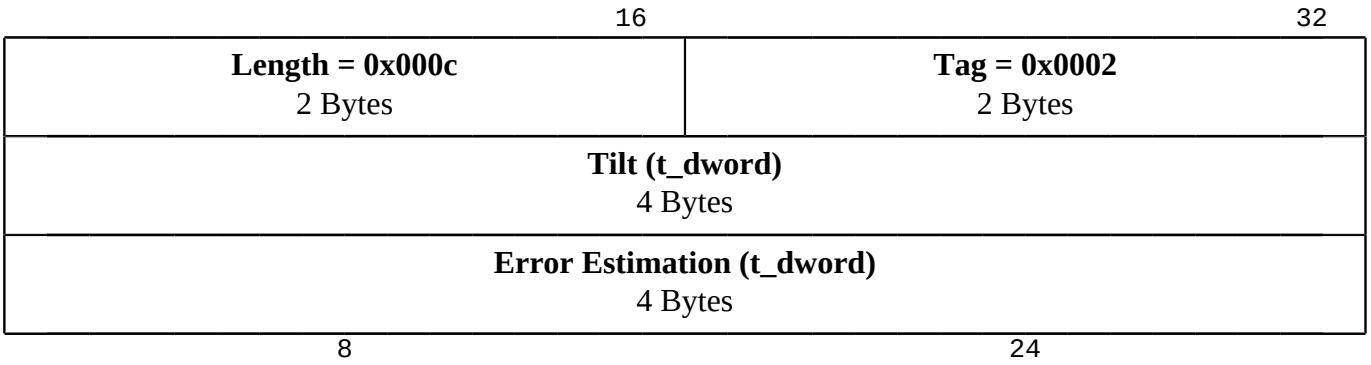
Specifies the pan angle in units of (2*pi/(2^32))

Error Estimation

Specifies the estimation error in units of (2*pi/(2^32))

Tag 2: Tilt

Tilt angle (acceleration sensor present).



Tilt

Specifies the tilt angle in units of (2*pi/(2^32))

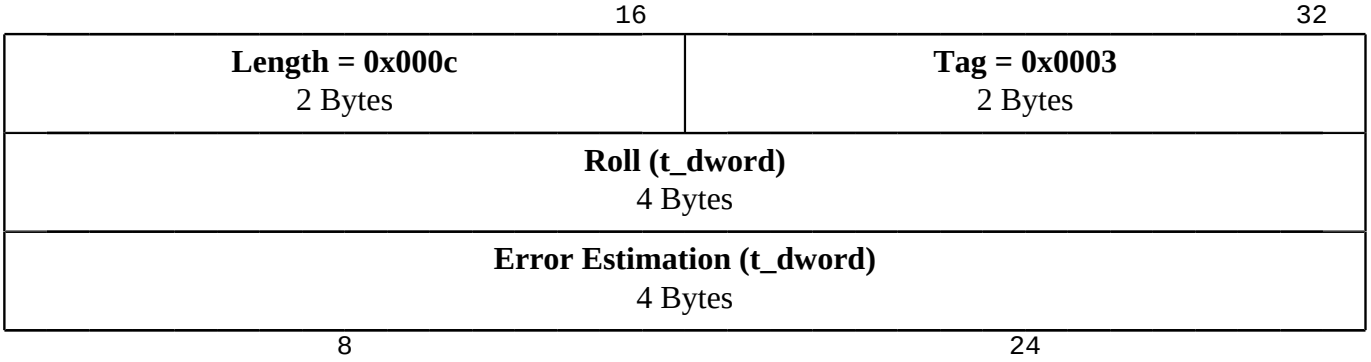
Error Estimation

Specifies the estimation error in units of (2*pi/(2^32))

© Copyright 2001-2021 Bosch Security Systems

Tag 3: Roll

Roll angle (acceleration sensor present).



Roll

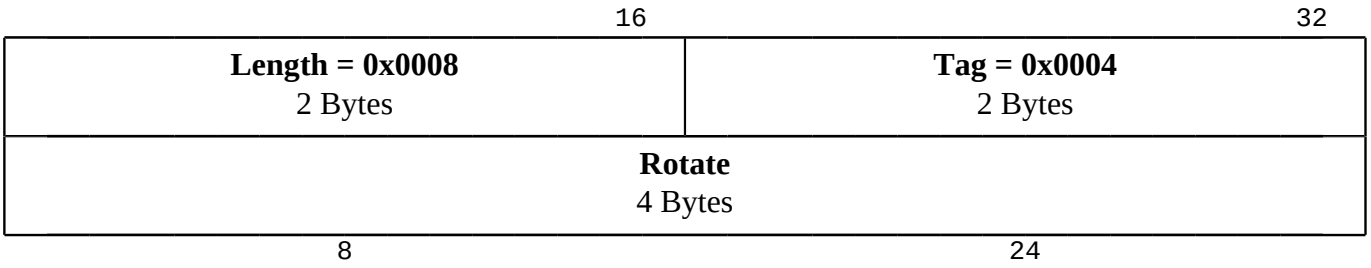
Specifies the roll angle in units of (2*pi/(2^32))

Error Estimation

Specifies the estimation error in units of (2*pi/(2^32))

Tag 4: Rotate

Image Sensor rotation is active (if tag is missing an unrotated image has to be assumed).



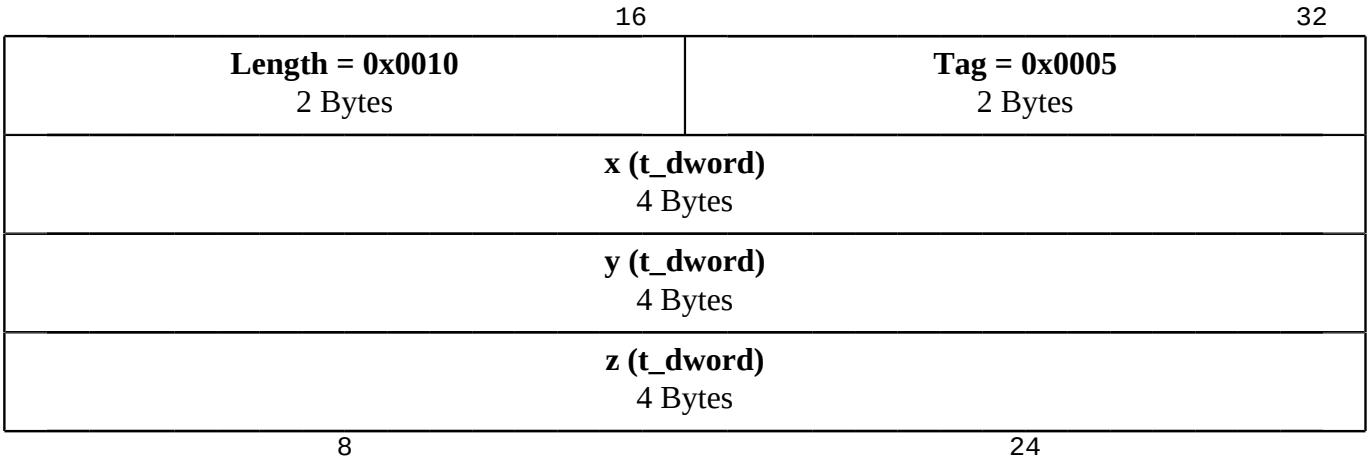
Rotate

Specifies the rotate angle in units of (2*pi/(2^32)). This value can be added to the roll angle.

© Copyright 2001-2021 Bosch Security Systems

Tag 5: Vector

Acceleration vector (acceleration sensor present).



x

Accelaration in x direction in milli g

y

Accelaration in y direction in milli g

z

Accelaration in z direction in milli g

Note: Missing tags are to be considered as unknown (e.g., no sensor available)

© Copyright 2001-2021 Bosch Security Systems

2.549 CONF_SERIAL_ACTS_ACCESS_RIGHTS

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1a	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Return access rights of ACTS/ OSRD commands which are sent via transparent data.	
Write	t_octet	service	Set access rights of ACTS/OSRD commands which are sent via transparent data.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

user rights	0	mainly PTZ) (default
service rights	1	change settings

© Copyright 2001-2021 Bosch Security Systems

2.550 CONF_SERIAL_NUMBER

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy	Returns the serial number of the device. The representation of this serial number is binary octets	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.551 CONF_SERIAL_PORT_APP_VAL

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01f1	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

terminal mode	0	serial port application is in terminal mode
transparent datamode	0xFF	serial port application is in transparent datamode
camera type mode	1	
camera type mode	2	
camera type mode	3	
camera type mode	4	
camera type mode	5	
camera type mode	6	
camera type mode	7	
camera type mode	8	
camera type mode	9	
camera type mode	10	
piped mode	0xF8	serial port application is in a mode where incoming CONF_BICOM_COMMAND are piped to the serial port (0xF8 is only applicable on encoder devices and only for video line 1, for 0xF8 a device reboot is necessary!)

© Copyright 2001-2021 Bosch Security Systems

2.552 CONF_SERIAL_PORT_BITS

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x027f	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

seven data bits	7
eight data bits	8

© Copyright 2001-2021 Bosch Security Systems

2.553 CONF_SERIAL_PORT_HD_MODE_VAL

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x020b	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

halfduplex mode off	0	
halfduplex mode on	1	
halfduplex mode on and buffered data	2	NOT avalibale in conventional VideoJet

© Copyright 2001-2021 Bosch Security Systems

2.554 CONF_SERIAL_PORT_MODE_VAL

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0208	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

port mode RS232	1	
port mode RS422/RS485	2	NOT available in conventional VideoJet

© Copyright 2001-2021 Bosch Security Systems

2.555 CONF_SERIAL_PORT_PAR

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0281	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

no parity	0
even parity	1
odd parity	2

© Copyright 2001-2021 Bosch Security Systems

2.556 CONF_SERIAL_PORT_RATE

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x027e	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Bitrate=300; bitrate=600; bitrate=1200; bitrate=2400; bitrate=4800; bitrate=9600; bitrate=19200	
Write	t_dword	service	Bitrate=300; bitrate=600; bitrate=1200; bitrate=2400; bitrate=4800; bitrate=9600; bitrate=19200	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.557 CONF_SERIAL_PORT_STBITS

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0280	physical port	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

1 modemport stopbit	1
2 modemport stopbits	2

© Copyright 2001-2021 Bosch Security Systems

2.558 CONF_SET_REC_BUFFER_SIZE

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ae1	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get size of recording buffer in bytes (Bosch streaming gateway only) first 4 bytes : DWORD rec_idx second 4 bytes : DWORD size in bytes(return direction)	
Write	p_octet	service	Set size of recording buffer in bytes (restart of bsg nessessary) (Bosch streaming gateway only, default 16 MB) first 4 bytes : DWORD rec_idx second 4 bytes : DWORD size in bytes	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.559 CONF_SET_VIRTUAL_ALARM_ID

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b41	virtual alarm number	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_unicode	service	Set virtual alarm with num	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.560 CONF_SHIFT_CALIBRATION_ELEMENT

[API: calibration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c82	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	4 bytes reserved, 2 bytes shiftx (0x8000 normalized), 2 bytes shifty (0x8000 normalized), calibration element	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.561 CONF_SIGNALLED_POE_CLASS

[API: lldp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca1	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	service	Read the signalled poe class of the device (incl. power adder); 1: class0, 2: class1, 3: class2, etc. 255: no PoE	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.562 CONF_SND_MSS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a02	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Max tcp send mss for all connections	
Write	t_dword	service	Set the global tcp send mss; use this to reduce the max. send segment size for all tcp connections; higher settings than default will only be used by iSCSI connections (0 means default)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.563 CONF_SNMP_READ_COMMUNITY

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b16	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read the community string for snmp read access	
Write	p_string	service	Write the community string for snmp read access	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.564 CONF_SNMP_SERVER_MODE

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c4e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	service	Read the snmp server mode (0=v1 legacy mode; 1=v1; 8=v3; 9=v1+v3)	
Write	t_octet	service	Write the snmp server mode (0=v1 legacy mode; 1=v1; 8=v3; 9=v1+v3); Support for options 1=v1 and 9=v1+v3 removed in V6.50 and later	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.565 CONF_SNMP_SRV_PORT

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a25	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Read the snmp server port	
Write	t_word	service	Set the snmp server port (reset nescessary)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.566 CONF_SNMP_TRAP_COMMUNITY

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b18	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read the community string used in snmp traps	
Write	p_string	service	Write the community string used in snmp traps	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.567 CONF_SNMP_TRAP_LIST

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a11	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List of commands to be posted as snmp trap. See detailed description.	
Write	p_octet	service	List of commands to be posted as snmp trap. See detailed description.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

Trap Descriptor [0] (see description)
...
Trap Descriptor [N] (see description)

Trap Descriptor

16	32
RCP Msg Code 2 Bytes	**Bitmask of Flags** 2 Bytes
8	24

RCP Msg Code

The code of the RCP message that is to be forwarded as SNMP trap.

Bitmask of Flags

SNMP_TRAP_ENABLED 1

© Copyright 2001-2021 Bosch Security Systems

2.568 CONF_SNMP_TRAPS_HOST

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00b6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Not documented	
Write	t_dword	service	Not documented	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.569 CONF_SNMP_TRAPS_HOST_STR

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00b7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Not documented	
Write	p_string	service	Not documented	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.570 CONF_SNMP_USER_PROFILE

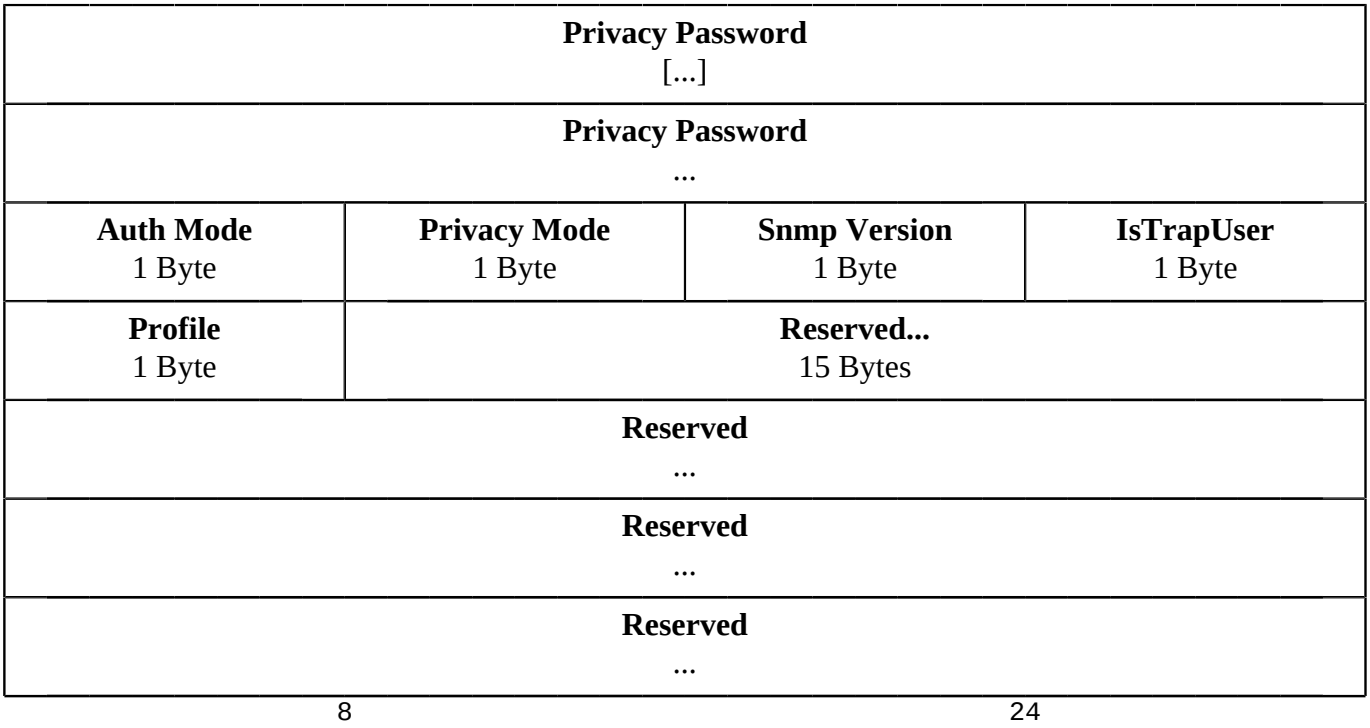
[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c4f	yes 1=(static bound to local service user profile and its auth-password); 3=(profile for a remote account sending SNMP traps to)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Read a snmp (non-legacy) mode user profile	
Write	p_octet	service	Write a snmp (non-legacy) mode user profile	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16	32
User Name... 32 Bytes	
User Name [...]	
User Name ...	
Auth Password... 32 Bytes	
Auth Password [...]	
Auth Password ...	
Privacy Password... 32 Bytes	

© Copyright 2001-2021 Bosch Security Systems



User Name

Zero terminated string with user name. Or zero to delete this user entry.

Auth Password

Zero terminated string with authentication password.

Privacy Password

Zero terminated string with privacy password.

Auth Mode

No authentication	1
MD5 authentication	2
SHA1 authentication	3

Privacy Mode

No privacy	1
DES privacy	2
AES privacy	4

Snmp Version

SNMP version for this user

SNMP Version 3	3

© Copyright 2001-2021 Bosch Security Systems

IsTrapUser

Define if this account is a local account or a account to sent traps to.

Local Account	0
Remote Trap Account	1

Profile

Read Only	0
Read Write	1

© Copyright 2001-2021 Bosch Security Systems

2.571 CONF_SNMP_WRITE_COMMUNITY

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b17	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read the community string for snmp write access	
Write	p_string	service	Write the community string for snmp write access	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.572 CONF_SOCKET_KNOCKER_DESTINATION

[API: Socket Knocker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aee	destination number (starting with 1)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get socket knocker destination and connection configuration. See detailed description for payload structure	
Write	p_octet	service	Specify socket knocker destination and connection configuration. See detailed description for payload structure	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16		32
Port 2 Bytes	**SSL** 1 Byte	**Number of sockets** 1 Byte
Try connect timeout 1 Byte	**Reserved** 3 Bytes	
URL N Bytes		
8	24	

Port

Destination port

SSL

Provide ssl socket:

No SSL	0
SSL	1

Number of sockets

Number of idle sockets to distribute. As soon as a socket is used, a new one is provided.

© Copyright 2001-2021 Bosch Security Systems

Try connect timeout

Timeout in seconds. If this timeout is non-zero, the camera first tries to connect to the new destination for up to the given timeout. The new destination is only stored in the camera's configuration if a connection could be established within this time. If not, the new destinaton will not be stored and the camera will re-connect again to the previous destination.

Reserved

Reserved for future use. Set to zero.

URL

Destination url. Zero-terminated ascii string. Currently max. 128 characters (incl. zero termination).

© Copyright 2001-2021 Bosch Security Systems

2.573
CONF_SOCKET_KNOCKER_DESTINATION_NAME

[API: Socket Knocker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c83	destination number (starting with 1); see CONF_SOCKET_KNOCKER_DE	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the 'friendly name' for a socket knocker destination url (configured by CONF_SOCKET_KNOCKER_DESTINATION Shown to the customer to identify any connected Cloud Service. Max. 64 characters.	
Write	p_string	service	Set a 'friendly name' for a socket knocker destination url (configured by CONF_SOCKET_KNOCKER_DESTINATION Shown to the customer to identify any connected Cloud Service. Max. 64 characters.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.574 CONF_SOCKET_KNOCKER_MODE

[API: Socket Knocker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b5c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get mode of the socket knocker: 0=off, 1=on, 2=auto	
Write	t_dword	service	Switch socket knocker on/off: 0=off, 1=on, 2=auto	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.575 CONF_SOCKET_KNOCKER_STATUS

[API: Socket Knocker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b98	destination number (starting with 1)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the current status of the socket knocker. See detailed description for payload structure	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16		32
State 1 Byte	**Reason** 1 Byte	**Reserved** 2 Bytes
8		24

State

Current status of socket knocker

Not running	0	
Trying to connect	1	
Connected	2	(ready for cloud service)

Reason

Reason for current state

As expected	0
Unknown	1
Auto mode: DHCP off or not successful	2
Auto mode: max knocking attempts reached	3
Auto mode: max knocking time reached	4
URL could not be resolved	5
Destination not responding	6

© Copyright 2001-2021 Bosch Security Systems

2.576 CONF_SOFT_VARIANT_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb2	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the soft variant Id if set; (notice: if no soft variant ID is set, the 'original' variantID will be returned)	
Write	t_dword	service	Set a soft variant ID. Only possible on certain devices. For a list of allowed soft variants see CONF_SOFT_VARIANT_ID_OPTIONS. All configuration except IP/subnet mask and DHCP settings will be set to default and the device reboots automatically. (Notice: only for some special soft variants which have the option flag 'no factory defaults performed' (see CONF_SOFT_VARIANT_ID_OPTIONS), the device configuration will be kept and not set to defaults.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.577 CONF_SOFT_VARIANT_ID_OPTIONS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get a list of possible soft variant IDs plus the according variant name.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16		32
Number of variants 1 Byte	**Flags** 1 Byte	**reserved** 2 Bytes
Variant Description [0] (see description)		
...		
Variant Description [Number of variants - 1] (see description)		
8		24

Number of variants

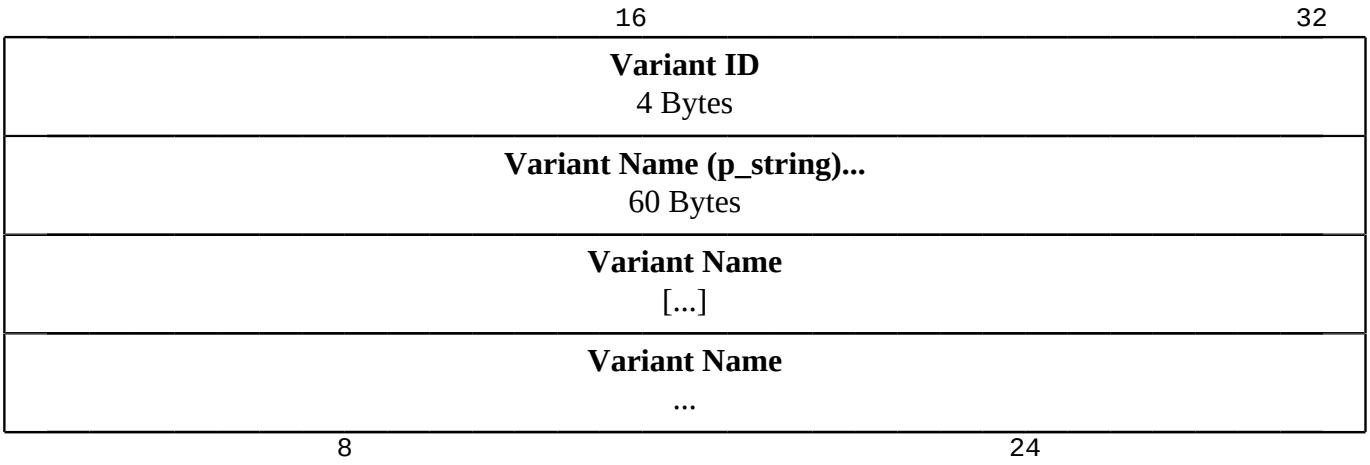
The number of software variants described in this payload.

Flags

	Mask	Name
Bit 0	0x01	No 'factory defaults' performed

© Copyright 2001-2021 Bosch Security Systems

Variant Description



Note: If no soft variant IDs are possible, an empty payload will be returned.

© Copyright 2001-2021 Bosch Security Systems

2.578 CONF_SOFTWARE_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string t_dword	always	Read the software version	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.579 CONF_SOFTWARE_VERSION_FORMATTED

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the software version in the form <major>.<minor>.<build>	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.580 CONF_SPAN_ADDRESS_LIST

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e7	1 - primary spanlist, 2 - secondary spanlist	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The reply to an read request always contains up to n (n=512) entrys. The response in case of an empty span list is one zeroed span entry.

On a write request, a list of max n (n=512) span addresses must be supplied. List entrys that are zero are not used by the unit for recording. The unit uses the supplied ordering of the list and begins with the first entry. The order of the supplied list is the preferred order, not the guaranted order. There are some cases, in which the device changes the use order (e.g. one target not reachable).

If the device is already recording on spans (a span list is already present on the device), the new list must contain the span addresses, that the unit is currently recording on. The latter information can be obtained by reading the current span address list form the unit and check the corresponding write lock header (if the units IP and MAC address is found this means the unit is recording). If the unit receives a list and the span addresses of the spans the device is currently recording on is not present, the list is rejected (not saved in the configuration) and an error is returned.

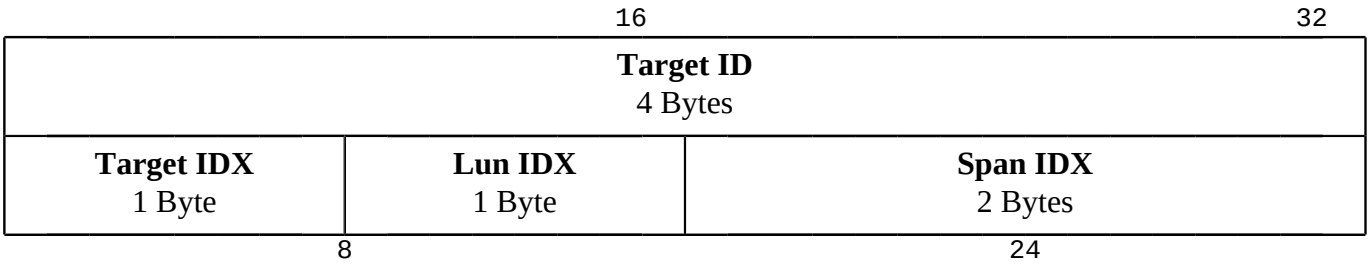
A message of this cmd tag will be send always, if somone sets the span address lists by using rcp set spans list commands or if the span list is cleared by indirectly by changing other settings of the device (e.g. rec mgmt).

Payload Structure

Span Address [0] (see description)
...
Span Address [N] (see description)

© Copyright 2001-2021 Bosch Security Systems

Span Address



Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

Target IDX

Index of the iSCSI server target. The index of the desired target can be obtained from the reply to an iSCSI discover.

Lun IDX

Index of the lun of the specified target.

Span IDX

Index of the span in the specified lun.

Command Specific Errors

SPAN_ERR_INTERNAL	0x01
SPAN_ERR_INV_ADDR_LIST	0x04

Note: A list of all defined error types can be found in the Appendix.

© Copyright 2001-2021 Bosch Security Systems

2.581 CONF_SPAN_ADDRESS_LIST_NEW

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a52	1 - primary spanlist, 2 - secondary spanlist	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	See detailed description of CONF_SPAN_ADDRESS_LIST	
Write	p_octet	service	See detailed description of CONF_SPAN_ADDRESS_LIST (no need to include mounted spans in list)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.582 CONF_SPAN_CERTIFICATES_LIST

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c11	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Read certificates list from a span, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This Command can be used to query the certificates of a span, in order to verify the signed video record data within that span. The response will deliver all certificates of the specified span and time intervall. The maximum size of the response can be limited by the caller but it will not exceed 16 kb for the certificates list. If the response payload size isnt't enough to hold all valid certificates, a flag in the response will signal the existence of further certificates, which didn't fit in the response payload. A second query in that case with a smaller time interval may help to get the remaining certificates. The response will have a certificates list with several enries of different size, there are no gaps between the entries.

Request Payload Structure

16		32
Target Id 4 Bytes		
Target IdX 1 Byte	**LUN** 1 Byte	**Span IDX** 2 Bytes
Start Time 4 Bytes		
End Time 4 Bytes		
TZ QH 1 Byte	**Reserved** 3 Bytes	
Max List Len 4 Bytes		
8		24

Target Id

Target ID of the Span

© Copyright 2001-2021 Bosch Security Systems

Target IdX

Target Index of the Span

LUN

LUN of the Span

Span IDX

Span Index of the Span

Start Time

Start time of the interval in seconds since 2000 local time based on the Timezone offset in TZ QH

End Time

End time of the interval in seconds since 2000 local time based on the Timezone offset in TZ QH

TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Max List Len

Maximum size of the certificates list in response payload in bytes.

Response Payload Structure

16		32
Target Id 4 Bytes		
Target IdX 1 Byte	**LUN** 1 Byte	**Span IDX** 2 Bytes
Start Time 4 Bytes		
End Time 4 Bytes		
TZ QH 1 Byte	**Flags** 1 Byte	**Reserved** 2 Bytes
List Len 4 Bytes		
Certificate [0] (see description)		

© Copyright 2001-2021 Bosch Security Systems

Target Id

Target ID of the Span

Target IdX

Target Index of the Span

LUN

LUN of the Span

Span IDX

Span Index of the Span

Start Time

Start time of the interval in seconds since 2000 local time based on the Timezone offset in TZ QH

End Time

End time of the interval in seconds since 2000 local time based on the Timezone offset in TZ QH

TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Flags

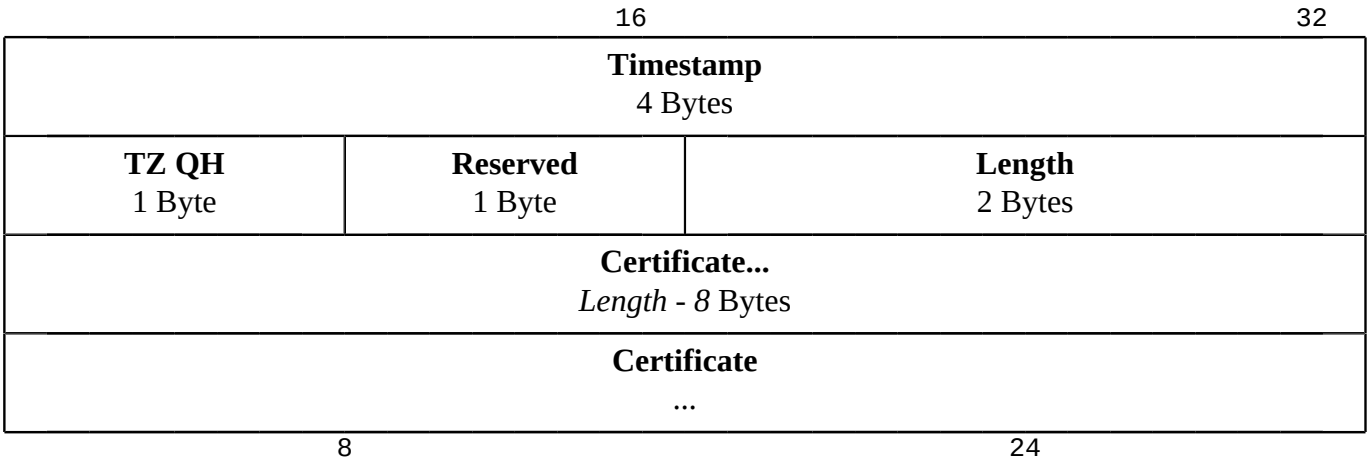
	Mask	Name	Description
Bit 0	0x01	Len Exeeded	More certificates available, but the max length of the certificates list was exceeded.

List Len

Size of the certificates list in response payload in bytes.

© Copyright 2001-2021 Bosch Security Systems

Certificate



Timestamp

Local time in seconds since 2000

TZ QH

Timezone offset (from utc) in quarter hours as signed char value

Length

Length of the Certificates List Entry (certificate including these 8 bytes header infos)

Certificate

One certificate

© Copyright 2001-2021 Bosch Security Systems

2.583 CONF_SPAN_FILES_DIR

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aa1	entry index 1 - 32768	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the path entry end exported iscsi addr. (gen dll only)	
Write	p_octet	service	Set path to span files (gen dll only).	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

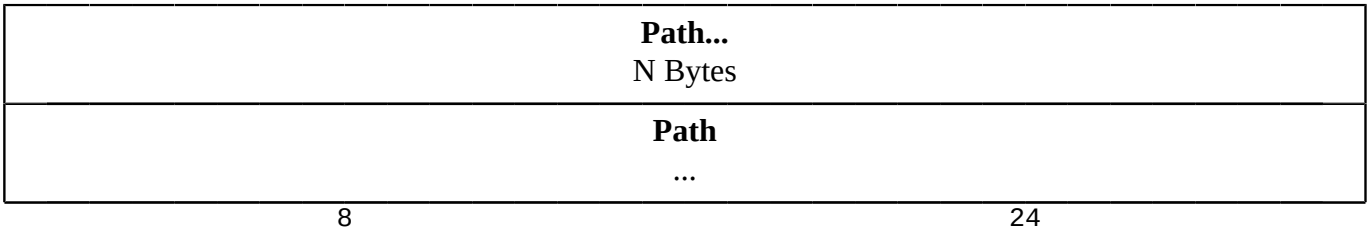
This command allows to set the path to a mounted span formatted storage or directory. You have to send the in payload to set the path. The min payload size is 16 bytes. The max path name length is 1024 bytes including the zero termination. payload size must be set to the size that includes the zero termination but min size is 16 bytes even in case of an empty string (at least zero termination). You can set up to 32768 pathes. Use the num param to specify the index of an entry. If you set the path it will check the path. To pass this check, at least a valid Lun info file (INFO.TXT) has to exist. The span are needed later when accessing the storage(e.g header access, replay). The response to the write direction is has the out payload format. If success the error field is zero and the payload includes the set path and the exported lun address. This storage is now accessable by using this address for other commands. if the set failes, the error field is non zero and contains an error code. It is allowed to set path on an index that was already set. The old values will be overwritten, but remember that the effect takes place later, if there are still open references based on the old entry (e.g. running replay, header access).

The cmd can be used for read direction. The response payload contains the export lun address and the path. error code is always zero. If the entry is empty, the path is empty and has only the zero termination.

Write Payload Structure

16	32
Action 4 Bytes	
Reserved... 8 Bytes	
Reserved ...	

© Copyright 2001-2021 Bosch Security Systems



Action

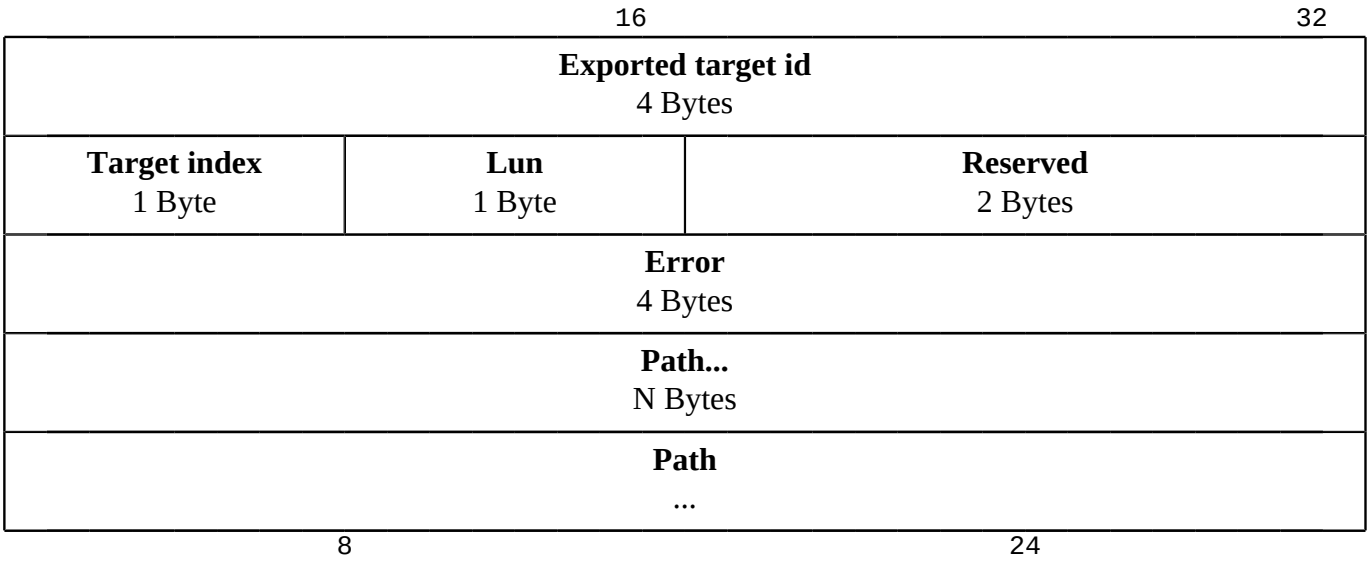
Specifies the action, you can set an entry or clear an entry.

Clear entry	0
Set entry	1

Path

Path to a mounted span formatted storage or directory containing the span files.

Response/Read Payload Structure



Exported target id

Target id of the exported lun, specified by the directory path.

Target index

Target index of the exported lun, specified by the directory path.

Lun

Lun id of the exported lun, specified by the directory path.

© Copyright 2001-2021 Bosch Security Systems

Error

Error code, in case of set path failed. Is always 0 in the payload of read direction. Only relevant for return payload of write direction.

No error	0
Invalid directory path	1
Invalid lun info file	2
Common error	3

Path

Path to a mounted span formatted storage or directory containing the span files.

© Copyright 2001-2021 Bosch Security Systems

2.584 CONF_SPAN_HDR_ACCESS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09e8	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Access span header information.	
Write	p_octet	service	Access span header information.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

LOCK HEADER	0x01
MANAGER HEADER	0x02
UNIT HEADER	0x03

Payload Structure

16	32
Span Header (see description)	
8	24

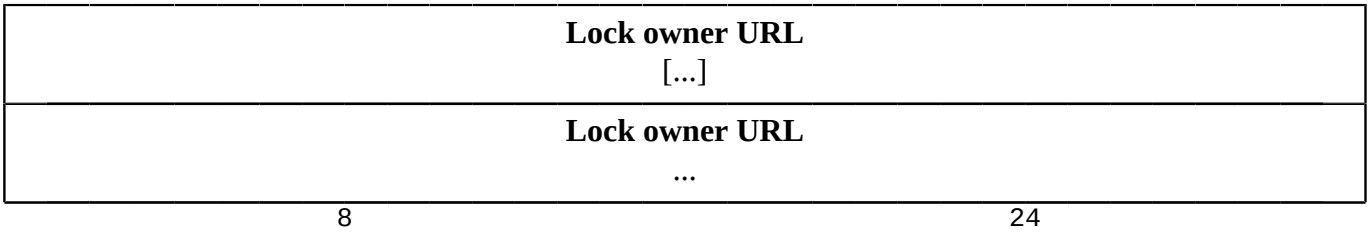
Span Header

Data of the span header. Length and structure differs for the specified header type. The header type is specified with the numeric descriptor.

'Span Header' payload for numeric descriptor = 'Lock Header' (0x01)

16		32
Lock owner IP 4 Bytes		
Lock owner MAC... 6 Bytes		
Lock owner MAC ...	**Cluster Id** 1 Byte	**Reserved** 1 Byte
Lock owner URL... 128 Bytes		

© Copyright 2001-2021 Bosch Security Systems



Lock owner IP

IP address of the unit that holds the write lock of the span.

Lock owner MAC

Hardware address of the unit that holds the write lock of the span.

Cluster Id

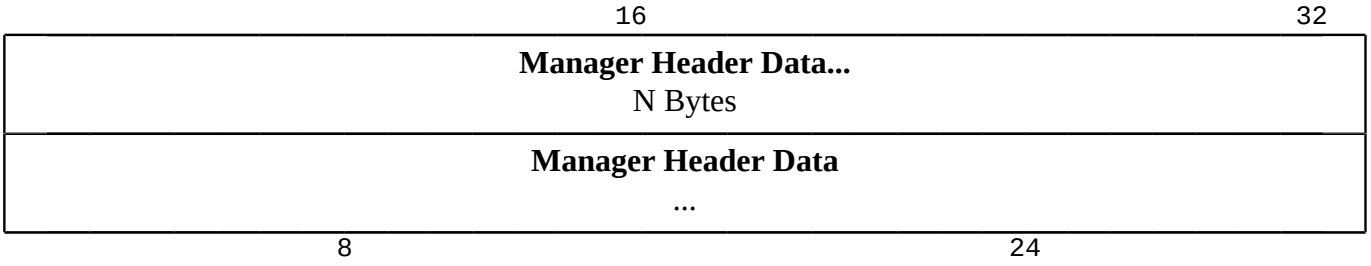
Cluster id of the recorder 1 - n, default value is 0 if not set, should be treated as cluster id 1.

Lock owner URL

Url (ipv4 or ipv6) of the unit that holds the write lock of the span.

Note: In the request packet, the header values are ignored (may be zero). The reply contains the values read or written. On write requests, the local IPv4 and MAC addresses are converted to ascii strings and written to the file LCKxxxxx.txt (xxxx = span index) on disk. On read requests, the ascii strings of that file are scanned into binary values and send with the reply.

'Span Header' payload for numeric descriptor = 'Manager Header' (0x02)



Manager Header Data

Transparent header data of the storage manager

Note: The reply always returns the 1024 Bytes of the manager header file MGRxxxx.txt (xxxx = span index).

© Copyright 2001-2021 Bosch Security Systems

'Span Header' payload for numeric descriptor = 'Unit Header' (0x03)

<table>
<tr><th colspan="3">16 32</th></tr>
<tr><td colspan="3">Span owner IP
4 Bytes</td></tr>
<tr><td colspan="3">Span owner MAC...
6 Bytes</td></tr>
<tr><td colspan="2">Span owner MAC
...</td><td>Span owner Camera
2 Bytes</td></tr>
<tr><td colspan="3">Retention Time
4 Bytes</td></tr>
<tr><td colspan="3">Modification seal random
4 Bytes</td></tr>
<tr><td colspan="3">Modification seal time
4 Bytes</td></tr>
<tr><td>Recording
1 Byte</td><td>Cluster Id
1 Byte</td><td>Reserved
2 Bytes</td></tr>
<tr><td colspan="3">Max Retention Time
4 Bytes</td></tr>
<tr><td colspan="3">Span owner URL...
128 Bytes</td></tr>
<tr><td colspan="3">Span owner URL
[...]</td></tr>
<tr><td colspan="3">Span owner URL
...</td></tr>
<tr><td colspan="3">Modification MAC...
6 Bytes</td></tr>
<tr><td colspan="2">Modification MAC
...</td><td>Reserved
2 Bytes</td></tr>
<tr><td colspan="3">User Data...
16 Bytes</td></tr>
<tr><td colspan="3">User Data
...</td></tr>
<tr><td colspan="3">User Data
...</td></tr>
<tr><td colspan="3">User Data
...</td></tr>
</table>

© Copyright 2001-2021 Bosch Security Systems

GUID... 32 Bytes
GUID [...]
GUID ...

8 24

Span owner IP

IP address of the unit that holds the write lock of the span.

Span owner MAC

Hardware address of the unit that holds the write lock of the span

Span owner Camera

The index of the camera the unit uses for recording.

Retention Time

Retention Time for the recordings on this span in sec since 2000

Modification seal random

Random number set on the last modifacation of this span, this random will be set on each write access on any span header or on starting or stopping a recording of a device on this span, there will be also updates while a device is recording in a period of several minutes

Modification seal time

This information is the time of the last modification on this span that causes a modification seal update

Recording

Primary Recording	1	(default)
Secondary Recording	2	

Cluster Id

Cluster id of the recorder 1 - n, default value is 0 if not set, should be treated as cluster id 1

Max Retention Time

Max Retention Time for the recordings on this span in sec since 2000, 0 means no max retention time

Span owner URL

Url (ipv4 or ipv6) of the unit that used the span

© Copyright 2001-2021 Bosch Security Systems

Modification MAC

Hardware address of the instance, which modifies the unit header (must be non zero for write direction)

User Data

Byte field for any user data (16 bytes)

GUID

GUID (managed by vrm) of the device which recorded on this span (32 bytes)

Note: In the read request packet, the header values are ignored (may be zero). In the write requst packet, only the camera field is evaluated, the values for IP and MAC are taken from local configuration. The reply packet contains the values read or written. On write requests, the local IPv4 and MAC addresses and the supplied camera value are converted to ascii strings and written to the file UNTxxxxx.txt (xxxx = span index) on disk. On read requests, the ascii strings of that file are scanned into binary values and send with the reply. The modification seal values cannot be set by this command as write request directly, they will be set automatically on span header write requests or on starting or stopping a recording. The content of this fields will be ignored in a write request.

Command Specific Errors

SPAN_ERR_INTERNAL	0x01
SPAN_ERR_INV_SPN_IDX	0x02
SPAN_ERR_INV_HDR_TYPE	0x03
SPAN_ERR_NOT_MNTD	0x05
SPAN_ERR_INV_FS	0x06
SPAN_ERR_INV_LUN_NFO	0x07
SPAN_ERR_BAD_HDR_CKSM	0x08
SPAN_ERR_RD_ONLY	0x0a
ISCSI_ERROR_CONNECT	0x31
ISCSI_ERROR_INV_LUN	0x33
ISCSI_ERROR_LOGIN	0x34
ISCSI_ERROR_INV_TRG_IDX	0x35

Note: A list of all defined error types can be found in the Appendix.

© Copyright 2001-2021 Bosch Security Systems

2.585 CONF_SPAN_HISTORY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ace	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read span history	
Write	p_octet	service	Clear span history or add a span history entry	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used by the device to store the span history in the device config, that means it adds (action add) an span adress entry to the config, for each recorded span. The history can be seen by read commando and action show. The history can be cleared by sending a action clear command.

Payload Structure

16	32
cam 2 Bytes	**rec idx** 2 Bytes
action 2 Bytes	**span count** 2 Bytes
span address [0] (see description)	
...	
span address [span count - 1] (see description)	
8	24

cam

Camera line from 1 to max cam

rec idx

Primary Recording	1	(default)
Secondary Recording	2	

© Copyright 2001-2021 Bosch Security Systems

action

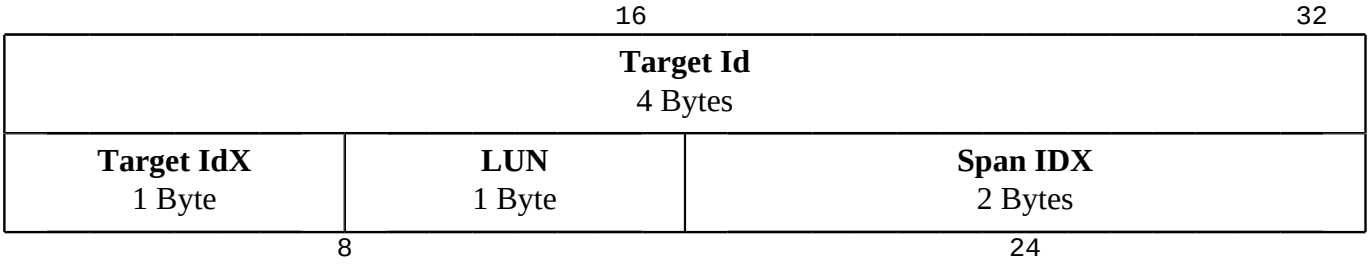
Action to do in this command.

Show	0	Show the span history, read direction only.
Add	1	Add a new entry to the history, write direction only.
Clear	2	Clear the complete history, write direction only.
Show Remount	3	Show the remount span, read direction only.

span count

Number of span address following in this command payload. For action show you can specifie the max number of span history entries to show for in payload. the reply payload will then return a max. of this number of span address. max 32.

span address



Target Id

Target ID of the Span

Target IdX

Target Index of the Span

LUN

LUN of the Span

Span IDX

Span Index of the Span

© Copyright 2001-2021 Bosch Security Systems

2.586 CONF_SPAN_PARTITION_FILE_INFO

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a2d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the file info of a span.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure

16		32
Target ID 4 Bytes		
Target IDX 1 Byte	**LUN** 1 Byte	**Span IDX** 2 Bytes
Start Time (optional) 4 Bytes		
Stop Time (optional) 4 Bytes		
MaxEntries (optional) 4 Bytes		
Optional flags (optional) 4 Bytes		
8		24

Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES).

Target IDX

Index of the iSCSI server target. The index of the desired target can be obtained from the reply to an iSCSI discover.

LUN

The logical unit.

© Copyright 2001-2021 Bosch Security Systems

Span IDX

Index of the span in the specified lun.

Start Time (optional)

Seconds since 2000, optional, if missing, default is 0

Stop Time (optional)

Seconds since 2000, optional, if missing, default is unlimited

MaxEntries (optional)

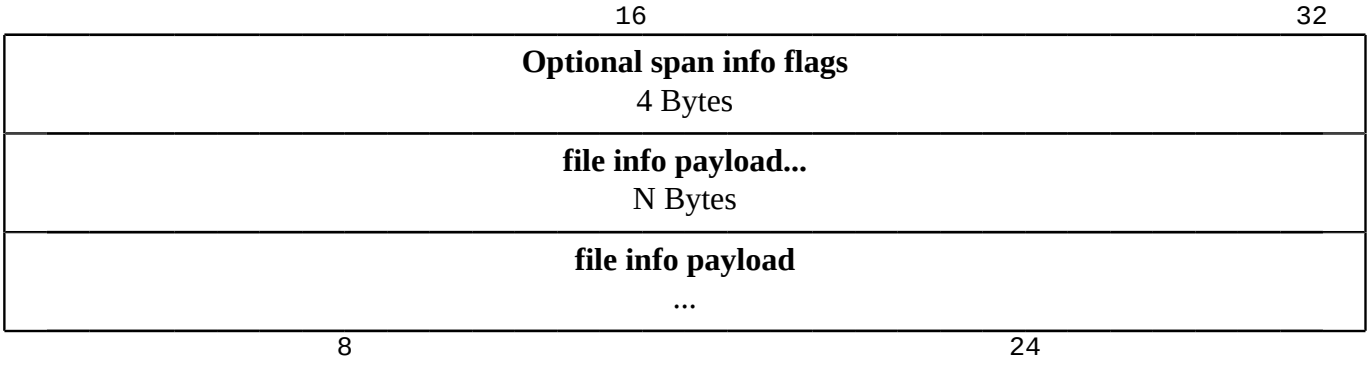
Max Number of entries, optional, if missing, default is max. 256 files which is also the max. limit for this value

Optional flags (optional)

Additional options for the file info

	Mask	Name
Bit 0	0x00000001	Add span info flags

Response Payload Structure



For the payload structure of the response see documentation of HD_PARTITION_FILE_INFO. In case of the optional "add span info flag" flag the first four bytes of the responspayload will contain a flag field with additional infos. Without the option, the payload will start with the file info without offset.

Optional span info flags

Additional info of the span

	Mask	Name
Bit 0	0x00000001	Encrypted span key list present

© Copyright 2001-2021 Bosch Security Systems

2.587 CONF_SPAN_PARTITION_PROP

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09fa	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Return partition properties for primary span recording (4 bytes) video type (1 byte: 0= NO, 1=MPEG2, 3=MPEG4), audio type (1 byte: 0= NO, 1=G711, 2=L16, 3=AAC) reserved (2 byte)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.588 CONF_SPAN_PARTITION_PROP_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a4b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Return partition properties for secondary span recording (4 bytes) video type (1 byte: 0= NO, 1=MPEG2, 3=MPEG4), audio type (1 byte: 0= NO, 1=G711, 2=L16, 3=AAC) reserved (2 byte)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.589 CONF_SPAN_SWITCH

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a53	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Not supported	
Write	p_octet	service	Switch spans, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command can be used to influence the span usage of the recording. It is able to cause a replace of the premounted span and or switch to the next premounted span.

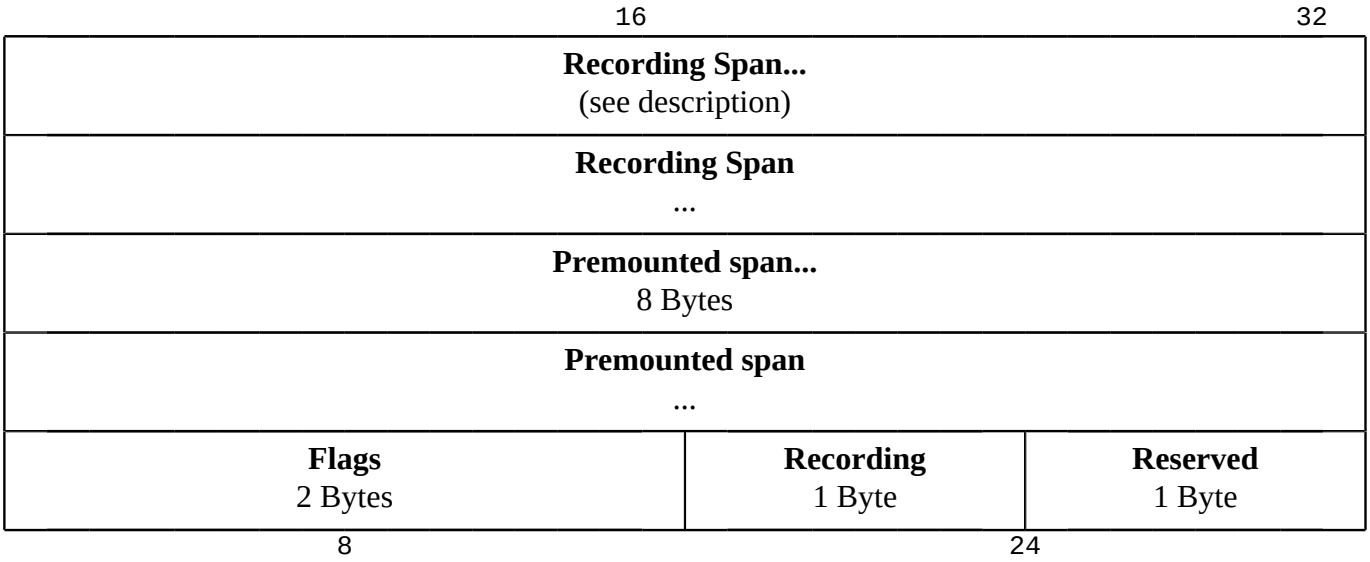
The flag SWITCH_SPAN causes the switch to the next span and releasing the actual recording span. Flag REPLACE_PREMOUNTED will cause the release of the premounted span and mounting another one if available. If both flags are activated, the premounted block will be replaced, as soon as a new premounted span is mounted and available, the next step switch to premounted span will be performed. Before replacing the premounted span, the actual recording span will be checked, if the replace could get in conflict with a device internal triggered span switch. In that case the whole Span switch command will fail with rcp error RCP_ERROR_TRY_LATER, no switch or replace will be performed.

When performing a switch to next span job, the recording span address in the payload will be checked. If on execution of this job the recording span doesn't match the recording span from the payload, the switch will not be performed. When performing a replace premounted span job, the premounted span address in the payload will be checked. If on execution of this job the premounted span doesn't match the premounted span from the payload, the replace will not be performed. The idea of these checks is to avoid conflicts with the automatic span switching of the recording, that can lead to gaps in the recording and unwanted waste of recording spans. The flag FORCE will skip these checks. Flag CHECK_PREMOUNTED is only relevant for the switch to next span job without replacing the premounted span and without FORCE flag. If this flag is active on switching ton next span, the premounted span parameter in the payload will be matched against the actual premounted span. If no match the switch won't be performed. If this flag including the FORCE flag isn't set on switch, a negative match will be performed on the premounted span. The TARGET_RETREAT Bit means, than a complete retreate from this target is intended, the span history will be modified in order to avoid a remount on that target. That state will stay active for that target until the device records on a new target and adds a span from the new target to the history, or if the span history will be cleared or the the target will be reintroduced with the bit TARGET_REINTRODUCE. The TARGET_REINTRODUCE Bit is needed to undo a TARGET_RETREAT action, which is specialy in the case needed, when a device shall return to a target, from that it had retreated erlier and no recording took place on another target between these events. In that case the remount feature for that target is still disabled. The TARGET_REINTRODUCE Bit will reenable the remount feature for that target without the nessessarity of clearing the whole span history.

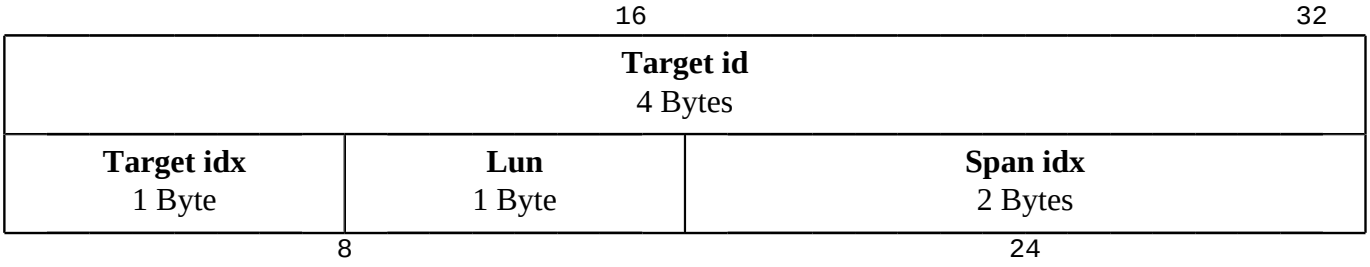
Warning: A possible conflict, when requesting a replace of the premounted span with an automatic span switch of the recording that causes gaps in the recording is unavoidable. So an excesive usage of replacing premounted spans should be avoided.

© Copyright 2001-2021 Bosch Security Systems

Payload Structure



Recording Span



Target id

The target id of the span

Target idx

The target index of the span

Lun

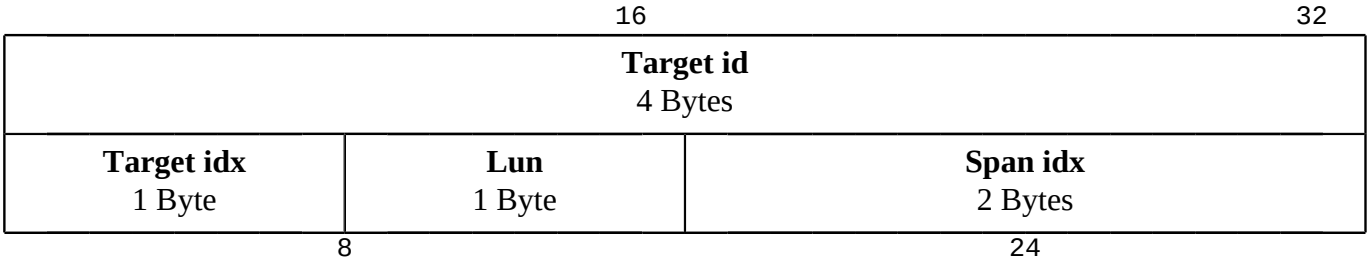
The lun of the span

Span idx

The span index of the span

© Copyright 2001-2021 Bosch Security Systems

Premounted span



Target id

The target id of the span

Target idx

The target index of the span

Lun

The lun of the span

Span idx

The span index of the span

Flags

These flags are used to specify the behavior of the command

	Mask	Name
Bit 15	0x8000	ERROR_CHECK_RECORDING
Bit 14	0x4000	ERROR_CHECK_PREMOUNTED
Bit 5	0x0020	TARGET_REINTRODUCE
Bit 4	0x0010	TARGET_RETREAT
Bit 3	0x0008	SWITCH_SPAN
Bit 2	0x0004	REPLACE_PREMOUNTED
Bit 1	0x0002	CHECK_PREMOUNTED
Bit 0	0x0001	FORCE

Recording

Primary Recording	1	(default)
Secondary Recording	2	

© Copyright 2001-2021 Bosch Security Systems

2.590 CONF_SPAN_USE_STATUS

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09f8	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Usage status of a span, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This messages reports the use status of a span for this device and which cammera for primary recording. If a span was opened for a planned recording, this message will be send with status MOUNTED. If the mounting fails status will be ERROR. If a span was closed after recording status will be RELEASE, if it was closed before a recording ever took place, status will be RELEASE_UNUSED. HD_SPAN_USE_STATUS_RETENTION_TIME status will be send as message if the first time an retention time update took place on the span. In case of a read request the reply payload cantains this status message upto twice. the command can be used to query the actual used spans from a camera specified by the num parameter. the status for a span can be RECORDING, if the cam is actual recording on this span or MOUNTED, if the span is mounted and prepared for a future recording by this camera.

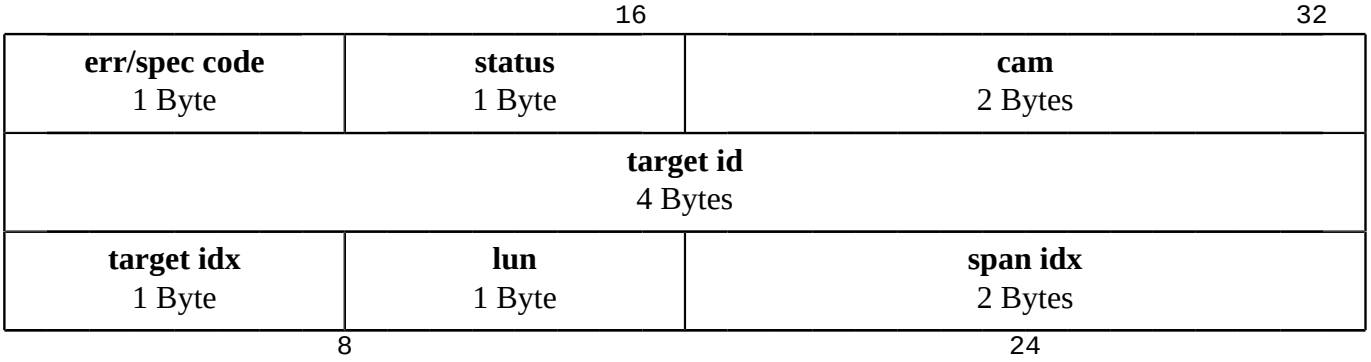
Payload Structure

Use Status Message [0] (see description)
...
Use Status Message [N] (see description)

Use Status Message

Up to 2 span use status messages. If no span is in use by the camera specified by num, the reply payload size will be 0. In case of the rcp message the payload contains always 1 span use status message.

© Copyright 2001-2021 Bosch Security Systems



err/spec code

Additional error or status code to the status, for status HD_SPAN_USE_STATUS_ERROR_MOUNT see section 'status HD_SPAN_USE_STATUS_ERROR_MOUNT error codes', for status RELEASE and RELEASE_UNUSED see section 'status RELEASE error codes', for status MOUNTED see section 'status MOUNTED special codes'

status

Use status of this span

MOUNTED	0x00
RELEASE	0x01
RELEASE_UNUSED	0x02
ERROR	0x03
RECORDING	0x04
PENDING_SPAN_REQUEST	0x05
HD_SPAN_USE_STATUS_RETENTION_TIME	0x06
HD_SPAN_USE_STATUS_RELEASE_REMOUNT_INTENDED	0x07
HD_SPAN_USE_STATUS_ERROR_WRT_UNT_HDR	0x10
HD_SPAN_USE_STATUS_ERROR_FORMAT_REC_REGION	0x11
HD_SPAN_USE_STATUS_ERROR_MOUNT	0x12
HD_SPAN_USE_STATUS_ERROR_RETENTION_TIME	0x13

If the status is PENDING_SPAN_REQUEST, the ip will be 255.255.255.255, target index and lun will be 255, and span index will be 65535, which should be interpreted as still unknown, because this cam had requested a span, but it doesn't know yet which span it gets.

Status HD_SPAN_USE_STATUS_RELEASE_REMOUNT_INTENDED means, the cam has released the block but has left the span lock, because it will try to remount the span later. The span is stored in the span history for remount(see CONF_SPAN_HISTORY).

cam

Which camera is or was using the reported span

© Copyright 2001-2021 Bosch Security Systems

target id

Target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES)

target idx

Target Index of the storage device from 0 to 255.

lun

Lun of the storage device from 0 to max. 255.

span idx

Index of the span on the storage device.

Command Specific Errors

Specific errors for 'status' = 'HD_SPAN_USE_STATUS_ERROR_MOUNT' (0x12)

None	0x00
SPAN_ERROR_INTERNAL	0x01
SPAN_ERROR_INV_SPN_IDX	0x02
SPAN_ERROR_INV_HDR_TYPE	0x03
SPAN_ERROR_INV_ADDR_LIST	0x04
SPAN_ERROR_NOT_MNTD	0x05
SPAN_ERROR_INV_FS	0x06
SPAN_ERROR_INV_LUN_NFO	0x07
SPAN_ERROR_BAD_HDR_CKSM	0x08
SPAN_ERROR_INV_IDX	0x09
SPAN_ERROR_RD_ONLY	0x0a
SPAN_ERROR_NO_REC_DAT	0x0b
SPAN_ERROR_INV_PART_NFO	0x0c
SPAN_ERROR_SPAN_REQUEST_FAILED	0x0e
SPAN_ERROR_SPAN_REQUEST_RETENTION_TIME	0x0f
SPAN_ERROR_REMOUNT_REFUSED	0x10
ISCSI_ERROR_CONNECT	0x31
ISCSI_ERROR_INV_LUN	0x33
ISCSI_ERROR_LOGIN	0x34
ISCSI_ERR_PWD	0x36
ISCSI_ERR_PROTO	0x37
ISCSI_ERR_TARG_NOT_REACH	0x38
ISCSI_ERR_NO_MEM	0x3a
ISCSI_ERR_SESS_CREATE	0x3b
ISCSI_ERR_INV_PARAMS	0x3c
ISCSI_ERR_SESS_NOT_FOUND	0x3d
ISCSI_ERR_DISCONN	0x3e
ISCSI_ERR_TIMEOUT	0x3f

© Copyright 2001-2021 Bosch Security Systems

ISCSI_ERR_SOCK	0x5f
ISCSI_ERR_TARG_PM	0x6f
ISCSI_SOCK_CLOSED	0x7f
ISCSI_ERR_TCP_CONN_RST	0x8f
ISCSI_ERR_INTR_NOT_SUPP	0x9f
ISCSI_ERR_IP_ZERO	0xa0
ISCSI_ERR_OUT_OF_RES	0xa1
HDD_ERROR_TIMEOUT	0x12
HDD_ERROR_CREATE_FAILED	0x22
HDD_ERROR_ACCESS_DENIED	0x32
HDD_ERROR_DEVICE_PRESENT_TIMEOUT	0x42
HD_PMM_ERROR_LUN_LOCK	0x52
HD_PMM_ERROR_INVALID_ACCESS	0x62
HD_PMM_ERROR_LUN_MGMT_FILE_NOT_FOUND	0x72
HD_PMM_ERROR_LUN_WRITE_PROTECTED	0x82
HD_PMM_ERROR_COMMON	0xf2

Specific errors for 'status' = 'RELEASE' (0x01)

None	0x00	
RELEASE_ERROR_HD_MGR_ERROR	0x01	
RELEASE_ERROR_SPAN_LIST_INCONSISTENCY	0x02	
RELEASE_ERROR_RECORD_HANDLE_DISMISS_ERROR	0x03	
RELEASE_ERROR_BIG_TIME_JUMP	0x04	
RELEASE_ERROR_CLOSED_WHILE_MOUNTING	0x05	
RELEASE_ERROR_INITIAL_MOUNT_ABORT	0x06	
RELEASE_ERROR_INITIAL_MOUNT_UNFINISHED	0x07	
RELEASE_ERROR_NEXT_SPAN_NOT_MOUNTED	0x08	
RELEASE_ERROR_RECORDING_ERROR	0x09	
RELEASE_ERROR_ENCODER_ERROR	0x0a	
RELEASE_ERROR_REPLACE	0x0b	(release was caused by CONF_SPAN_SWITCH command)
RELEASE_ERROR_SWITCH_REQUEST	0x0c	(release was caused by CONF_SPAN_SWITCH command)
RELEASE_ERROR_SPAN_TIME_LIMIT_REACHED	0x0d	(one day limit per span, if max retention configured)
RELEASE_ERROR_RECORD_FORMAT_HARD_LIMIT	0x0e	(initiated span switch by recording internal for record format related issues (gop alignment violation at span end))
RELEASE_ERROR_RECORD_FORMAT_RING_BREAK	0x0f	(initiated span switch by recording internal for record format related issues (ring limit violation at span end))

Specific errors for 'status' = 'MOUNTED' (0x00)

© Copyright 2001-2021 Bosch Security Systems

Standard mount	0x00
HD_MOUNT_CODE_REMOUNTED	0x01

© Copyright 2001-2021 Bosch Security Systems

2.591 CONF_SPAN_USE_STATUS_SECONDARY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a4a	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Span use status of the secondary recording, payload is the same as in command CONF_SPAN_USE_STATUS	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.592 CONF_STAMP

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c71	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Region Info [0] (see description)
...
Region Info [N] (see description)

Region Info

16	32
Stamping Region ID 2 Bytes	**Region Payload Length** 2 Bytes
Region Payload (see description)	
8	24

Stamping Region ID

All regions (global settings)	0
Time Stamp Region	1
Name Stamping Region	2
Alarm Stamping Region	3
Info Stamping Region	4
Spinner Region	5
System Controller Region 1	6
System Controller Region 2	7

© Copyright 2001-2021 Bosch Security Systems

System Controller Region 3	8
Seperate Logo Area	9
Custom stamp area	10

The following table shows the capabilities of each region

Note: Use CONF_ENC_STAMPING_PROPERTIES to query basic options. Read of CONF_STAMP will return all available Tags. Depending on the device different regions are available

Region Payload Length

Number of bytes of all enclosed tags inside this region.

Region Payload

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 1: Visibility

Tag = 0x0001 2 Bytes	**Length = 0x0001** 2 Bytes
Visibility (t_octet) 1 Byte	

Values:

Hidden	0
Visible	1

© Copyright 2001-2021 Bosch Security Systems

Tag 2: Position

Tag = 0x0002 2 Bytes	**Length = 0x0003** 2 Bytes
Position (p_octet) 3 Bytes	

Position

	12	24
Position Type 1 Byte	**Position X** 1 Byte	**Position Y** 1 Byte
6		18

Position Type

OFF	0	
Bottom	1	in this case position X anx Y is ignored
Top	2	in this case position X anx Y is ignored
Custom	3	

Position X

Position X is an integer value from 0 to 255, where 0 is the left border of the image and 255 is the right border of the image.

Position Y

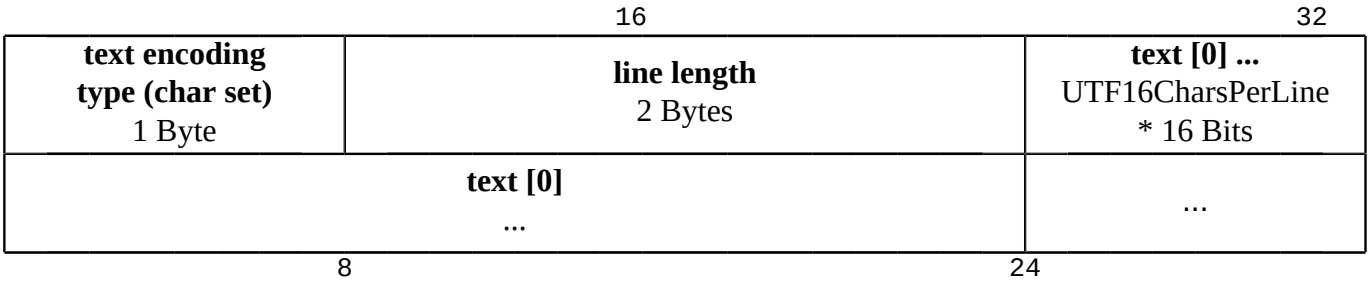
Position Y is an integer value from 0 to 255, where 0 is the top border of the image and 255 is the bottom border of the image.

Tag 3: Text

Tag = 0x0003 2 Bytes	**Length = 0x000** 2 Bytes
Text (p_octet) *Length* Bytes	

© Copyright 2001-2021 Bosch Security Systems

Text



text encoding type (char set)

1: utf16 2 bytes per character

line length

Characters per line, e.g. 32 utf16 symbols. Notice: please use the value 'UTF16CharsPerLine' returned by CONF_ENC_STAMPING_PROPERTIES.

text

Consists of n * UTF16CharsPerLine utf16 chars

Tag 4: Font size

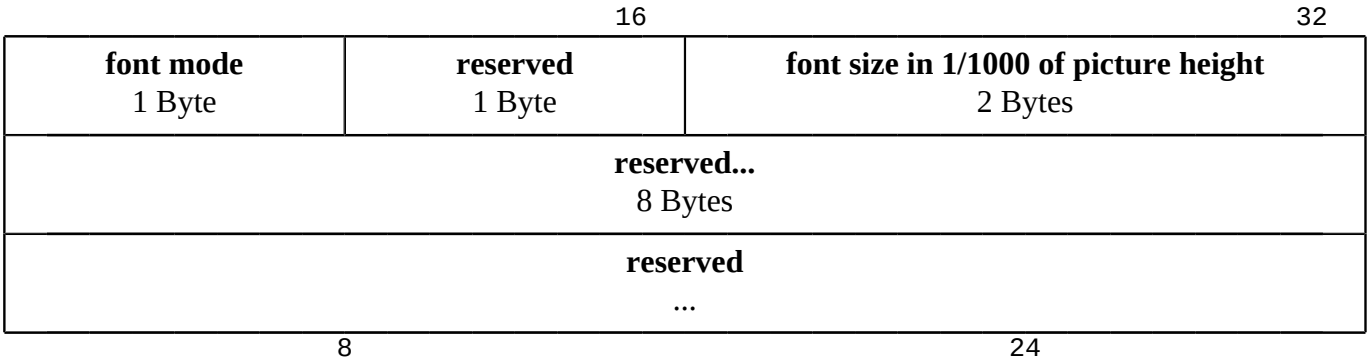
will be read per region and replace preceeding variantof the tag check CONF_ENC_STAMPING_PROPERTIES: customFontSupport:1 -> new variant 0-> old variant

Tag = 0x0004 2 Bytes	**Length = 0x000** 2 Bytes
Font size (p_octet) *Length* Bytes	

Font size

© Copyright 2001-2021 Bosch Security Systems

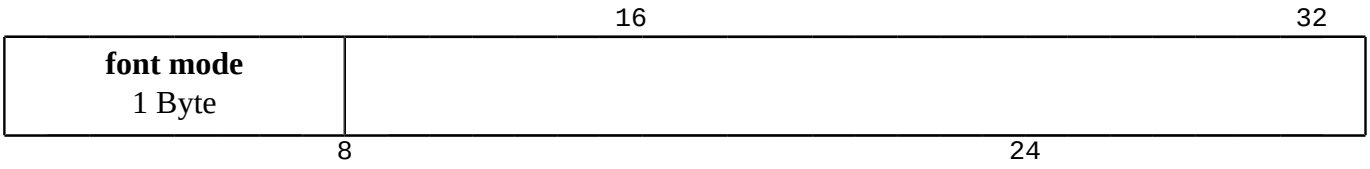
Payload for 'customFontSupport' of command CONF_ENC_STAMPING_PROPERTIES = 'customFontSupported'



font mode

Normal	0	
Big	1	
Custom	255	custom font size

Payload for any other case



font mode

Normal	0
Big	1

Tag 5: Transparent Background

Tag = 0x0005 2 Bytes	**Length = 0x0001** 2 Bytes
Transparent Background (t_octet) 1 Byte	

Values:

No transparent background	0
Transparent background	1

© Copyright 2001-2021 Bosch Security Systems

Tag 6: Color

Tag = 0x0006 2 Bytes	**Length = 0x000c** 2 Bytes
Color (p_octet) 12 Bytes	

Color

Color Mode 1 Byte	**Reserved** 3 Bytes		
Background - Red 1 Byte	**Background - Green** 1 Byte	**Background - Blue** 1 Byte	**Reserved** 1 Byte
Text - Red 1 Byte	**Text - Green** 1 Byte	**Text - Blue** 1 Byte	**Reserved** 1 Byte

16 32

8 24

Color Mode

RGB	0	Red, Green, Blue values are used.

Background - Red

Value 0 - 255

Background - Green

Value 0 - 255

Background - Blue

Value 0 - 255

Text - Red

Value 0 - 255

© Copyright 2001-2021 Bosch Security Systems

Text - Green

Value 0 - 255

Text - Blue

Value 0 - 255

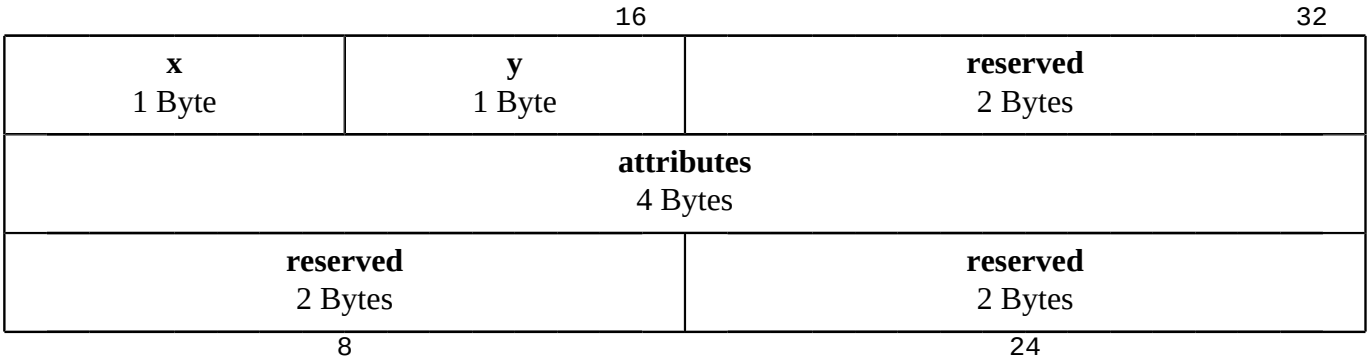
© Copyright 2001-2021 Bosch Security Systems

2.593 CONF_STAMP_ATTR_ALARM

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0938	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



x

X-Position (0...255) where the string should be displayed (0 is left)

y

Y-Position (0...255) where the string should be displayed (0 is up)

attributes

(use CONF_ENC_STAMPING_PROPERTIES to determine supported options)

	Mask	Name	Description
Bit 20	0x00100000	Draw Text Box	Draw a box around the text or transparent text background
Bit 9	0x00000200	cust_color	0=0ff, 1=on (use custom font and background color from payload)

© Copyright 2001-2021 Bosch Security Systems

2.594 CONF_STAMP_ATTR_DEC_FREEZE

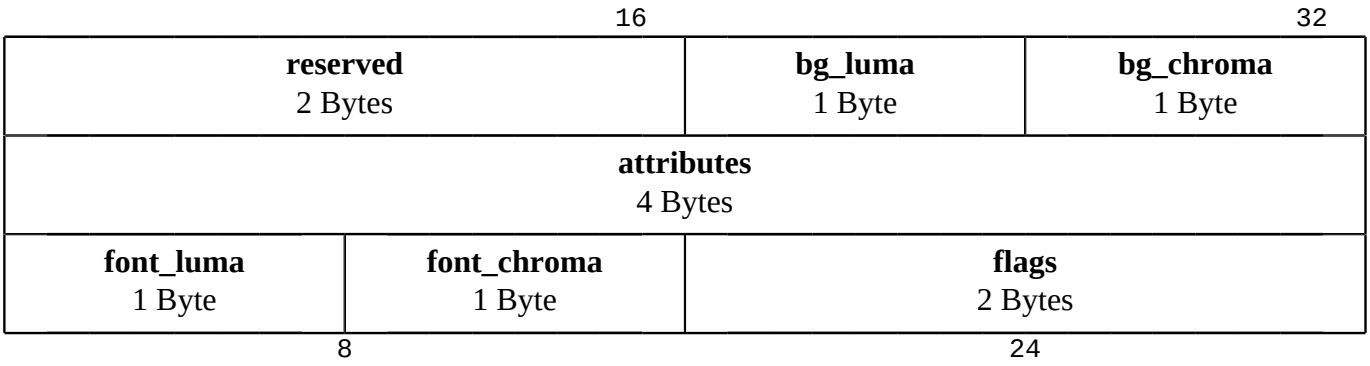
[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0933	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used only on decoders

Payload Structure



bg_luma

Background color: 8 bit luma (Y)

bg_chroma

Background color: 2 x 4 bit chroma (U/V); 'lower 4 bit' = U, 'upper 4 bit' = V

attributes

	Mask	Name	Description
Bit 10	0x00000400	Background Box	Big text background box
Bit 8	0x00000100	Custom Font	Use custom font and background color given in bg_luma, bg_chroma, font_luma and font_chroma

font_luma

Font color: 8 bit luma (Y)

© Copyright 2001-2021 Bosch Security Systems

font_chroma

Font color: 2 x 4 bit chroma (U/V); 'lower 4 bit' = U, 'upper 4 bit' = V

flags

Flags not in use for freeze string, set to all-zero

© Copyright 2001-2021 Bosch Security Systems

2.595 CONF_STAMP_ATTR_INFO

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc1	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	p_octet	service	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.596 CONF_STAMP_ATTR_LOGO

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c0f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	p_octet	service	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.597 CONF_STAMP_ATTR_NAME

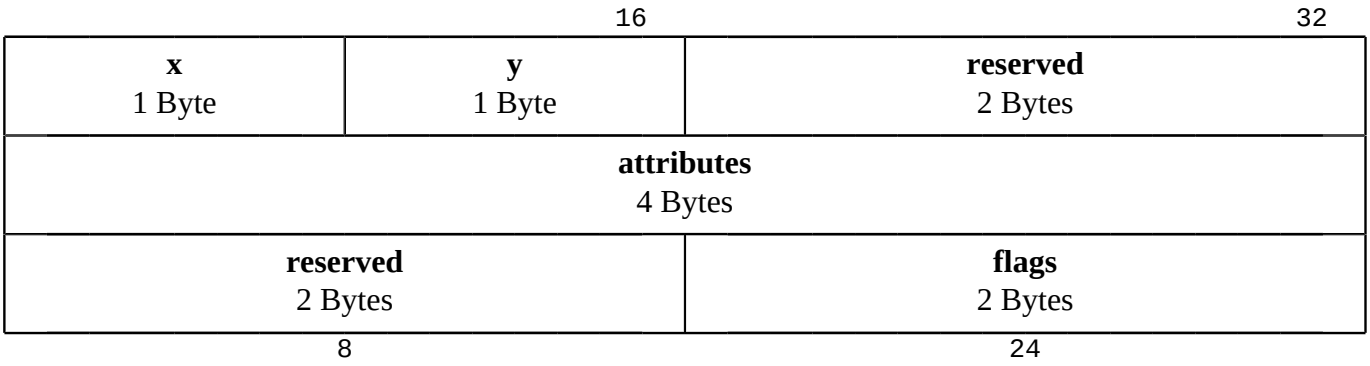
[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0936	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Supported for CPP4, CPP6, CPP7 and CPP7.3. For other platforms use CONF_STAMP instead.

Payload Structure



x

X-Position (0...255) where the string should be displayed (0 is left)

y

Y-Position (0...255) where the string should be displayed (0 is up)

attributes

(use CONF_ENC_STAMPING_PROPERTIES to determine supported options)

	Mask	Name	Description
Bit 21	0x00200000	Draw Banner	Draw a banner from top to end of name text if y pos <= 128 from bottom if ypos > 128)
Bit 20	0x00100000	Draw Text Box	Draw a box around the text or transparent text background
Bit 9	0x00000200	cust_color	0=0ff, 1=on (use custom font and background color from payload)

© Copyright 2001-2021 Bosch Security Systems

flags

	Mask	Name
Bit 3	0x0008	Display only logo
Bit 2	0x0004	Logo position relative to font
Bit 1	0x0002	Enable logo stamping

© Copyright 2001-2021 Bosch Security Systems

2.598 CONF_STAMP_ATTR_TIME

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0937	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	p_octet	service	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16		32
x 1 Byte	**y** 1 Byte	**reserved** 2 Bytes
attributes 4 Bytes		
reserved 2 Bytes		**reserved** 2 Bytes
8		24

x

X-Position (0...255) where the string should be displayed (0 is left)

y

Y-Position (0...255) where the string should be displayed (0 is up)

attributes

(use CONF_ENC_STAMPING_PROPERTIES to determine supported options)

	Mask	Name	Description
Bit 20	0x00100000	Draw Text Box	Draw a box around the text or transparent text background
Bit 9	0x00000200	cust_color	0=0ff, 1=on (use custom font and background color from payload)

© Copyright 2001-2021 Bosch Security Systems

2.599 CONF_STAMP_DEC_FREEZE_STRING

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092f	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	String displayed on the local monitor if DEC_SHOW_FREEZE is enabled and no vido data is coming in	
Write	p_string	service	String displayed on the local monitor if DEC_SHOW_FREEZE is enabled and no vido data is coming in	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.600 CONF_START_CLIENT

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b20	account info (1...MAX_ACCOUNT	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	service	Starts a ftp client session, might be needed for commands like CONF_FTP_CDUP, CONF_FTP_CWD, CONF_FTP_LIST, CONF_FTP_PWD. This session stays alive for 5 min. It can be should be shut down using CONF_FTP_STOP_CLIENT. This command returns a 2 byte sessionId that can be used to reference the session	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.601 CONF_START_MULTICAST_STREAMING

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b8b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Video streaming on coder (1st coder-> 1st octet, 2nd coder-> 2nd octet....)	
Write	p_octet	service	Enable video streaming on coder. 1st Coder->1 octet, 2nd Coder ->2nd octet....	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.602 CONF_START_RECORD

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0acc	1 - primary recording, 2 - secondary recording	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Start recording on choosen cams, payload needs a variable count pairs of 32 bit fiels for cam state (posibble states: 1 : on, 0 : off) and 32 bit fiels for cam selection mask, this mask will layed over the state bit field and can be used to choose the cams specific cams, other cams will be ignored, each bit field pair represents 32 cams: first pair - cam 1 to cam 32, second pair -cam 33 to cam 64 and so on	
Write	p_octet	service	Start recording on choosen cams, payload needs a variable count pairs of 32 bit fields for cam state (posibble states: 1 : on, 0 : off) and 32 bit fiels for cam selection mask, this mask will layed over the state bit field and can be used to choose the cams specific cams, other cams will be ignored, each bit field pair represents 32 cams: first pair - cam 1 to cam 32, second pair -cam 33 to cam 64 and so on	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.603 CONF_START_SPAN_RECORD

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09f7	1 - primary recording, 2 - secondary recording	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword p_octet	minimal	Obsolete, please use CONF_START_RECORD instead	
Write	t_dword p_octet	service	Obsolete, please use CONF_START_RECORD instead	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.604 CONF_START_VIPROC_CONFIG_EDITING

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a38	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	iva	Moves to the dome to the scene corresponding to the specified config and freezes the dome there. If already a config is in configuration mode, this command returns with an error. All subsequent viproc commands are related to this configuration. This command works also for non-dome devices. There it won't try to move the dome to the corresponding scene.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.605 CONF_STARTPAGE_BACKGROUND_URL

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x028d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the URL of the startpage' background image (max 64 char)	
Write	p_string	service	Set the URL of the startpage' background image (max 64 char)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.606 CONF_STARTPAGE_LOGO_URL

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x028e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the URL of the startpage' OEM logo image (max 64 char)	
Write	p_string	service	Set the URL of the startpage' OEM logo image (max 64 char)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.607
CONF_STARTPAGE_PRESENTATION_SWITCHES

[API: browser]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x028f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get switches for HTML startpage presentation (usage is up the the user)	
Write	t_dword	service	Set switches for HTML startpage presentation (usage is up the the user)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.608 CONF_STATELESS_IP_V6_PREFIX_LEN

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get unit's IPv6 address prefix length (Automatic assigned IP)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.609 CONF_STATELESS_IP_V6_STR

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	IPv6 String (Automatic assigned IP)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.610 CONF_STATIC_PRIV_MSK_OVERLAY

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd3	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

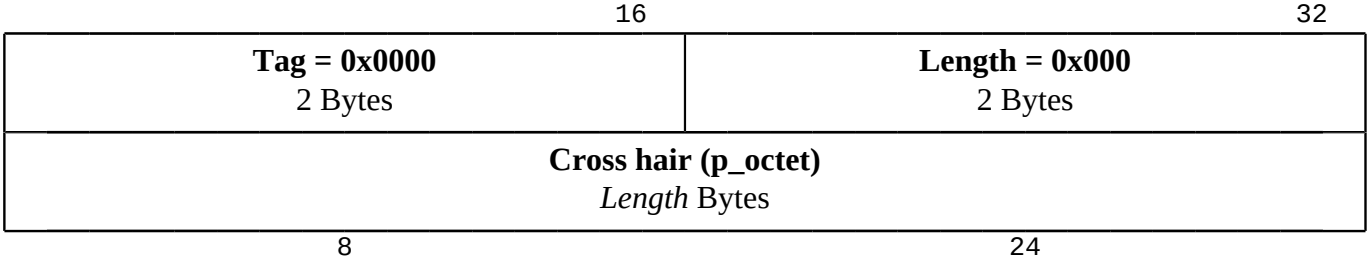
Length

Length of tagged value without length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 0: Cross hair



© Copyright 2001-2021 Bosch Security Systems

Cross hair

		16		32
Enable 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte
Reserved 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte
	8		24	

Enable

OFF	0
On	1

© Copyright 2001-2021 Bosch Security Systems

2.611
CONF_STATIC_PRIV_MSK_OVERLAY_OPTIONS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cd9	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] 0: cross hair not supported / 1: basic support of cross hair ; Byte[1-15] reserved;	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.612 CONF_STATUS_AUTO_TRACKER

[API: autotracker]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b86	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets current mode of auto tracker. 0=off; 1=idle; 2=seeking; 3=tracking active; 4=tracking passive	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.613 CONF_STD_MEDIA_CONNECTION_DIRECTION

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x030c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

outgoing media for alarm connections	0
req. incoming media for alarm connections	1
req. bidirectional media for alarm connections	2

© Copyright 2001-2021 Bosch Security Systems

2.614
CONF_STD_MEDIA_ENCAPSULATION_PROTOKOL

[API: alarm connections]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0309	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Defines the standard media encapsulation protokol: 1=RTP over UDP, 2=TCP	
Write	t_octet	service	Defines the standard media encapsulation protokol: 1=RTP over UDP, 2=TCP	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.615 CONF_STEALTH_MODE

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d2f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read stealth mode setting	
Write	t_dword	service	Set to stealth mode: force login also for 'always_legacy' commands + device does not reply to network scans anymore.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

disabled	0
enabled	1

© Copyright 2001-2021 Bosch Security Systems

2.616 CONF_STOP_CLIENT

[API: account]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b21	SessionId (see CONF_START_CLIEN	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_word	service	Shut down the ftp session indified via the num parameter, might be needed for commands like CONF_FTP_CDUP, CONF_FTP_CWD, CONF_FTP_LIST, CONF_FTP_PWD	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.617 CONF_STOP_VIPROC_CONFIG_EDITING

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a39	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	iva	Stops editing of the specified config (see CONF_START_VIPROC_CONFIG_EDITING). If the specified scene does not match the currently edited config, a dword with value 0 is returned.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.618 CONF_STORAGE_IO

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a61	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List the currently running storage io tasks.	
Write	p_octet	service	Start or stop a storage io task.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure

16	32
Lun Address... (see description)	
Lun Address ...	
Start Burst Size 4 Bytes	
End Burst Size 4 Bytes	
Burst Size Duration 4 Bytes	
Burst Size Increase 4 Bytes	
Maximum Read Rate 4 Bytes	
Maximum Write Rate 4 Bytes	
Repeat 4 Bytes	
Mode 2 Bytes	**Flags** 2 Bytes
ID 4 Bytes	
Action 1 Byte	**Reserved...** 47 Bytes

© Copyright 2001-2021 Bosch Security Systems

Reserved [...]
Reserved ...
Reserved ...

8 24

Lun Address

16 32

iSCSI IP 4 Bytes
Target Idx 4 Bytes
Lun 4 Bytes
Span Idx 4 Bytes

8 24

iSCSI IP

The ip address of the iscsi server

Target Idx

The index of the target on the iscsi server.

Lun

The logical unit.

Span Idx

The index of the span. Only used, if the flag RESTRICT TO SPAN is set.

Start Burst Size

The size in LBAs of the burst the test starts with. This value must be lower or equal to the end burst size.

End Burst Size

The size in LBAs of the burst the test ends with. This value may be clipped to the maximum burst value of the storage.

© Copyright 2001-2021 Bosch Security Systems

Burst Size Duration

The number of milliseconds each bust size is read and written.

Burst Size Increase

The number of LBAs the burst size is increased after each burst-duration.

Maximum Read Rate

The maximum datarate that is read in KBit/sec.

Maximum Write Rate

The maximum datarate that is written in KBit/sec.

Repeat

The number of times the test is repeated which means how often the specified burst sequence is run.

Mode

The test may run in the following modes. The modes RECORDING SIMULATION, LATENCY and REPLAY SIMULATION use different fields in the request packet which are not documented yet.

LINEAR	0x0000
RANDOM	0x0001
RECORDING SIMULATION	0x0002
LATENCY	0x0003
REPLAY SIMULATION	0x0004

If the mode is set to LINEAR then the test reads/writes from/to consecutive lba addresses. If the maximum lba is reached and the STORAGE_IO_FLAG_MAX_LBA_END flag is set, the test stops. Otherwise the test wraps its lba pointer to the beginning of its storage.

If the mode is set to RANDOM then the test reads/writes from/to randomly chosen lba addresses of the storage.

Flags

	Mask	Name	Description
Bit 9	0x0200	MAX LBA END	If set, the tests stops when the write pointer reaches the maximum lba address of the lun/ span (only in sequential access mode).
Bit 8	0x0100	ISCSI SESSION EXCLUSIVE	If set, the tests use an exclusive iscsi session which is not shared among other tests.
Bit 7	0x0080	RESTRICT TO SPAN	If set, the tests only uses the span of the lun, which is specified in the 'Span Idx' field of the lun-address.
Bit 6	0x0040	ERROR STOP	If set, the tests stops when an io error occurrs.

© Copyright 2001-2021 Bosch Security Systems

Bit 5	0x0020	EXP BURST INCREASE	If set, the field 'Burst Size Increase' is ignored. Instead the burst size is doubled after each run.
Bit 4	0x0010	VERIFY WRITE	If set, every lba that was written is read afterwards to verify that the data was correctly written.
Bit 3	0x0008	KEEP DATA	If set, all lbas that are written are read before so that no recording data should be destroyed. Be careful that no one else writes to this lun simultaneously. In the latter case data on the storage could be lost.
Bit 1	0x0002	WRITE ONLY	If set, only write operations are performed. May not be used simultaneously with the READ ONLY flag.
Bit 0	0x0001	READ ONLY	If set, only read operations are performed. May not be used simultaneously with the WRITE ONLY flag.

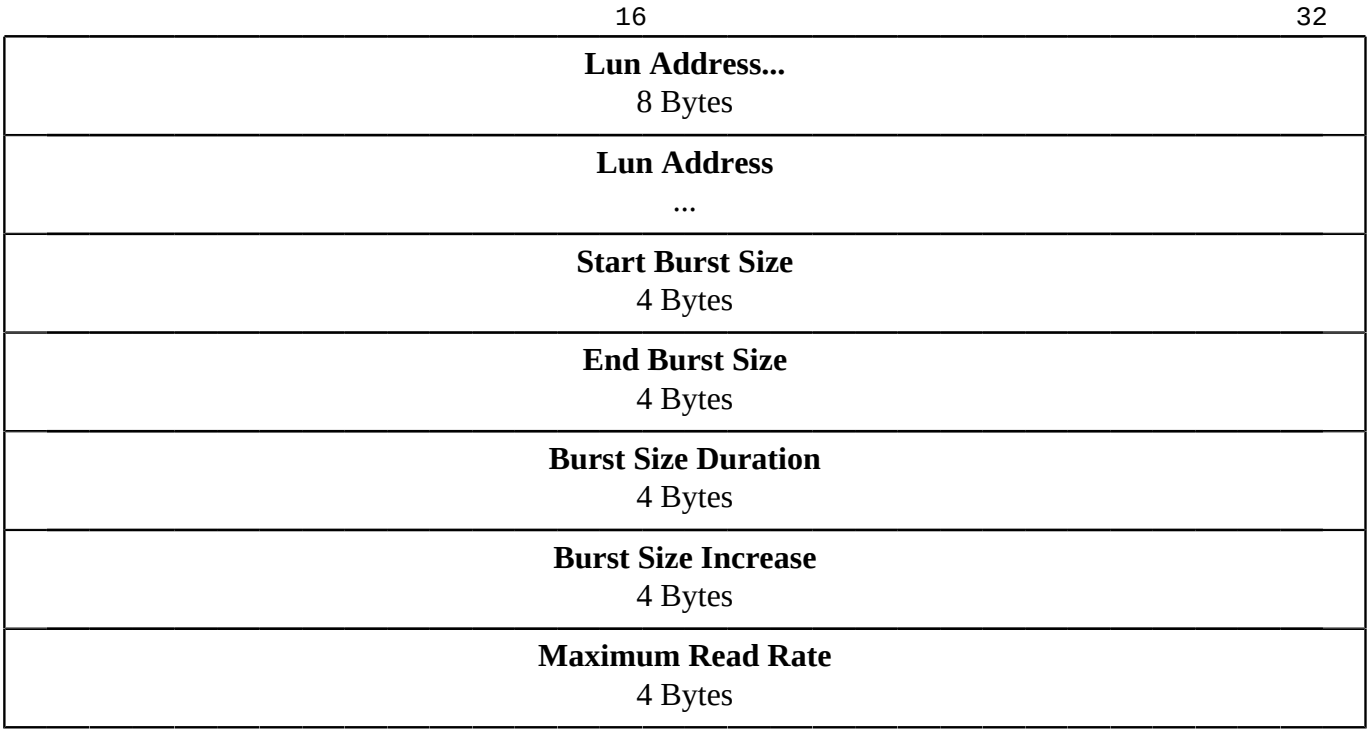
ID

Identifier of the storage io. Set to zero when starting a new storage io. Set to the from the start request returned value for stopping a storage io.

Action

STOP	0x00
START	0x01

Response Payload Structure

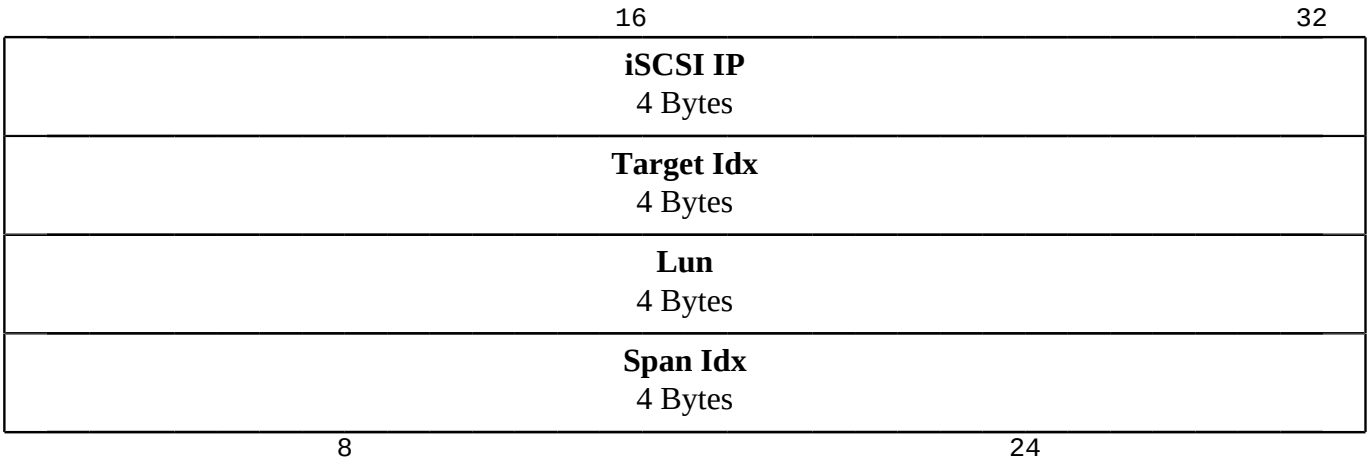


© Copyright 2001-2021 Bosch Security Systems

Maximum Write Rate 4 Bytes		
Repeat 4 Bytes		
Mode 2 Bytes		**Flags** 2 Bytes
ID 4 Bytes		
Status 1 Byte	**Error** 1 Byte	**Reserved** 2 Bytes
Repeat Count 4 Bytes		
Current Burst Size 4 Bytes		
Current Offset 4 Bytes		
Current Read Datarate 4 Bytes		
Current Write Datarate 4 Bytes		
Read Bytes 4 Bytes		
Write Bytes 4 Bytes		
Read Operations 4 Bytes		
Write Operations 4 Bytes		
Read Errors 4 Bytes		
Write Errors 4 Bytes		
8		24

© Copyright 2001-2021 Bosch Security Systems

Lun Address



iSCSI IP

The ip address of the iscsi server

Target Idx

The index of the target on the iscsi server.

Lun

The logical unit.

Span Idx

The index of the span. Only used, if the flag RESTRICT TO SPAN is set.

Start Burst Size

The size in LBAs of the burst the test starts with. This value must be lower or equal to the end burst size.

End Burst Size

The size in LBAs of the burst the test ends with. This value may be clipped to the maximum burst value of the storage.

Burst Size Duration

The number of milliseconds each bust size is read and written.

Burst Size Increase

The number of LBAs the burst size is increased after each burst-duration.

© Copyright 2001-2021 Bosch Security Systems

Maximum Read Rate

The maximum datarate that is read in KBit/sec.

Maximum Write Rate

The maximum datarate that is written in KBit/sec.

Repeat

The number of times the test is repeated which means how often the specified burst sequence is run.

Mode

The test may run in the following modes. The modes RECORDING SIMULATION, LATENCY and REPLAY SIMULATION use different fields in the request packet which are not documented yet.

LINEAR	0x0000
RANDOM	0x0001
RECORDING SIMULATION	0x0002
LATENCY	0x0003
REPLAY SIMULATION	0x0004

If the mode is set to LINEAR then the test reads/writes from/to consecutive lba addresses. If the maximum lba is reached and the STORAGE_IO_FLAG_MAX_LBA_END flag is set, the test stops. Otherwise the test wraps its lba pointer to the beginning of its storage.

If the mode is set to RANDOM then the test reads/writes from/to randomly chosen lba addresses of the storage.

Flags

	Mask	Name	Description
Bit 9	0x0200	MAX LBA END	If set, the tests stops when the write pointer reaches the maximum lba address of the lun/ span (only in sequential access mode).
Bit 8	0x0100	ISCSI SESSION EXCLUSIVE	If set, the tests use an exclusive iscsi session which is not shared among other tests.
Bit 7	0x0080	RESTRICT TO SPAN	If set, the tests only uses the span of the lun, which is specified in the 'Span Idx' field of the lun-address.
Bit 6	0x0040	ERROR STOP	If set, the tests stops when an io error occurrs.
Bit 5	0x0020	EXP BURST INCREASE	If set, the field 'Burst Size Increase' is ignored. Instead the burst size is doubled after each run.
Bit 4	0x0010	VERIFY WRITE	If set, every lba that was written is read afterwards to verify that the data was correctly written.
Bit 3	0x0008	KEEP DATA	If set, all lbas that are written are read before so that no recording data should be destroyed. Be careful that no one else writes to this lun

© Copyright 2001-2021 Bosch Security Systems

			simultaneously. In the latter case data on the storage could be lost.
Bit 1	0x0002	WRITE ONLY	If set, only write operations are performed. May not be used simultaneously with the READ ONLY flag.
Bit 0	0x0001	READ ONLY	If set, only read operations are performed. May not be used simultaneously with the WRITE ONLY flag.

ID

Identifier of the storage io. This value must be specified on actions other than start.

Status

STOPPING	0x00
STARTING	0x01
RUNNING	0x02
DONE	0x03
ERROR	0x04

Error

INTERNAL	0x01
LUN	0x02
INVALID PARAMS	0x03
READ	0x04
WRITE	0x05
WRITE VALIDATE	0x06

Repeat Count

The number of turns the test repeated so far.

Current Burst Size

The number of LBAs of the current burst.

Current Offset

The current position of the i/o pointer in percentage of the whole storage. This value is not significant in random access mode.

Current Read Datarate

The current read datarate in KBit/s.

© Copyright 2001-2021 Bosch Security Systems

Current Write Datarate

The current write datarate in KBit/s.

Read Bytes

The number of bytes that were read since the test was started.

Write Bytes

The number of bytes that were written since the test was started.

Read Operations

The number of read operations successfully performed since the test was started.

Write Operations

The number of write operations successfully performed since the test was started.

Read Errors

The number of read errors since the test was started.

Write Errors

The number of read errors since the test was started.

© Copyright 2001-2021 Bosch Security Systems

2.619 CONF_STORAGE_LIST

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a37	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the device storage list.	
Write	p_octet	service	Write the stuff. (, not supported while recording is running)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

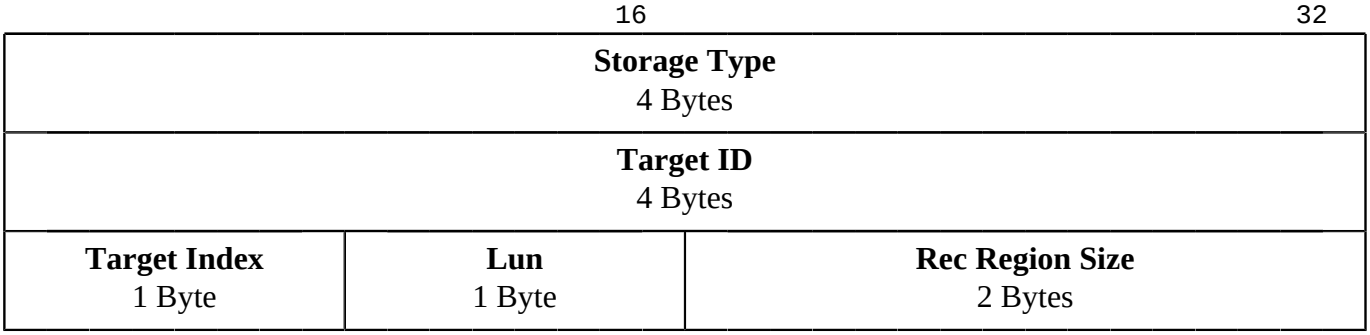
Use this command to configure and manage local or remote storage. The storage list may contain up to 16 entries. If a remote (iscsi) storage is to be managed, the lun address has to be specified. Realize, that if you export an remote storage, the device will have preformance drawbacks cause the ip- and iscsi-stack will be passed through twice. Prefer to access the remote storage directly instead.

The storage lun can be write protected. If this state was recognized, the Mgr Status will inform about this state by the value "WRITE PROTECTION". There are three ways for a storage lun to become write protected here. First to configure explicite the storage lun(Span Mgr) to be in read only mode, second by a defect storage lun, which will cause a fall back into a write protection mode, and third if the storage is set physically to write protection (e.g. user moves the lever on the sd card into the write protection position).

The storage devices will be exported through iscsi in the order of this list (storage devices with the 'iSCSI export' field set to 'For local use only' [0x00] are omitted) and the storage will appear in the same place of the target name list of an iscsi discovery on that device.

This command will be send as msg (can also be send as snmp trap with same pyload) in case of crossing the storage threshold with Mgr status ALARM THRESHOLD or in case of overwriting recording, that is still protected by unexpired retention time with mgr status ALARM OVERWRITE.

Write Payload Structure



© Copyright 2001-2021 Bosch Security Systems

iSCSI export 1 Byte	**Span Manager** 1 Byte	**Reserved...** 4 Bytes	
Reserved ...		**Flags** 1 Byte	**Storage threshold** 1 Byte
8		24	

Storage Type

The type of the storage.

FILE	0x02
RAM	0x03
ISCSI	0x04
USB	0x05
IDE	0x08
CF	0x09
SD	0x0b
SMB	0x11
SD OVER USB	0x16

Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES). Will be ignored, if the storage type is not iscsi.

Target Index

The target index on the iscsi server. Will be ignored, if the storage type is not iscsi.

Lun

The iscsi lun. Will be ignored, if the storage type is not iscsi.

Rec Region Size

Size in GB from the Storage used for recording. Default is 0 which means as much recording space as the storage size allowes. The same will happen, if this Size is to big for the storage.

iSCSI export

For local use only	0x00
Make storage available through iSCSI	0x01

Span Manager

Span manager off	0x00
Span manager on	0x01
Span format lun and start span manager	0x02

© Copyright 2001-2021 Bosch Security Systems

Migrate lun and start span manager	0x03
read only mode	0x04
erase lun and format	0x05

Flags

	Mask	Name	Description
Bit 0	0x01	default setting	just info, that this is a default setting

Storage threshold

This is the threshold of free storage in percent and means that at least this amount of memory should be free. A Message of this command will be send always, this treshold is crossed in eiher direction.

Read Payload Structure

16 32

Storage Type 4 Bytes			
Target ID 4 Bytes			
Target Index 1 Byte	**Lun** 1 Byte	**Fmt Progress** 1 Byte	**Utilization** 1 Byte
iSCSI export 1 Byte	**Span Manager** 1 Byte	**Mgr Status** 1 Byte	**Mgr Error** 1 Byte
Mgr Flags 1 Byte	**Reserved** 1 Byte	**Flags** 1 Byte	**Storage threshold** 1 Byte

8 24

Storage Type

The type of the storage.

FILE	0x02
RAM	0x03
ISCSI	0x04
USB	0x05
IDE	0x08
CF	0x09
SD	0x0b
SMB	0x11
SD OVER USB	0x16

© Copyright 2001-2021 Bosch Security Systems

Target ID

The target id of the lun (may be the ipv4 address for older versions or with default target resolve configuration, see CONF_TARGET_ID_RESOLVE_RULES). Will be ignored, if the storage type is not iscsi.

Target Index

The target index on the iscsi server. Will be ignored, if the storage type is not iscsi.

Lun

The iscsi lun. Will be ignored, if the storage type is not iscsi.

Fmt Progress

If the storage is in state formatting, the field indicates the formatting progress in percentage.

Utilization

If the storage is span formatted, this field indicates the amount capacity used (in percentage).

iSCSI export

For local use only	0x00
Make storage available through iSCSI	0x01

Span Manager

Span manager off	0x00
Span manager on	0x01
Span format lun and start span manager	0x02
Migrate lun and start span manager	0x03
read only mode	0x04
erase lun and format	0x05

Mgr Status

The status of the span manager in the response packet. Set to zero in the request packet.

OFF	0x00
ON	0x01
FORMATING	0x02
MIGRATING	0x03
ALARM THRESHOLD (msg only) / OFFLINE (read payload only)	0x04
ALARM OVERWRITE (msg only) / ONLINE (read payload only)	0x05

© Copyright 2001-2021 Bosch Security Systems

STOPPING	0x06
ERROR	0x07
ONLINE (WRITE PROTECTION)	0x08

Mgr Error

If the span manager status is ERROR, this field contains details of the error that occurred. See command CONF_SPAN_USE_STATUS for possible error codes.

Mgr Flags

	Mask	Name	Description
Bit 0	0x01	lun scan running	scan job for all spans on the lun is running

Flags

	Mask	Name	Description
Bit 0	0x01	default setting	just info, that this is a default setting

Storage threshold

This is the threshold of free storage in percent and means that at least this amount of memory should be free. A Message of this command will be send always, this treshold is crossed in eiher direction.

Note: If the storage list is empty or only contains entries with the 'iSCSI export' field set to 'For local use only' (0x00), the iscsi server is stopped, if it is running, cause no targets and luns would be provided. If the list contains at least one entry with the 'iSCSI export' field set to 'Make storage available through iSCSI' (0x01), the iscsi server is started, if not already running, to provide the iscsi service. The iscsi server can be controlled through the command CONF_ISCSI_SERVER_STATE too.

© Copyright 2001-2021 Bosch Security Systems

2.620 CONF_STORAGE_LOAD

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bba	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Load of local storage in percent	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.621 CONF_STORAGE_MEDIUM_AVAIL

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	List of dwords with available storage medium types (None = 0, USB = 5, IDE = 8, CF = 9, IMG File = 10, SD = 11, SPAN FILES = 12, SD NOT INSERTED = 14, CF NOT INSERTED = 15, IMG FILE NOT INSERTED = 16, SMB SPAN FILES = 17, SECOND SD = 18, SECOND SD NOT INSERTED = 19, SD OVER USB = 22, SD OVER USB NOT INSERTED = 23, SPAN FILES NOT INSERTED = 24)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.622 CONF_STORAGE_MEDIUM_TYPE

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09d3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Obsolete	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.623 CONF_STORAGE_REPORT

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09cf	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get information about the storage device	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Request Payload Structure

To get information about a storage device, a read packet with no payload must be send to the rcp server.

Response Payload Structure

16			32
Type 1 Byte	**Availability** 1 Byte	**Access Type** 1 Byte	**IO Error Type** 1 Byte
IO Error Count 4 Bytes			
Throughput Read 4 Bytes			
Throughput Write 4 Bytes			
iSCSI Login Connection [0] (see description)			
...			
iSCSI Login Connection [N] (see description)			
8		24	

Type

None	0x00
RAM Recording	0x03
iSCSI	0x04
USB	0x05

© Copyright 2001-2021 Bosch Security Systems

IDE	0x08
Compact Flash	0x09
SMB	0x0B

Availability

Unknown	0x00
No	0x01
Yes	0x02
locked by another device	0x03

Access Type

Unknown	0x00
Read Only	0x01
Read Write	0x02

IO Error Type

None	0x00
Read	0x01
Write	0x02

IO Error Count

Number of arised I/O errors.

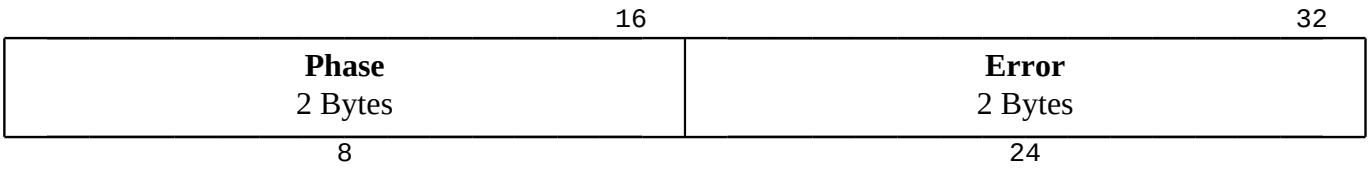
Throughput Read

Kilobytes read from device.

Throughput Write

Kilobytes written to device.

iSCSI Login Connection



Phase

Indicates the phase, the iSCSI client has reached on login.

© Copyright 2001-2021 Bosch Security Systems

Error

Indicates the error that happend in that state.

© Copyright 2001-2021 Bosch Security Systems

2.624 CONF_STORAGE_REPORT_SECONDARY

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a50	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get information about the storage device of secondary recording, payload same as in command CONF_STORAGE_REPORT	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.625 CONF_STORAGE_TARGET_ID

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c09	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the storage target id, if there is no explicite configured (default 0), it returns the device unit ipv4 address	
Write	t_dword	service	Set the storage target id of this device which is an system wide unique identifier, which can be used to resolve it to connect to the local iscsi exported storage (0 means default)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.626 CONF_STRATOCAST_ONOFF

[API: stratocast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cab	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Switch on or off the automatic start of stratocast service	
Write	f_flag	minimal	Switch on (true) or off (false) the automatic start of stratocast service	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.627 CONF_STRATOCAST_REGISTER

[API: stratocast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cad	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	Register at stratocast service and switch on automatic reconnect. As payload the activation code must be provided	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.628 CONF_STRATOCAST_STATE

[API: stratocast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the actual state of the stratocast connection (0=inactive, 1=connecting, 2=registering, 3=connected)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.629 CONF_STREAM_EXCLUSIVE_CHECK

[API: roi]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba0	no - addressing via Session ID	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Check if connected ROI stream session is exclusive (meaning only one instance to one client -> exclusive ROI steering).	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

non-exclusive	0
exclusive	1

© Copyright 2001-2021 Bosch Security Systems

2.630 CONF_STREAM_PRIORITY

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb0	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the h26x stream (1..n) which should have high priority. Set to 0 if all encoder should run free	
Write	t_dword	service	Select the h26x streams (1..n) which should have high priority, by setting the enc inst bit (1<<inst). Set to 0 if all encoder should run free. Currently only one instance is supported	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.631 CONF_STREAM_SECURITY_OPTIONS

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads rtp stream security settings options (2 bytes maxHashId, 2 bytes maxSigId, 4 bytes minSigDist, 4 bytes maxSigDist, 4 bytes minDistMid, 4 bytes minDistLow)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.632 CONF_STREAM_SECURITY_V2

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bb8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Reads rtp stream security settings (2 bytes hash id[0=off, 1=classic watermark, 2=MD5, 3=SHA1, 4=SHA256], 2 bytes signature id[0=Off, 1=RSAwithSHA1], 4 bytes signature interval)	
Write	p_octet	service	Configures rtp stream security settings (2 bytes hash id[0=off, 1=classic watermark, 2=MD5, 3=SHA1, 4=SHA256], 2 bytes signature id[0=Off, 1=RSAwithSHA1], 4 bytes signature interval, setting signature interval to 0xFFFFFFFF selects default)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.633
CONF_STREAMING_GATEWAY_ACTIVE_LINES

[API: streaming gateway]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b25	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the number of configured streaming gateway lines	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.634 CONF_STREAMING_GATEWAY_CONFIG

[API: streaming gateway]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b24	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read streaming gateway configuration, see detailed description	
Write	p_octet	service	Write streaming gateway configuration, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

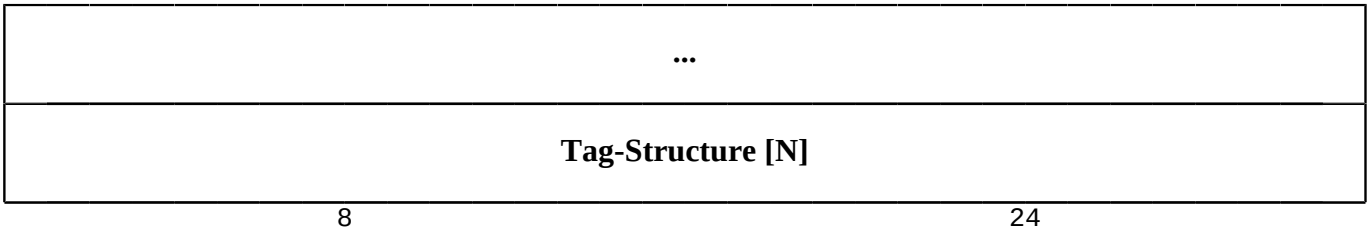
Description

This command is used to read and write the configuration of the streaming gateway. For the read direction only "gateway line" and "gateway stream" parameter are relevant for the send diretion. The reply payload will contain the full parameter set. For write direction use the full parameter set to configure a streaming gate camera, if the action is "add camera". Depending on the camera type and connection type, additional optional parameters are nessessary. E.g. the device simulator is able to connect to another bosch camera or device specified by the url ([ip]:[port]/[line (1,2,...)]/[coder idx (1,2,...)]). E.g. with url "10.1.10.20:80/1/1" it connects to the device on ip 10.1.10.20 and port 80 on the first line and the first video encoder.

Payload Structure

16			32
gateway line 2 Bytes		**gateway stream** 1 Byte	**action** 1 Byte
protocol 1 Byte	**version** 1 Byte	**connection** 1 Byte	**Reserved** 1 Byte
manufacturer id 4 Bytes			
device type... 32 Bytes			
device type [...]			
device type ...			
Tag-Structure [0]			

© Copyright 2001-2021 Bosch Security Systems



gateway line

Local line of streaming gateway id

gateway stream

Local stream of streaming gateway starting with 0 for the first stream

action

Action for write direction, choose between add camera and remove camera, in case of remove camera, except this field, 'gateway line', 'gateway stream' and 'version' all other fields in the payload are irrelevant

add camera	0
remove camera	1

protocol

Choose protocol type

none	0
onvif	1
jpeg	2
bosch	3
rtsp	4
rtsp h263	5
rtsp h264	6
rtsp jpeg	7

version

Version of the command, current version is 1

connection

Connection type

udp unicast	0
udp multicast	1
tcp	2

manufacturer id

Distinguish between bosch cameras and foreign cameras

© Copyright 2001-2021 Bosch Security Systems

unknown	0
bosch	1

device type

32 ascii characters including zero termination

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

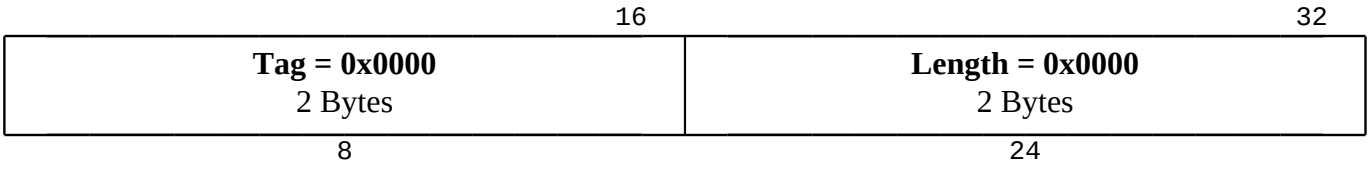
Length

Length of tagged value without length and tag field

Tag

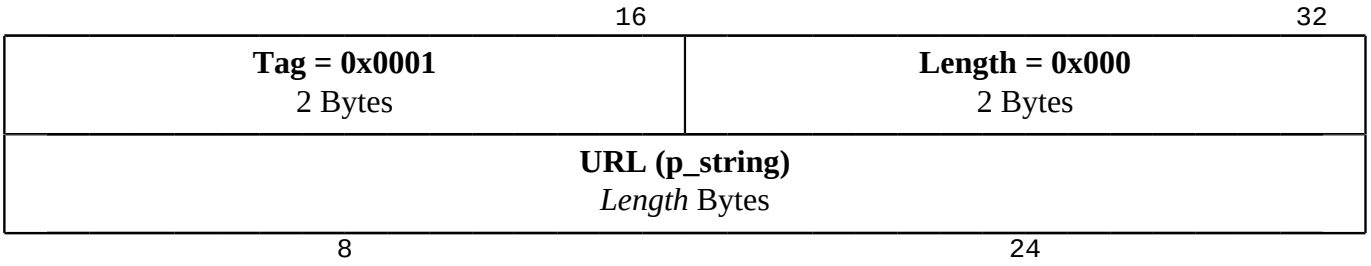
Tag specifying the encoding and meaning of the value

Tag 0: None



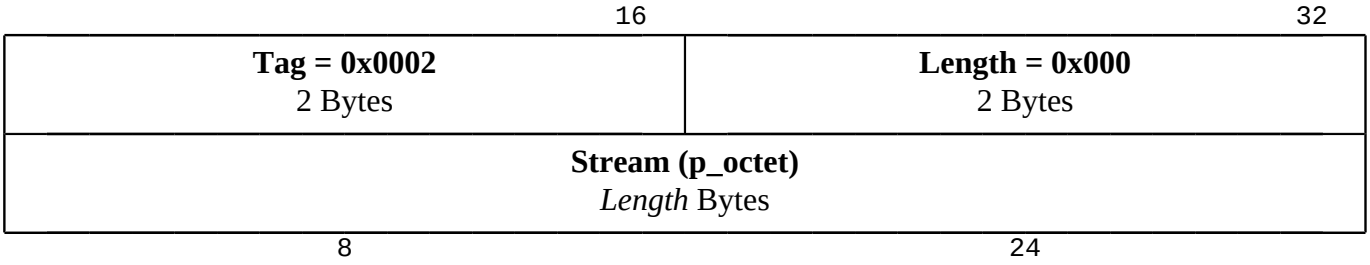
Tag 1: URL

zero terminated ascii string containing the url of the remote camera. Length of the string including zero termination is limited by the 'length' of the optional parameter

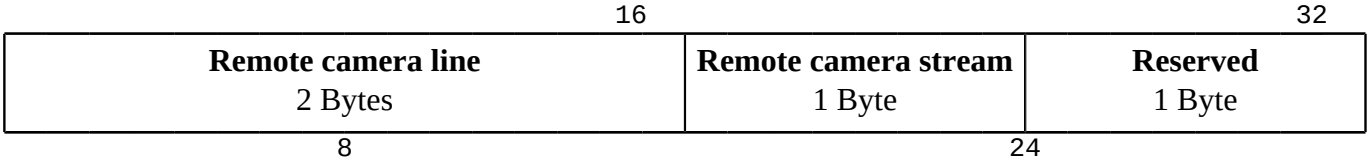


© Copyright 2001-2021 Bosch Security Systems

Tag 2: Stream



Stream

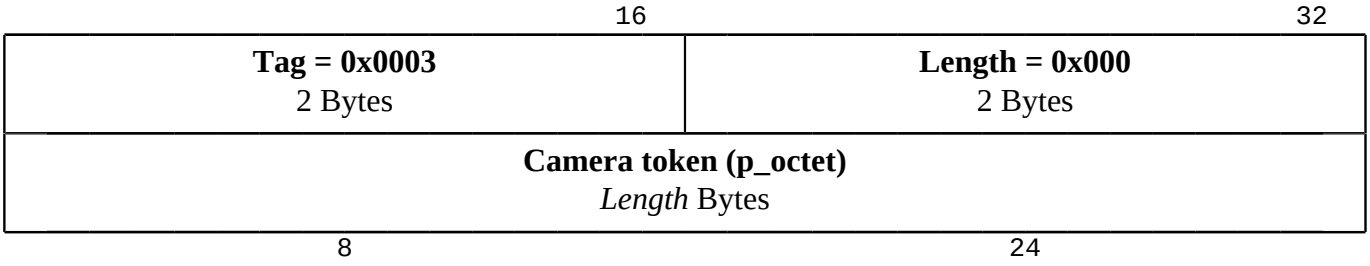


Remote camera line

In network order

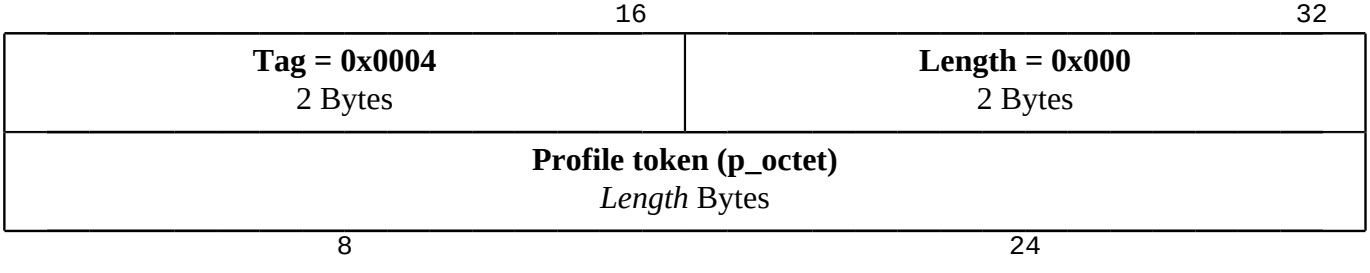
Tag 3: Camera token

see detailed description for camera token



Tag 4: Profile token

see detailed description for profile token



© Copyright 2001-2021 Bosch Security Systems

2.635 CONF_STREAMING_GATEWAY_MAX_LINES

[API: streaming gateway]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b26	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the max number of configurable streaming gateway lines	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.636 CONF_STREAMING_VAL

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01b9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	t_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

set multicast streaming mode to off	0
set multicast streaming mode mpeg4 to on	1
set multicast streaming mode mpeg2 to on	2
set multicast streaming mode mpeg4+mpeg2 to on	3

© Copyright 2001-2021 Bosch Security Systems

2.637 CONF_SUBNET

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0002	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the unit's subnet	
Write	t_dword	service	Set the unit's subnet	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Default	0	
Primary IP Subnet	1	
Auto IP Subnet	2	read direction

© Copyright 2001-2021 Bosch Security Systems

2.638 CONF_SUBNET_STR

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x007d	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the subnet mask using string notation (xxx.xxx.xxx.xxx)	
Write	p_string	service	Set the subnet mask using string notation (xxx.xxx.xxx.xxx)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Num Descriptor Values

Default	0	
Primary IP Subnet	1	
Auto IP Subnet	2	read direction

© Copyright 2001-2021 Bosch Security Systems

2.639 CONF_SUPPORTED_UPLOAD_TARGETS

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b19	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns all supported upload targets for this platform (List of n DWORDs)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.640 CONF_SYSCONTACT

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00ba	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the system contact required in SNMP	
Write	p_string	service	Write the system contact required in SNMP	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.641 CONF_SYSLOCATION

[API: snmp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00bb	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the system location required in SNMP	
Write	p_string	service	Write the system location required in SNMP	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.642 CONF_SYSLOG_HOST_STR

[API: syslog]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0950	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Get the SYSLOG host ip address	
Write	p_string	service	Set the SYSLOG host ip address	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.643 CONF_SYSLOG_LOG_LEVEL

[API: syslog]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c5e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the SYSLOG log level EMERGENCY=0 .. DEBUG=7 RFC 5424; value must be one above the highest allowed event, so 0= all off	
Write	t_octet	service	Set the SYSLOG log level EMERGENCY=0 .. DEBUG=7 RFC 5424; value must be one above the highest allowed event, so 0= all off	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.644 CONF_SYSLOG_MESSAGE

[API: syslog]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0caa	0: default log level. 1: Emergency, 2: Alert, 3: Critical, 4: Error, 5: Warning, 6: Notice, 7: Info, 8: Debug	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	service	Write a message string to the SYSLOG system	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.645 CONF_SYSLOG_PORT

[API: syslog]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c69	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Reads the SYSLOG host network port	
Write	t_word	service	Set the SYSLOG host network port	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.646 CONF_SYSLOG_PROTOCOL

[API: syslog]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c5f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Get the SYSLOG server protocol (0 = UDP, 1=TCP, 2=TLS)	
Write	t_octet	service	Set the SYSLOG server protocol (0 = UDP, 1=TCP, 2=TLS)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.647 CONF_SYSTEM_DATETIME_V2

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba8	if set to 0xFFFF the daylight saving values are not overwritten if not present	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Gets the system date and time and the daylight savings in one step: yyyy-mm-ddThh:mm:ss; PosixTimezone; optional string: yyyy->year, mm->month, dd->day, T->beginning of the time section, hh->hour, mm->minute, ss->second (all time values are utc time); The Time Zone format is specified by POSIX, refer to POSIX 1003.1 section 8.3. (the PosixTimezone string is only included if written by that command. If not present the corresponding settings are cleared).	
Write	p_string	service	Writes the system date and time and the daylight savings in one step: yyyy-mm-ddThh:mm:ss; PosixTimezone; optional string: The Time Zone format is specified by POSIX, refer to POSIX 1003.1 section 8.3	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.648 CONF_SYSTEM_LOAD

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cb5	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 1: CPU_LOAD_OF_CORE_N

Container tag which contains information about current CPU-load on a single CPU-core

Length 2 Bytes	**Tag = 0x0001** 2 Bytes
CPU_LOAD_OF_CORE_N (p_octet) *Length* - 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

CPU_LOAD_OF_CORE_N

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 3: CORE_IDX

Index of the core described by the current container.

Length = 0x0021 2 Bytes	**Tag = 0x0003** 2 Bytes
CORE_IDX (t_octet) 1 Byte	

Tag 5: IDLE_PERC

Idle value for this core in percent

Length = 0x0021 2 Bytes	**Tag = 0x0005** 2 Bytes
IDLE_PERC (t_octet) 1 Byte	

© Copyright 2001-2021 Bosch Security Systems

Tag 6: ENC_PERC

CPU-load in percent caused by the encoder on this core. Only supported on CPP6/CPP7/CPP7.3, '0' otherwise

Length = 0x0021 2 Bytes	**Tag = 0x0006** 2 Bytes
ENC_PERC (t_octet) 1 Byte	

Tag 7: VCA_PERC

CPU-load in percent caused by the VCA on this core. Only supported on CPP6/CPP7/CPP7.3, '0' otherwise

Length = 0x0021 2 Bytes	**Tag = 0x0007** 2 Bytes
VCA_PERC (t_octet) 1 Byte	

Tag 2: RECORDER_N

Container tag which contains information about a local recording media.

The availability of a index in this tag depends on the availability of a media slot on the device, and not on wether a media is currently plugged.

This means: If a device has 2 SD-Card slots then this command always returns 2 descriptors (idx 0,1) also when only one or none card is plugged.

Length 2 Bytes	**Tag = 0x0002** 2 Bytes
RECORDER_N (p_octet) *Length* - 4 Bytes	

© Copyright 2001-2021 Bosch Security Systems

RECORDER_N

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 4: REC_IDX

Index of the media slot, e.g. SD-Card, described by the current container.

Length = 0x0021 2 Bytes	**Tag = 0x0004** 2 Bytes
REC_IDX (t_octet) 1 Byte	

Tag 8: RECORDING_LOAD_PERC

Current load of a local recording media in percent

Length = 0x0021 2 Bytes	**Tag = 0x0008** 2 Bytes
RECORDING_LOAD_PERC (t_octet) 1 Byte	

© Copyright 2001-2021 Bosch Security Systems

Tag 9: RECORDING_MEDIA_STATUS

Current status of a local recording media

Length = 0x0024 2 Bytes	**Tag = 0x0009** 2 Bytes
RECORDING_MEDIA_STATUS (t_dword) 4 Bytes	

Values:

No device plugged	1
Device detected and can be used	2
Hardware init failed	3
Hardware driver IO error	4
Hardware CRC error	5
Hardware IO timeout	6
Warning, Media has sometimes slow IO	7
Warning, Media is read-only (Either write-protected or end of life)	8
Media not suitable	9
Media too small	10

© Copyright 2001-2021 Bosch Security Systems

2.649 CONF_SYSTEM_OS_VERSION

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d03	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Get the S&ST OS Version	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.650 CONF_SYSUPTIME

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x00b9	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Read the system uptime in seconds	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.651 CONF_TARGET_ID_RESOLVE_RULES

[API: storage]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c0a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the target id resolve rules, see detailed description	
Write	p_octet	service	Set the target id resolve rules, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command configures the rules for resolving a storage target id to the parameter for connecting the storage server. The target id is an 32 bit system wide unique identifier and replaces the ipv4 addresses for storages. The display format of an id in text will be "xxx.xxx.xxx.xxx" like ipv4 addresses. There are special values for the target id, which are 0.0.0.0 for invalid target id, 255.255.255.255 for wildcard and 127.0.0.1 for the local target in order to be backward compartible with older devices and software which expect an ipv4 address instead. The default state for the configured rules containes only one rule for the complete range 0.0.0.1 to 255.255.255.254 and lowest priority 0 for DIRECT_IPV4 for backard compartibility. In order to change or add rules, all which shall be changed or added needs to be send including the table entry index. Entries which shall not be changed doesn't need to be send. Also the read response payload will only contain valid rules, except if the table is empty, in that case only one empty rule will be in the payload.

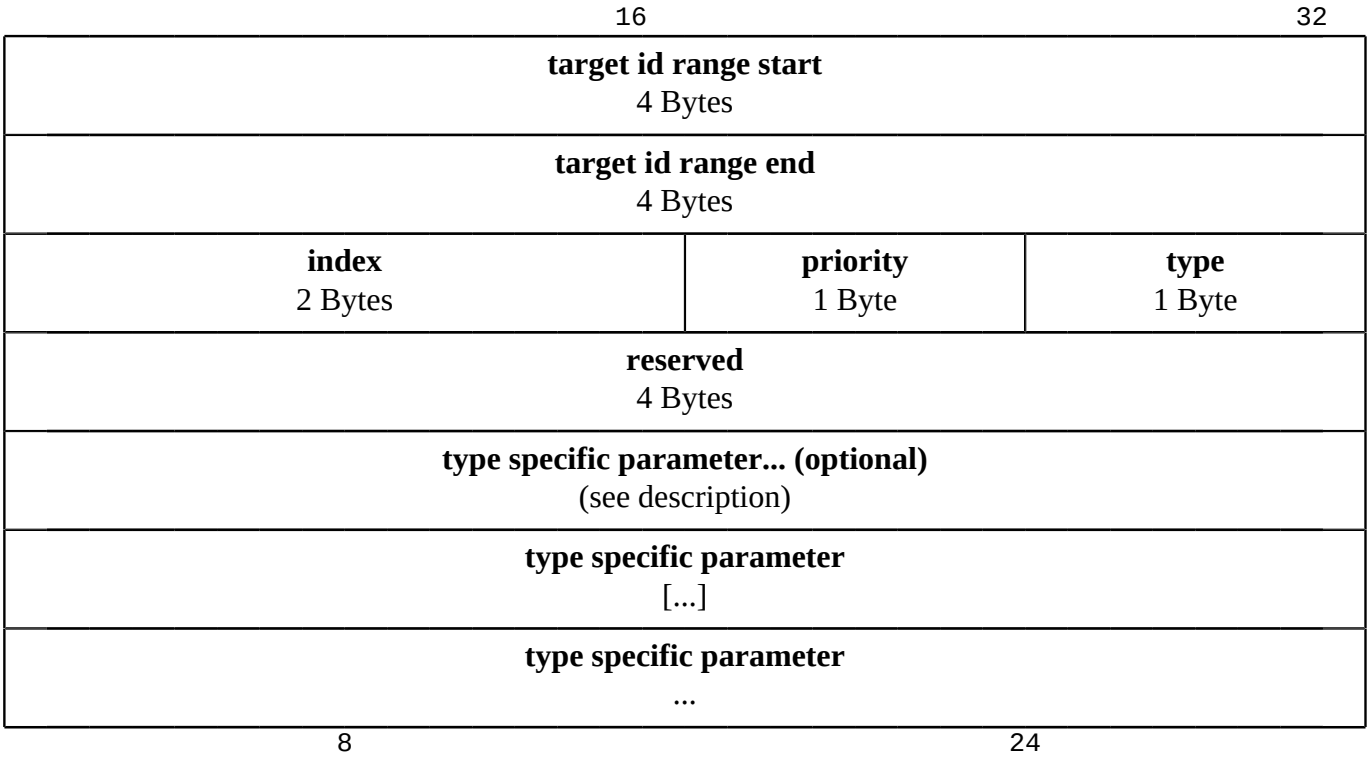
Payload Structure

target id resolve rule [0] (see description)
...
target id resolve rule [N] (see description)

target id resolve rule

Up to 127 target id resolve rules. The payload contains all rules to be changed or to be added in write direction or all valid rules for read direction, at least one zeroed rule if there are no rules actual configured.

© Copyright 2001-2021 Bosch Security Systems



target id range start

Begin of the target id range including the specified id of the rule.

target id range end

End of the target id range including the specified id of the rule.

index

Index of the target id resolve table, where to change the actual entry. Valid range is 0 to 127, as the table has an actual size of 128 entries.

priority

The rules target id range my overlap, therefor the priority will be used, which rule wins in conflic case. The valid range is from 0 to 255, where 0 is the lowest priority and 255 the highest.

type

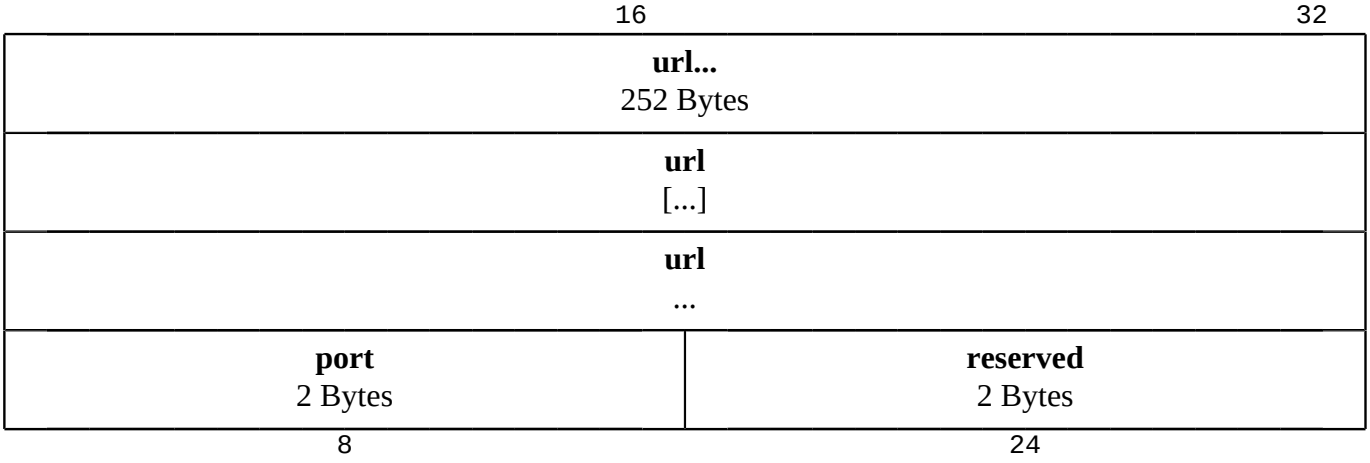
The rules target id range my overlap, therefor the priority will be used, which rule wins in conflic case. The valid range is from 0 to 255, where 0 is the lowest priority and 255 the highest.

NONE	0x00	no rule (delete rule)
DIRECT_IPV4	0x01	target id is the ipv4
ISCSI_SERVER	0x02	iscsi server parameter
SMB_SERVER	0x03	smb server parameter

© Copyright 2001-2021 Bosch Security Systems

type specific parameter (optional)

'type specific parameter' payload for 'type' = 'ISCSI Server' (0x02)



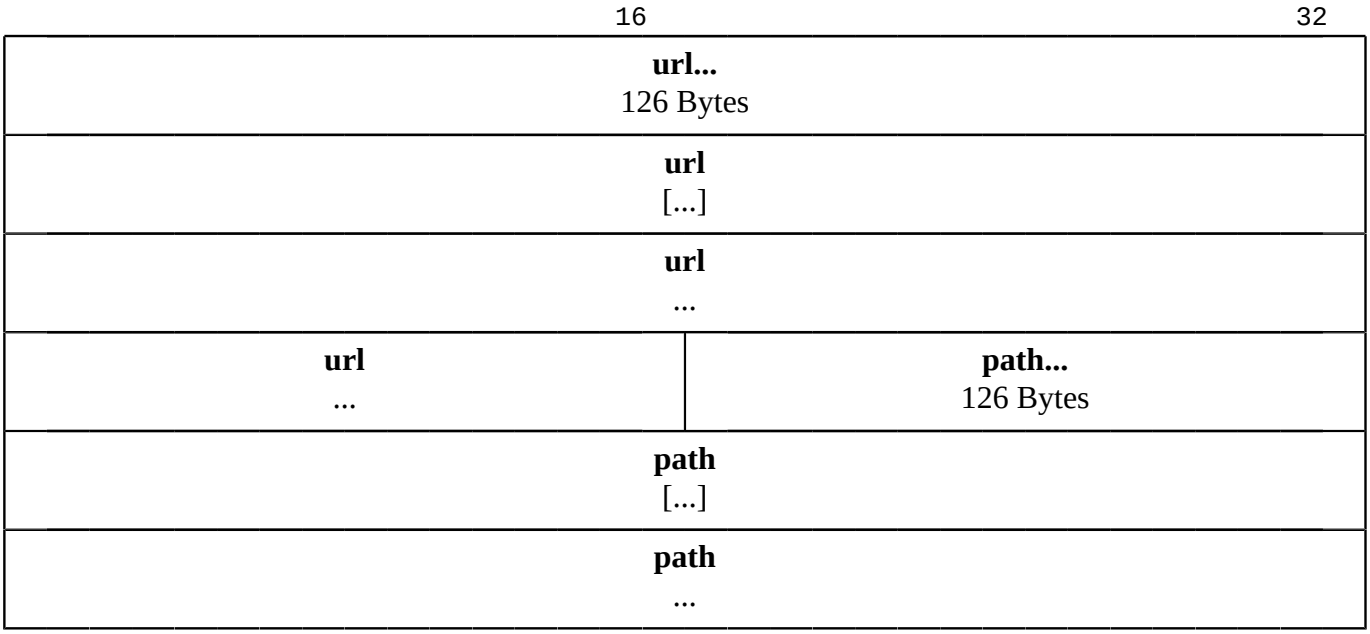
url

The url of the iscsi server as zero terminated ascii string. Only ipv4 as string is actual supported.

port

The port of the iscsi service socket on the server. Default 0 means use the default port. Actual it is just a place holder and other ports than the the default port are not supported.

'type specific parameter' payload for 'type' = 'SMB Server' (0x03)



© Copyright 2001-2021 Bosch Security Systems

path ...	
port 2 Bytes	**reserved** 2 Bytes
8	24

url

The url of the smb server as zero terminated ascii string. Only ipv4 or ipv6 as string is actual supported.

path

The path on the smb server as zero terminated ascii string.

port

The port of the iscsi service socket on the server. Default 0 means use the default port. Actual it is just a place holder and other ports than the the default port are not supported.

Note: On the Generic (Windows and Linux) the whole table will be persisted in the table. The generic dll won't persist it. All other devices will only persist the first four valid rules.

© Copyright 2001-2021 Bosch Security Systems

2.652 CONF_TCP_BANDWIDTH_CHECK

[API: transcoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b64	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get tcp bandwith check max duration in ms, sessionID to address session based params	
Write	t_dword	service	Set tcp bandwith check max duration in ms, sessionID to address session based params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.653 CONF_TCP_BANDWIDTH_CHECK_RESULT

[API: transcoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b66	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	user	Get result of tcp_bandwidth_check (session id required), 4bytes error, 4bytes kbps	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.654 CONF_TCP_FWD

[API: transcoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b2f	yes (Forwarder index 1...max, max=4: devices, max=32: generic)	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Reads the Settings of the tcp forwarder per device (WORD http listener port, WORD http forwarder port, WORD https listener port, WORD https forwarder port, string: ip)	
Write	p_octet	service	Writes the Settings of the tcp forwarder per device (WORD http listener port, WORD http forwarder port, WORD https listener port, WORD https forwarder port, string: ip)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.655 CONF_TCP_RATE_CONTROL

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b4c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Enables/disables tcp rate control	
Write	t_dword	service	Enables/disables tcp rate control	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.656 CONF_TELNET_PORT

[API: debug]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a18	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Get the local Telnet TCP port number	
Write	t_word	service	Set the local Telnet TCP port number	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.657 CONF_TEMP_SENS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09c5	temperature sensor	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Value of the temperature sensor in tenths of degrees Celsius, specified by numdes, in case of negative temperature, the value wraps as 32 bit DWORD and needs to be casted to int	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.658 CONF_TIME_HRS

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet p_string	minimal	Read the hours	
Write	t_octet p_string	service	Set the hours, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.659 CONF_TIME_MIN

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet p_string	minimal	Read the minutes	
Write	t_octet p_string	service	Set the minutes, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.660 CONF_TIME_SC_CONNECT_FAIL_MSG

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ce0	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.661 CONF_TIME_SEC

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x002b	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet p_string	minimal	Read the seconds	
Write	t_octet p_string	service	Set the seconds, not supported while recording is running or configured to active	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.662 CONF_TIME_STAMP_RESOLUTION

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a7f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	0=second, 1=ms	
Write	t_octet	user	0=second, 1=ms	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.663 CONF_TIME_STAMP_VAL

[API: stamping]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0085	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Deprecated! Use CONF_STAMP instead.	
Write	t_octet	user	Deprecated! Use CONF_STAMP instead.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.664 CONF_TIMEZONE

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x024e	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	The timezone in which the unit has to operate (GMT +- nbr of seconds +- nbr of seconds DLS)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.665 CONF_TPPP_PORT

[API: serial port]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Gets the port of the third party protocol proxy (0 means off)	
Write	t_word	service	Sets the port of the third party protocol proxy (0 means off)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.666 CONF_TRAFFIC_SHAPER_LIMIT

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d16	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Set the network traffic shaper to a limit value of x KBit/s; 0=Limiter off	
Write	t_dword	service	Set the network traffic shaper to a limit value of x KBit/s; 0=Limiter off	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.667 CONF_TRANSCODER_CAPABILITIES

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c0d	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns capabilities of a transcoder device in a tagged format	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Payload Structure

Number of tagged elements 4 Bytes
Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

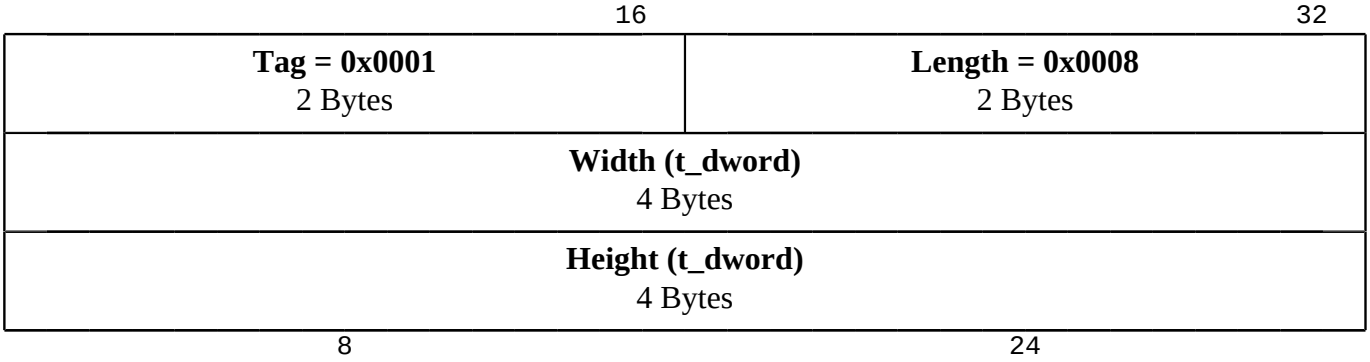
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

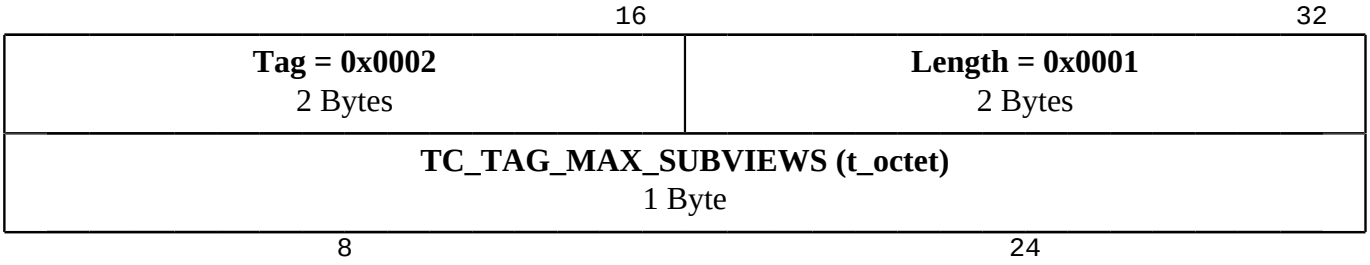
Tag 1: TC_TAG_MAX_INPUT_RESOLUTION

maximum transcoder input resolution



Tag 2: TC_TAG_MAX_SUBVIEWS

maximum transcoder subviews



© Copyright 2001-2021 Bosch Security Systems

2.668 CONF_TRANSCODER_INFORMATION

[API: rcp.connection]

	Tag Code	Num Descriptor	Message	SNMP Support
	0xd060	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value without length and tag field

Tag

Tag specifying the encoding and meaning of the value

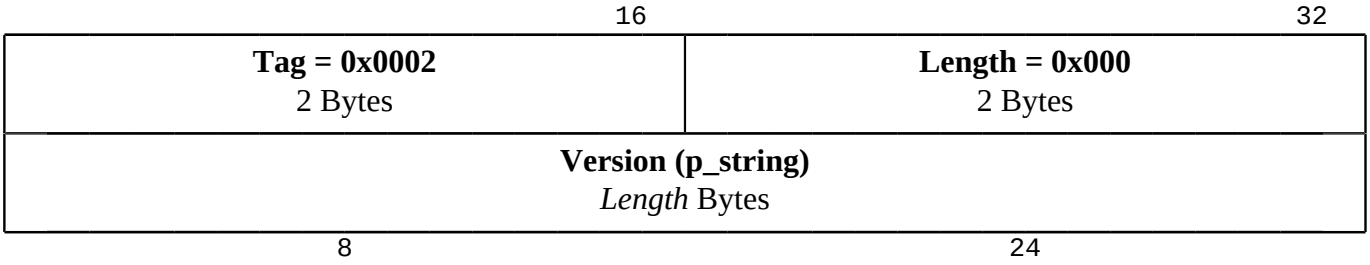
Tag 1: Transcoder state

16	32
Tag = 0x0001 2 Bytes	**Length = 0x0002** 2 Bytes
Transcoder state (t_word) 2 Bytes	
8	24

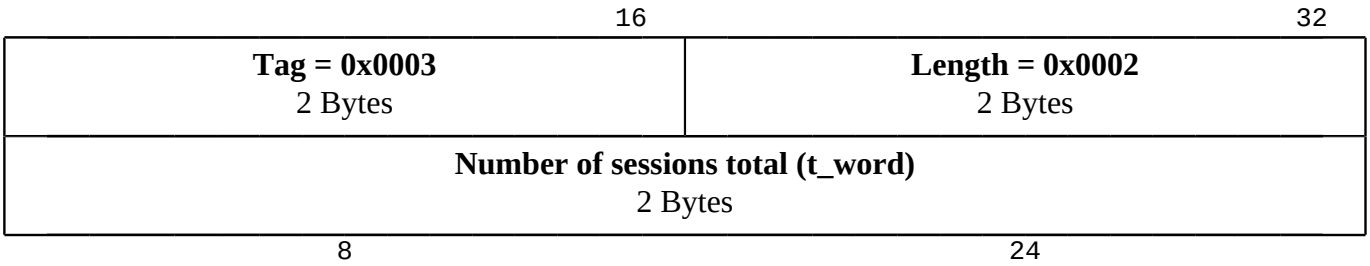
© Copyright 2001-2021 Bosch Security Systems

Not present	0
Connected	1

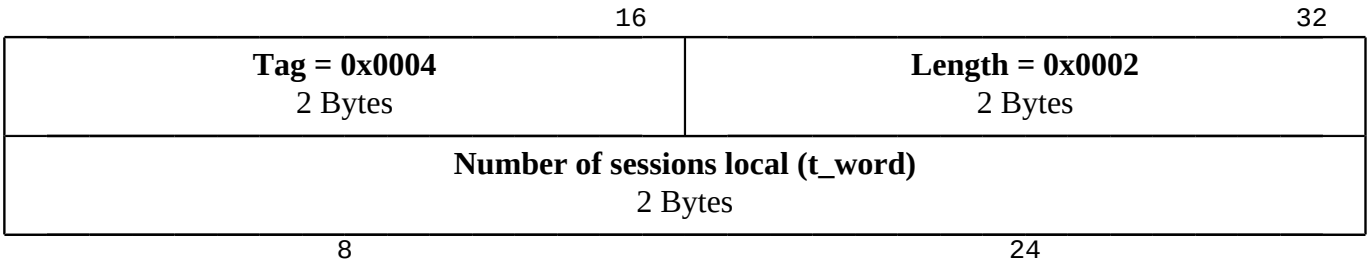
Tag 2: Version



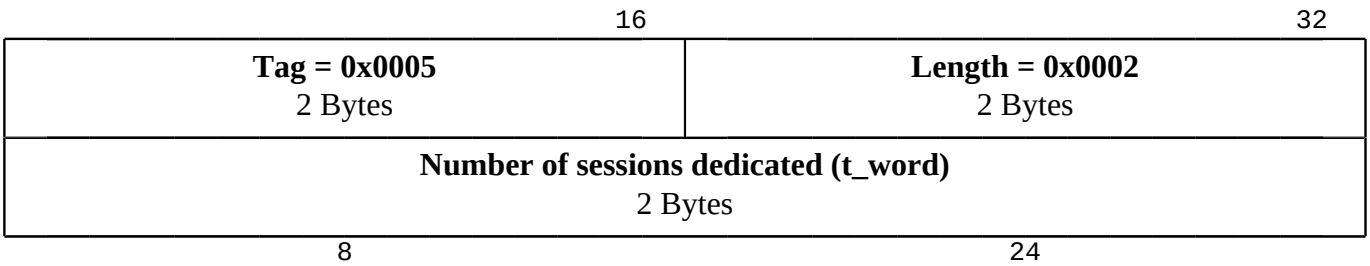
Tag 3: Number of sessions total



Tag 4: Number of sessions local

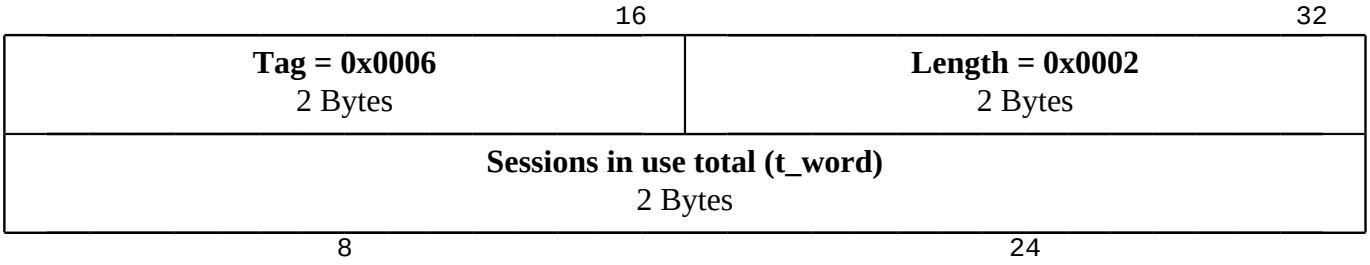


Tag 5: Number of sessions dedicated

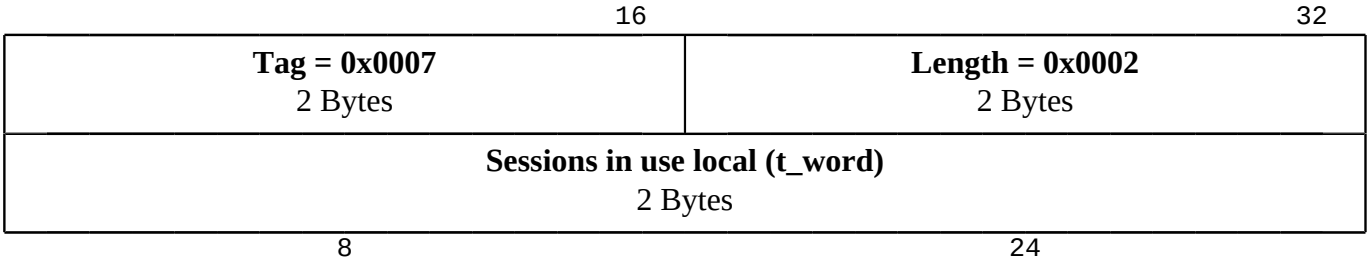


© Copyright 2001-2021 Bosch Security Systems

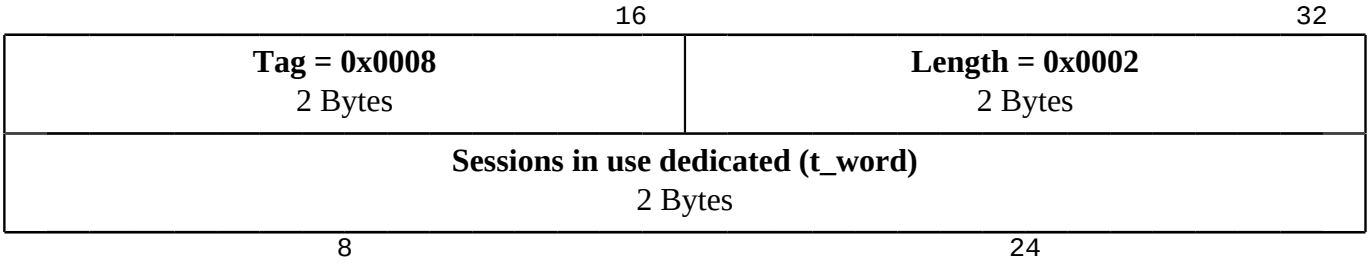
Tag 6: Sessions in use total



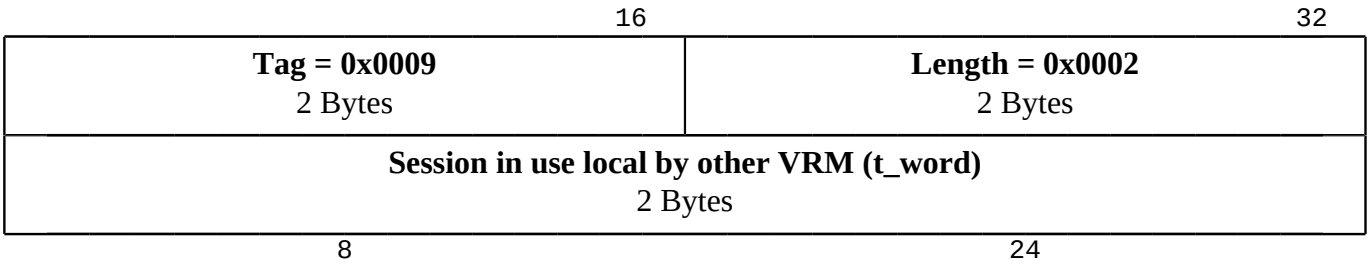
Tag 7: Sessions in use local



Tag 8: Sessions in use dedicated

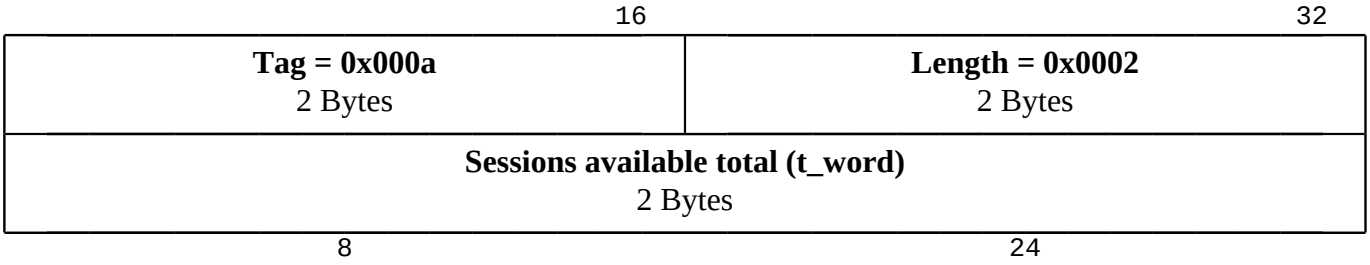


Tag 9: Session in use local by other VRM

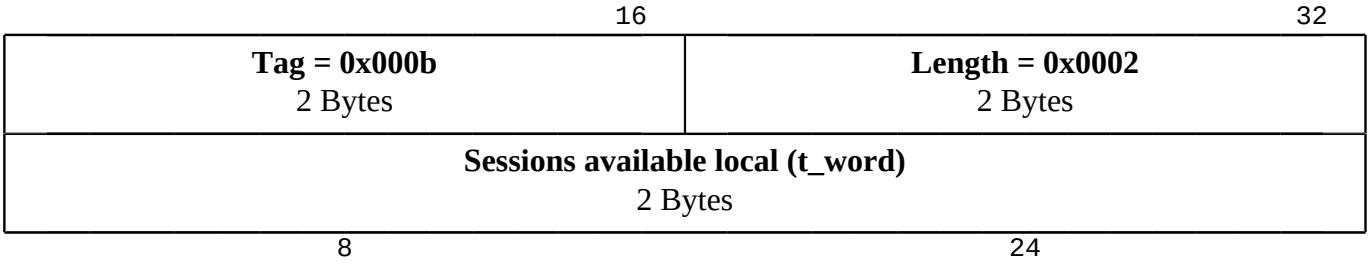


© Copyright 2001-2021 Bosch Security Systems

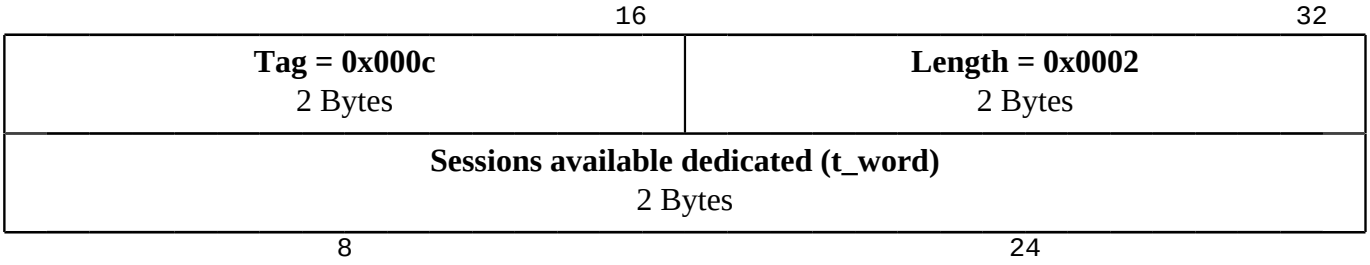
Tag 10: Sessions available total



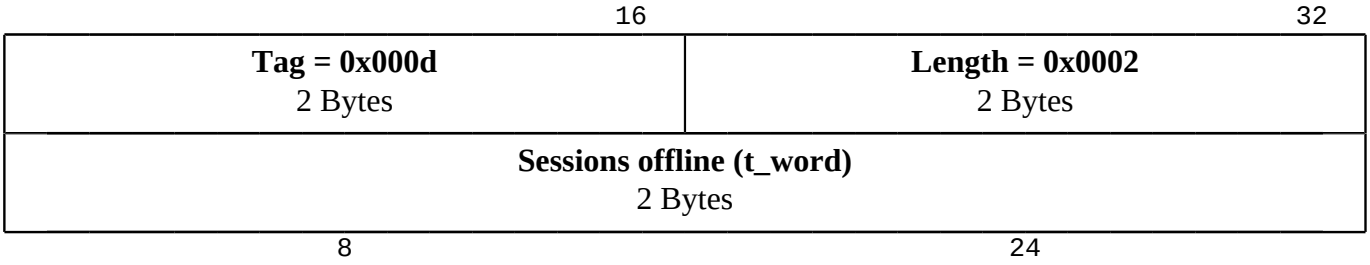
Tag 11: Sessions available local



Tag 12: Sessions available dedicated



Tag 13: Sessions offline



© Copyright 2001-2021 Bosch Security Systems

2.669 CONF_TRANSPARENT_DATA_OVER_IP

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af2	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Check availability in device capabilities, set IP-address and port to accept transparent data, first, 4 bytes define the IP-address (0 represents off) and second, two bytes are the defined port	
Write	p_octet	iva	Check availability in device capabilities, set IP-address and port to accept transparent data, first, 4 bytes define the IP-address (0 represents off) and second, two bytes are the defined port	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.670 CONF_UNIT_ID

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0025	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Read the unit ID	
Write	p_unicode	service	Set unit ID (max 31 unicode character + null delimiter)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.671 CONF_UNIT_NAME

[API: system.settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0024	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	always_legacy	Read the unit name	
Write	p_unicode	service	Set unit name (max 31 unicode character + null delimiter)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.672
CONF_UNSOLICITED_AUTODETECT_REPLY_TIME

[API: rcp.discovery]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0957	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the time in seconds, when the VJ shall send out unsolicited autodetect reply to the RCP port(off when set to 0)	
Write	t_dword	service	Set the time in seconds, when the VJ shall send out unsolicited autodetect reply to port 1800 (off when set to 0); uses broadcast, and multicast if AUTODETECT_REPLY_GROUP is set	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.673 CONF_UPLINK_KBPS

[API: ethernet]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c03	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	service	Uplink bandwidth of system in kbps	
Write	t_dword	service	Uplink bandwidth of system in kbps	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.674 CONF_UPLOAD_HISTORY

[API: system.status]

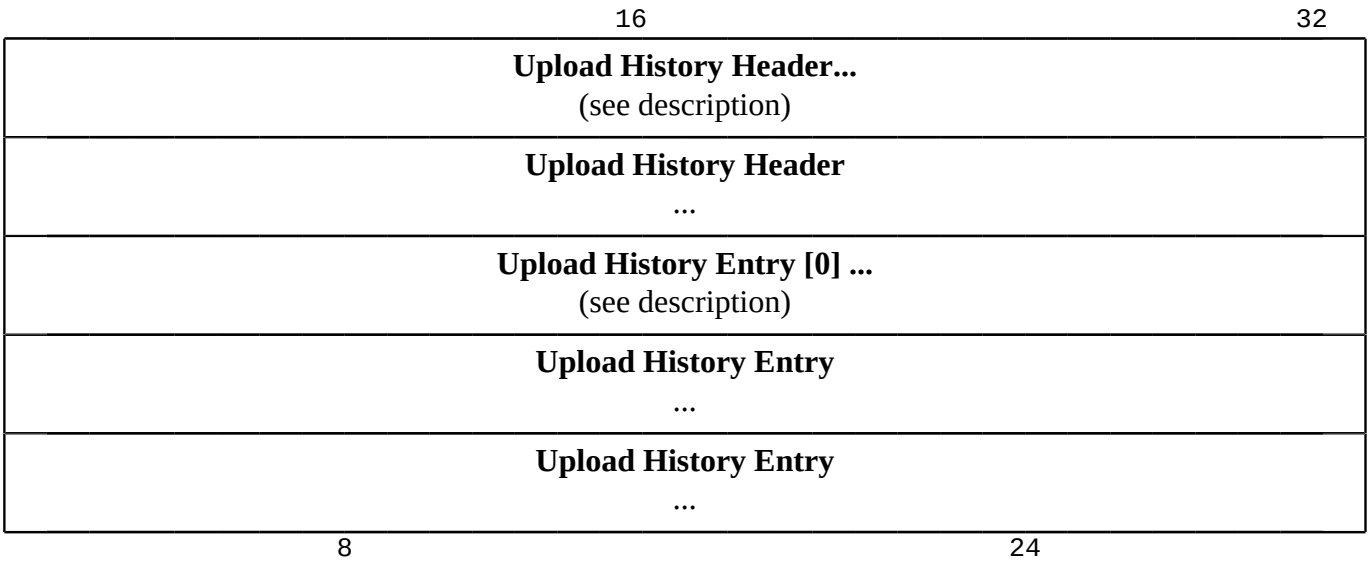
	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b44	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_UPLOAD_HISTORY allows to get insights on what was uploaded to the queried device lately. The order of the entries in the upload history is always a chronological order. New entries are appended at the end. In case all ten entries of the list are filled, the oldest entry is overwritten. Therefore, the Latest Entry number in the header needs to be considered to find the latest entry in the list. The 2nd newest entry is the preceding one in the list, and so on. While finding timely preceding entries, make sure to wrap in the list to the last entry, where necessary. Handling the order of the entries in the list this way makes parsing slightly more complicated than absolutely necessary, but helps to raise the MTBF of the flash hardware. Considering these two aspects lead to the aforementioned organization of the upload history.

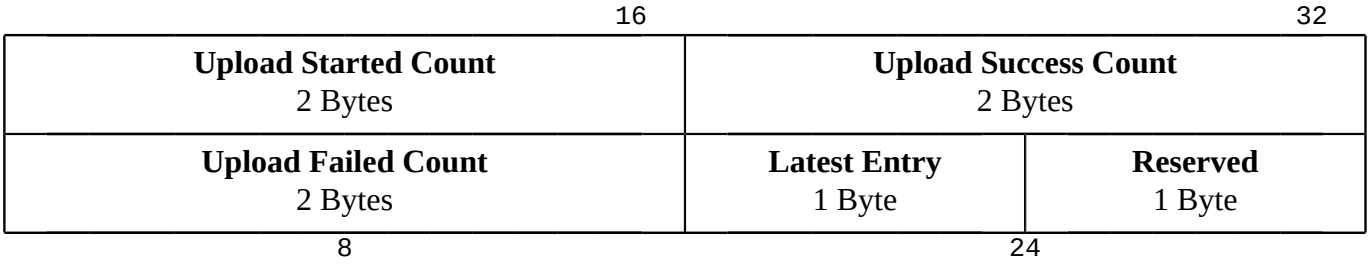
Uploads interrupted by power failure and consequent recovery measures like upload of a new firmware via the recovery application APP0 can make it necessary to interpret the content of the upload history manually with the help of the given flags, the firmware version, the file type and the date.

Response Payload Structure



© Copyright 2001-2021 Bosch Security Systems

Upload History Header



Upload Started Count

Number of Uploads started on this Device.

Upload Success Count

Number of Uploads completed successfully on this Device.

Upload Failed Count

Number of Uploads completed with an Error on this Device.

Latest Entry

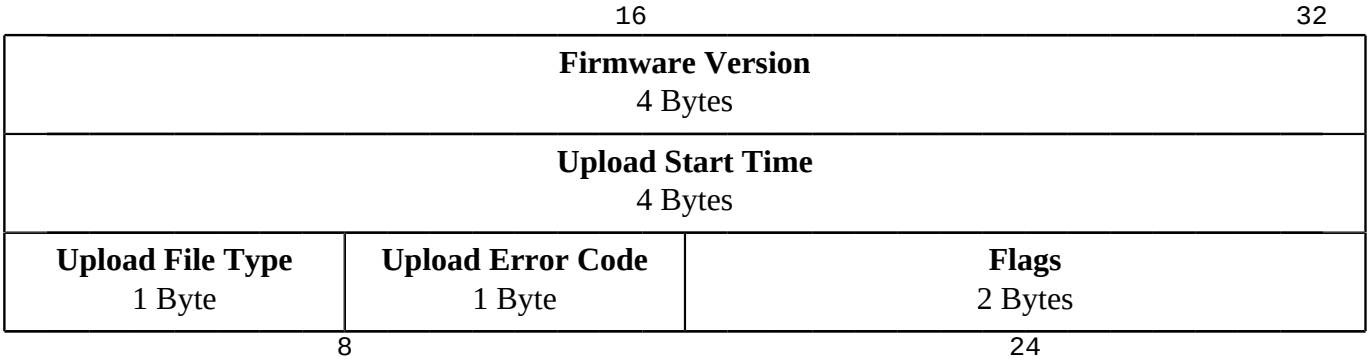
Number that points to the most recent entry in the upload history.

Reserved

Reserved Bits.

Upload History Entry

The upload history consists of 10 entries. Not more, not less. Format of each entry is the following:



Firmware Version

Version Information of the Firmware, that was uploaded.

© Copyright 2001-2021 Bosch Security Systems

Upload Start Time

Time, when the upload was started. Format: Seconds since 1st of January of the year 2000

Upload File Type

File type that was uploaded.

Possible Values:
BSP_ROM_AREA_UNKNOWN 0
BSP_ROM_AREA_APP_0 3
BSP_ROM_AREA_APP_1 4

Note: Other file type values are possible, but documentation of those is kept confidential.

Upload Error Code

Error code, that was encountered during upload.

Possible Values:
UPLOAD_GEN_ERR_NO_ERR 0
UPLOAD_GEN_ERR_FLUSH_NO_HEADER 1
UPLOAD_GEN_ERR_FLUSH_WRITE 2
UPLOAD_GEN_ERR_FLUSH_READBACK 3
UPLOAD_GEN_ERR_FLUSH_VERIFY 4
UPLOAD_GEN_ERR_FINALFLUSH_FLASH_CHKSUM 5
UPLOAD_GEN_ERR_FINALFLUSH_HEADER_CHKSUM 6
UPLOAD_GEN_ERR_MAGIC 10
UPLOAD_GEN_ERR_VERSION_TOO_LOW 11
UPLOAD_GEN_ERR_VERSION_VS_FLASH 12
UPLOAD_GEN_ERR_CHECK_DEVICE 13
UPLOAD_GEN_ERR_FILE_ENTRY_MARKER 14
UPLOAD_GEN_ERR_CHUNK_TOO_BIG 15
UPLOAD_GEN_ERR_AREA_ALREADY_WRITTEN 16
UPLOAD_GEN_ERR_BLACK_WHITE_LIST_CHECK 17
UPLOAD_GEN_ERR_SIGNATURE_REQUIRED 18
UPLOAD_GEN_ERR_SIGNATURE_INVALID 19

Flags

Flags related to this upload history entry.

	Mask	Name	Description
Bit 4	0x0010	DEVICE_RESTARTED	If set, the device restarted after this upload successfully.
Bit 3	0x0008	UPLOAD_SUCCESS	If set, the upload was finished successfully.

© Copyright 2001-2021 Bosch Security Systems

Bit 2	0x0004	UPLOAD_FAILED	If set, the upload was finished with an error.
Bit 1	0x0002	UPLOAD_FINISHED	If set, the upload was finished at all.
Bit 0	0x0001	UPLOAD_STARTED	If set, the upload was started.

Note: For a successful upload, all defined bits should be set, except for the UPLOAD_FAILED bit.

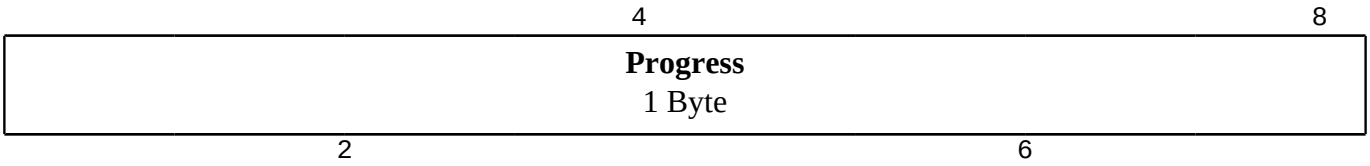
© Copyright 2001-2021 Bosch Security Systems

2.675 CONF_UPLOAD_PROGRESS

[API: system.status]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0701	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



Progress

Informs about current progress of an upload or possible error states. Values in the range of 1 ... 100 represent the upload progress in percent. Other values need to be handled as error during the upload and inform about the kind of the error that was encountered during the upload.

Percent	1 - 100	
header error	101	
write error	102	
read back error	103	
verify error	104	
checksum mismatch written data	105	read back flash content vs. written data
checksum mismatch received data	106	announced in header vs. received data
magic error	110	
version too low	111	
flash type incompatible	112	
device check failed	113	
file entry marker failed	114	
chunk size error	115	
area already written	116	
black white list check	117	
wrong or no signature	118	
signature invalid	119	
invalid file name	130	
ROM init error	131	

© Copyright 2001-2021 Bosch Security Systems

ROM write error	132
ROM close error	133
socket error	134
overflow error	135
final flush error	136
file format error	137
logo size error (size of picture is limited to 128x128 pixel)	138
logo compression error (only uncompressed bitmap files allowed)	139
logo colour error (max. 256 colour bitmap supported)	140
certificate/key already exists (filename must be unique)	141
certificate/key format error	142
no matching certificate found for uploaded key (upload certificate first or upload combined file)	143
no free key entry available	144
no certificate storage space available	145
device not fully booted	146
password required for this upload (encrypted config file)	147
no memory to store the file	148
file is to large for that type of upload	149

© Copyright 2001-2021 Bosch Security Systems

2.676 CONF_UPNP_SEARCH_IP_CONN_SERVICE

[API: upnp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b59	if set to 0xFFFF a extended payload is returned	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Searches for devices that support the upnp WANIPConnection service. Returns a list of devices entries (64 bytes addr, 64 bytes name, opt. 64 byte unique id (if extended payload is requested))	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.677 CONF_UPNP_TCP_FWD

[API: upnp]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b5a	index of the remote device (1...4); local device: 0	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Configures port forwarding in a upnp device that support the WANIPConnection:1 service. Input 64 bytes: address of the WANIPConnection:1 service in the router (returned by CONF_UPNP_SEARCH_IP_CONN_SERVICE	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.678 CONF_USER_AUTH_MODE

[API: user management]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	live		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

Command CONF_USER_AUTH_MODE allows to query or enable/disable user authentication modes.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

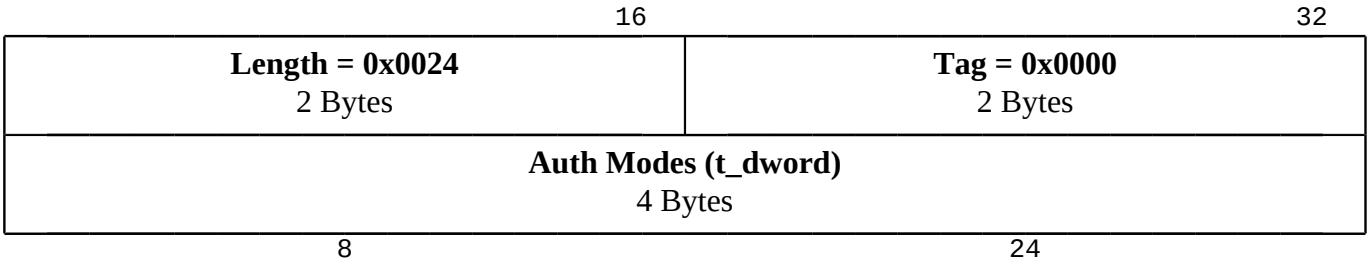
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 0: Auth Modes

tag specifying the encoding and meaning of the value



	Mask	Name
Bit 2	0x00000004	Active Directory (ADFS) Auth
Bit 1	0x00000002	Certificate Auth
Bit 0	0x00000001	Password Auth

© Copyright 2001-2021 Bosch Security Systems

2.679 CONF_UTC_ZONEOFFSET

[API: local time and time server settings and configuration]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x031f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	The timezone in which the unit has to operate (GMT +- nbr of seconds)	
Write	t_int	service	The timezone in which the unit has to operate (GMT +- nbr of seconds)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.680 CONF_VCA_BRIGHTNESS_INFO

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c25	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Luminance (1..255), the value 0 means N/A	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.681 CONF_VCA_MASK_DOME_PTZ_POS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c70	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

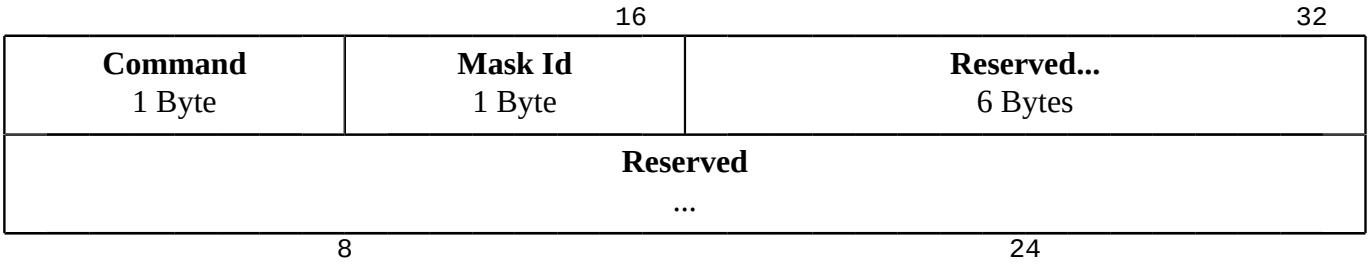
Tag specifying the encoding and meaning of the value

Tag 0: vca mask dome ptz

16	32
Length 2 Bytes	**Tag = 0x0000** 2 Bytes
vca mask dome ptz (p_octet) *Length* - 4 Bytes	
8	24

© Copyright 2001-2021 Bosch Security Systems

vca mask dome ptz



Command

MoveDomeToDefinedPos	0	Move dome to position stored at maskId

Mask Id

Use CONF_VCA_MSK_OPTIONS to query number of supported masks (@CONF_VCA_MSK_OPTIONS Byte[4] maxNrOfVcaMasks per line). Define maskID which is used in conjunction with the command

© Copyright 2001-2021 Bosch Security Systems

2.682 CONF_VCA_MSK_OPTIONS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6f	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16 32

reserved 1 Byte	**Rectangle Support** 1 Byte	**Polygon Support** 1 Byte	**Max Number Of Polygon Vertices** 1 Byte
Max Number Of Priv Masks Per Line 1 Byte	**reserved** 1 Byte	**reserved** 1 Byte	**rcp vca mask support @ dome** 1 Byte
reserved 1 Byte	**reserved** 1 Byte	**reserved** 1 Byte	**reserved** 1 Byte
reserved 1 Byte	**reserved** 1 Byte	**reserved** 1 Byte	**reserved** 1 Byte

8 24

Rectangle Support

No	0
Yes	1

Polygon Support

No	0	
Yes	1	May not be self intersecting

Max Number Of Polygon Vertices

Maximum points for polygon definition

Max Number Of Priv Masks Per Line

Maximum Number Of VCA Masks Per Line

© Copyright 2001-2021 Bosch Security Systems

rcp vca mask support @ dome

No	0
Yes	1

© Copyright 2001-2021 Bosch Security Systems

2.683 CONF_VCA_MSK_POLY

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c6e	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	iva	Configure polygon VCA mask (aka 'virtual mask')	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 2: Polygon description

16	32
Length 2 Bytes	**Tag = 0x0002** 2 Bytes
Polygon description (p_octet) *Length* - 4 Bytes	
8	24

© Copyright 2001-2021 Bosch Security Systems

Polygon description

16			32
Enable 1 Byte	**Id** 1 Byte	**Number Of Vertices** 1 Byte	**Reserved...** 5 Bytes
Reserved ...			
Vertex [0] (see description)			
...			
Vertex [Number Of Vertices - 1] (see description)			
8			24

Enable

Inactive	0	vca mask is not active
Active	1	vca mask is active

Id

0... max number vca masks (defined via CONF_VCA_MSK_OPTIONS)

Number Of Vertices

0... max number points (defined via CONF_VCA_MSK_OPTIONS)

Vertex

16	32
X 2 Bytes	**Y** 2 Bytes
8	24

X

X Coordinate	0 - 32678

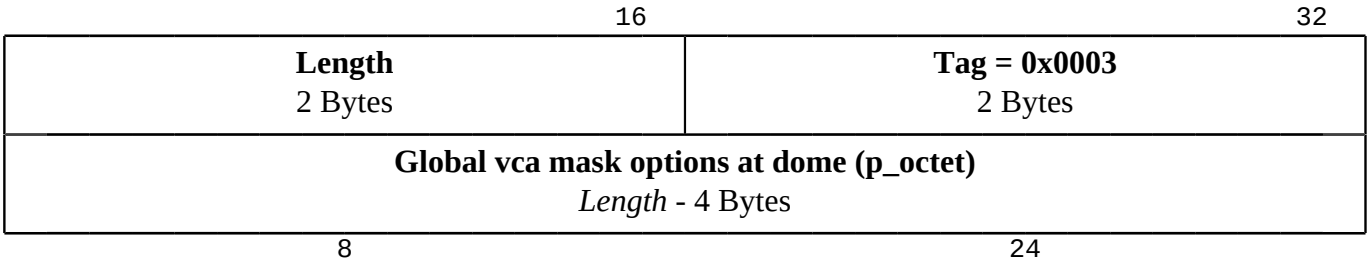
© Copyright 2001-2021 Bosch Security Systems

Y

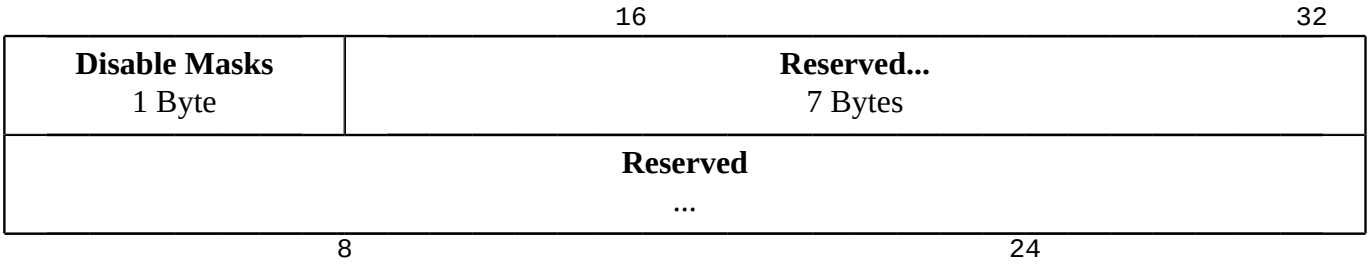
Y Coordinate 0 - 32678

Coordinate system for vca masking:
Upper left edge : (0,0)
Lower right edge: (32768,32768)

Tag 3: Global vca mask options at dome



Global vca mask options at dome



Use CONF_VCA_MSK_OPTIONS to query if supported (Byte[7] rcp vca mask support @ dome)

Disable Masks

Do not change 0
Disable Masks 1

© Copyright 2001-2021 Bosch Security Systems

2.684 CONF_VCA_ORIENTATION_INFO

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf4	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Reads the current difference in the orientation of the camera. The difference is an angle in units of 2*pi/2^32.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

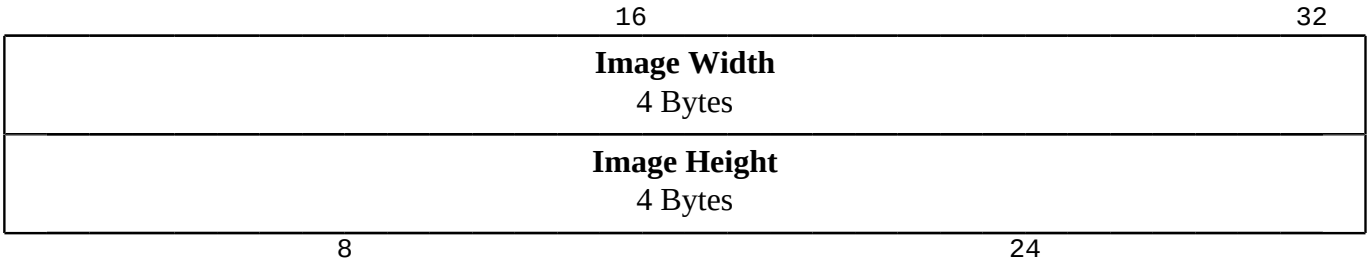
© Copyright 2001-2021 Bosch Security Systems

2.685 CONF_VCA_OUTPUT_FORMAT

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure



© Copyright 2001-2021 Bosch Security Systems

2.686 CONF_VCA_SHAPES

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc8	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	live	SessionID references the transcoder session; enable vca shape overlay	
Write	t_dword	live	SessionID references the transcoder session; enable vca shape overlay;	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.687 CONF_VCA_SUGGESTED_FIREMASK

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c31	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get suggested fire mask from algorithm (payload beyond the scope of this document)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.688 CONF_VCA_TASK_RUNNING_STATE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a96	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Returs the current processing state of the VCA engine; TRUE when vca engine is running with no problems; FALSE if vca engine lack cpu performance	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.689 CONF_VCD_OPERATOR_PARAMS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a1b	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read parametrization of vcd operator which corresponds to line <num>	
Write	p_octet	iva	Configures the vcd manager. if configuration fails an error is returned.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.690
CONF_VID_H264_ENC_BASE_OPERATION_MODE_CAPS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af9	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the base operation mode capability list of the h.264 encoders per line. The list contains all possible mode combinations for all streams.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

16	32
Number of streams 4 Bytes	
Mode Combination [0] (see description)	
...	
Mode Combination [N] (see description)	

8	24

Mode Combination

Mode [0] 4 Bytes
...
Mode [Number of streams - 1] 4 Bytes

© Copyright 2001-2021 Bosch Security Systems

Mode

See 'Stream Operation Mode' of command ' CONF_VIDEO_H264_ENC_BASE_OPERATION_MODE ' for details.

© Copyright 2001-2021 Bosch Security Systems

2.691 CONF_VID_IN_BRIGHTNESS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092a	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the brightness value (0-255)	
Write	t_octet	service	Sets the brightness value (0-255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.692 CONF_VID_IN_CONTRAST

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the contrast value (0-255)	
Write	t_octet	service	Sets the contrast value (0-255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.693 CONF_VID_IN_SATURATION

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x092c	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the saturation value (0-255)	
Write	t_octet	service	Sets the saturation value (0-255)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.694 CONF_VIDEO_ALARM_STATE

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x01c2	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=video alarm is off; 1=video alarm is on (no video signal detected at video input)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.695 CONF_VIDEO_BITRATE_AVERAGING_PERIOD

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0622	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the (maximum) period in seconds for bitrate averaging (0=no bitrate averaging)	
Write	t_dword	service	Sets the (maximum) period in seconds for bitrate averaging (0=no bitrate averaging)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.696 CONF_VIDEO_CURRENT_PARAMS_CODNBR

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0982	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the current profile preset number for the specified encoder (NumDesc)	
Write	t_dword	service	Sets the video encoder (given by NumDesc) to a preset (payload). If recording is running and the iframe distance is 0, the iframe distance of the preset will be patched to a recording suitable value. For transcoder: if the request contains the replay session id and no num descriptor, the device will assign the preset to the transcoder of the replay session. If the preset is empty the transcoder will be disabled. For exclusive (dual) stream instances: if the request contains the session id and the num descriptor is 0, the previously configured preset with its session based (non-permanent) modifications will be applied to the encoder.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.697 CONF_VIDEO_ENC_P_REF_LIST_SIZE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0627	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Max. number of reference frames used by a P-slices in H.26x streams with 0 = unrestricted	
Write	t_octet	service	Max. number of reference frames used by a P-slice in H.26x streams with 0 = unrestricted. Query largest support value via GET_P_REF_LIST_SIZE_LIMIT	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.698 CONF_VIDEO_ENC_PRIO

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a81	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Reads the video encoder priorisation 0:= no, 1:=h26x, 2:=jpeg, 3:=h26x_2nd stream	
Write	t_dword	service	Writes the video encoder priorisation 0:=no, 1:=h26x, 2:=jpeg, 3:=h26x_2nd stream	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.699 CONF_VIDEO_ENC_STREAMING

[API: multicast]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x099a	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Video streaming on coder (video encoder n: coderbits=1<<(n-1))	
Write	t_dword	service	Enables video streaming on coder (video encoder n: coderbits=1<<(n-1))	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.700 CONF_VIDEO_ENC_TEMP_QUANT_ADJ

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0626	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get temporary encoding quality adjustment see	
Write	p_octet	live	Temporary encoding quality adjustmens see	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

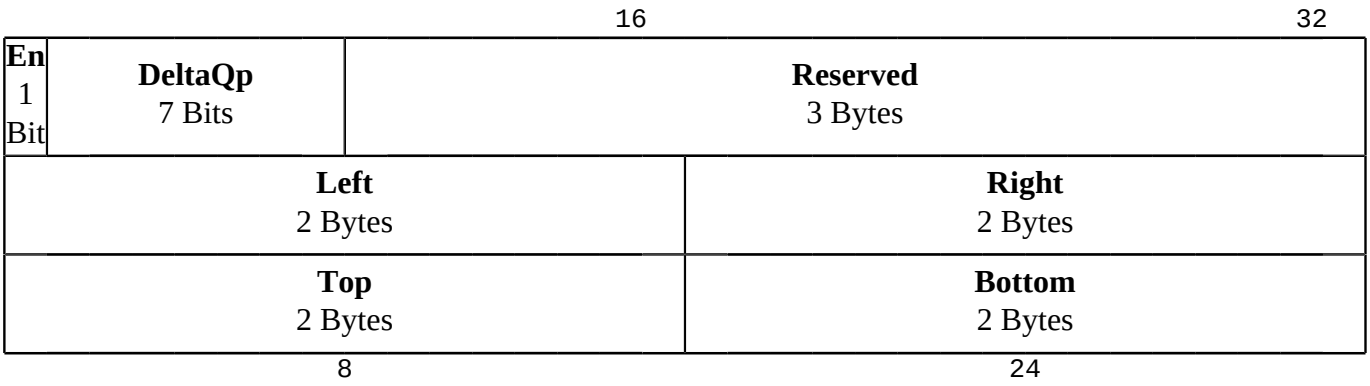
Description

The encoder, to which the quality adjustment is to be applied is identified via the RCP session.

The area, in which the qualtiy is adjusted is characterized by the position of the edges of a rectangle within a virtual coordinate system with a nominal range of [0..+32768]x[0..32768] for the complete encoded picture (0=top/left edge, 32768=bottom/right edge). The actually adjusted area will be enlarged as needed by the encoder.

Note, that the payload may contaion only first octet, if enable is set to 0

Payload Structure



En

Setup temporary quality adjustments:

Enable	0
Disable	1

© Copyright 2001-2021 Bosch Security Systems

DeltaQp

Intended quality adjustment in H.264 QP units. If enable is 0, this field must also be set to 0. Otherwise the allowed range is -51..51 with negative values indicating higher quality and positive ones lower quality.

Left

Relative position of the left edge of the adjustment region within the complete encoded image

Right

Relative position of the right edge of the adjustment region within the complete encoded image

Top

Relative position of the upper edge of the adjustment region within the complete encoded image

Bottom

Relative position of the lower edge of the adjustment region within the complete encoded image

© Copyright 2001-2021 Bosch Security Systems

2.701 CONF_VIDEO_ENCODER_ENCODED_BYTES

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0af6	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get encoded bytes (32 Bit Counter) since bootup. Only supported for h.264 and Jpeg encoder.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.702 CONF_VIDEO_ENCODER_STATUS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x09df	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get encoder status (8bytes) 4 (high order) bytes 10*frames per sec, 4 (low order) bytes KBPS	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.703 CONF_VIDEO_ENCODER_STATUS_EXT

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a90	video coder	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get encoder status (20 bytes) 1st DWORD: 10*frames per sec, 2nd DWORD: KBPS, 3rd DWORD: grab frame fps*10, 4th lost frame fps*10 due to overload, 5th DWORD skipped frame fps*10 due to configured enc interval in profile or STREAM_PRIORITY	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.704
CONF_VIDEO_H264_ENC_BASE_OPERATION_MODE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad3	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the base operation mode of the h.264 encoders per line. First DWORD stream 1, second DWORD stream 2...	
Write	p_octet	service	Set up the base operation mode of the h.264 encoders per line. First DWORD stream 1, second DWORD stream 2...	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

Stream Operation Mode [0] 4 Bytes
...
Stream Operation Mode [N] 4 Bytes

Stream Operation Mode

Copy other stream	0
compatibility mode to H2.64 BP+ (bitrate limited)	1
H.264 MP SD	3
H.264 MP 720p	4
H.264 MP 720p (full frame rate)	5
H.264 MP 1080p	6

Note: Only a few legacy IDs are listed above. Query the full set of supported Stream Operation Modes and their meaning for a specifc video input format of a camera via CONF_H264_ENC_BASE_OP_MODE_CAPS_VERBOSE (IDs are not guaranteed to have the same meaning across different video input modes and camera models in the future). Bit 24 signals that width and height need to be exchanged (picture is rotated by 90 degrees).

© Copyright 2001-2021 Bosch Security Systems

2.705 CONF_VIDEO_H264_ENC_CONFIG

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad2	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the profile for each h.264 encoder per line for each time. DWORD 1: Mode, 0=live mode, 1-10: schedule number; DWORD 2: profile for first stream, DWORD 3: profile for second stream...	
Write	p_octet	service	Set the profile for each h.264 encoder per line for each time. DWORD 1: Mode, 0=live mode, 1-10: schedule number; DWORD 2: profile for first stream, DWORD 3: profile for second stream... (if recording on that cam isn't running, the setting for live mode(0) will take place immediately); Notice: the number of expected streams can be queried with CONF_NBR_OF_ENC_STREAMS	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.706 CONF_VIDEO_H264_ENC_CONFIG_BULK

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad9	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Get the profiles for each h.264 encoder per line for each time; this request will supply all Modes 0-10 in one reply; returns: DWORD 1: NbrOfStreamsPerMode, remaining payload 11x payload of VIDEO_H264_ENC_CONFIG	
Write	p_octet	service	Set the profiles for each h.264 encoder per line for each time; this request must supply all Modes 0-10 in one request; DWORD 1: NbrOfStreamsPerMode, remaining payload 11x payload of VIDEO_H264_ENC_CONFIG; Notice: the expected 'NbrOfStreamsPerMode' can be queried with CONF_NBR_OF_ENC_STREAMS	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.707 CONF_VIDEO_H264_ENC_CONFIG_DEFAULTS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b4d	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_dword	service	Set the default profile for each h.264 encoder per line for each time. Payload: Mode, 0=live mode, 1-10: recording schedule number. (if recording on that cam isn't running, the setting for live mode(0) will take place imidiatly)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.708 CONF_VIDEO_H264_ENC_CURRENT_PROFILE

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad4	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the current active profile number for the h.264 streams (first DWORD: stream 1, second DWORD: stream 2...)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.709 CONF_VIDEO_INPUT_DESCRIPTION

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c78	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Byte[0] 0: optical video, 1: thermal video; reserved Bytes[1-15]	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.710 CONF_VIDEO_INPUT_FORMAT

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0504	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Obsolete to be removed	
Write	t_dword	service	Obsolete to be removed	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.711 CONF_VIDEO_INPUT_FORMAT_EX

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b10	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

16			32
Video Format Mode 1 Byte	**Video Format Id** 1 Byte	**Video Format Image Orientation** 1 Byte	**Reserved** 1 Byte
Rotation Low Byte 1 Byte	**Rotation High Byte** 1 Byte	**Rotation Mode** 1 Byte	**Reserved** 1 Byte
Reserved 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte
Reserved 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte	**Reserved** 1 Byte
8		24	

Video Format Mode

Use CONF_VIDEO_INPUT_FORMAT_EX_VERBOSE to query valid options:

Fixed	0	
Autodetect	1	Signals during read if device is in auto state -> 1 device is in auto mode; active id will be shown in Video Format Id

1 = Autodetect can be set if CONF_VIDEO_INPUT_FORMAT_EX_VERBOSE includes id = 0

Video Format Id

Valid ids can be queried with CONF_VIDEO_INPUT_FORMAT_EX_VERBOSE

Video Format Image Orientation

rotate 0° no mirror	0
rotate 0° mirror	1
rotate 90° no mirror	2

© Copyright 2001-2021 Bosch Security Systems

rotate 90° mirror	3
rotate 180° no mirror	4
rotate 180° mirror	5
rotate 270° no mirror	6
rotate 270° mirror	7

Rotation Low Byte

Low byte of 2 bytes rotation angle 0 -> 0xFFFF

Rotation High Byte

High byte of 2 bytes rotation angle 0 -> 0xFFFF

Rotation Mode

Standard	0
Crop	1
Zoom	2

© Copyright 2001-2021 Bosch Security Systems

2.712 CONF_VIDEO_INPUT_FORMAT_EX_OPTIONS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b9b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Obsolete to be removed	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.713 CONF_VIDEO_INPUT_FORMAT_EX_VERBOSE

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bfb	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

Tag

Tag specifying the encoding and meaning of the value

Tag 0: Basic verbose of CONF_VIDEO_INPUT_FORMAT_EX

(must always be present)

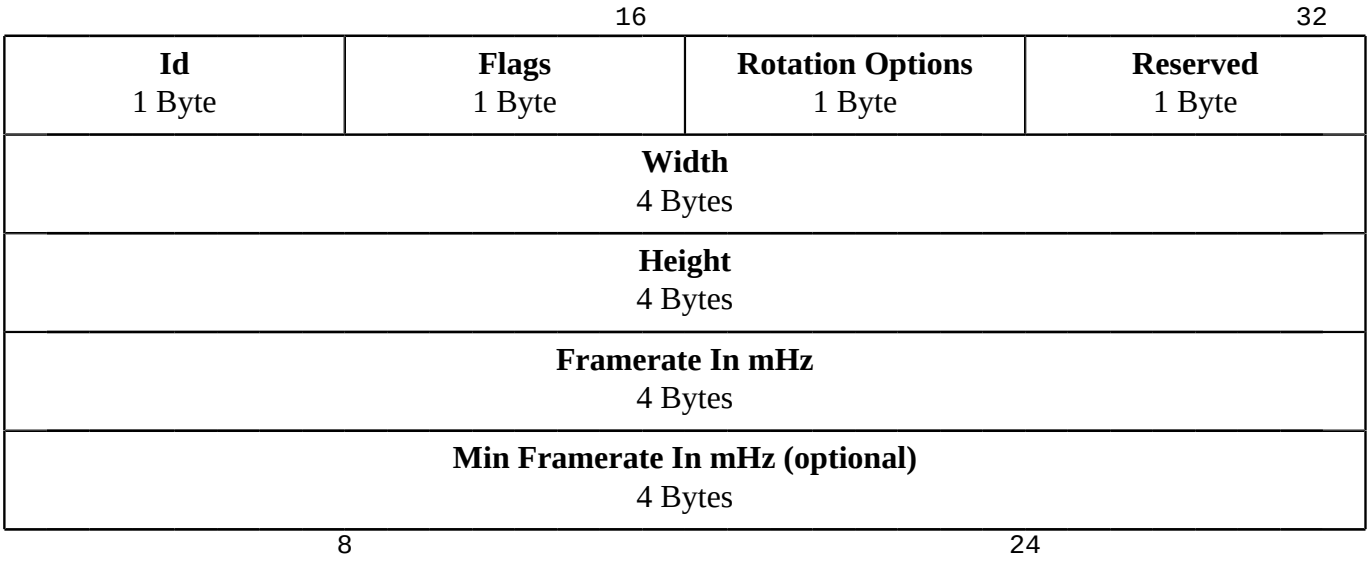
16 32

Length 2 Bytes	**Tag = 0x0000** 2 Bytes
Basic verbose of CONF_VIDEO_INPUT_FORMAT_EX (p_octet) *Length* - 4 Bytes	

8 24

© Copyright 2001-2021 Bosch Security Systems

Basic verbose of CONF_VIDEO_INPUT_FORMAT_EX



Id

Auto Mode Available	0	special id to signal availability of auto mode
ID	1 - 255	All available ids can be set via CONF_VIDEO_INPUT_FORMAT_EX defines (Video Format Id Byte [2])

Flags

	Mask	Name	Description
Bit 5	0x20	HDR-X Mode	Defines if mode is a HDR-X mode
Bit 4	0x10	CustomRotationSupport	Defines if custom rotation is supported (rotation low and heigh byte in CONF_VIDEO_INPUT_FORMAT_EX)
Bit 0	0x01	Cropped	If the crop is set, the view will be changed when switching between different resolutions takes place.

Combined Values

Mask	Name	Description
0x0E	nr of HDR exposure	nr of HDR exposure-1 0-> 1 exposure,1-> 2 exposure ...

Rotation Options

	Mask	Name
Bit 7	0x80	rotate 270° mirror
Bit 6	0x40	rotate 270° no mirror
Bit 5	0x20	rotate 180° mirror

© Copyright 2001-2021 Bosch Security Systems

Bit 4	0x10	rotate 180° no mirror
Bit 3	0x08	rotate 90° mirror
Bit 2	0x04	rotate 90° no mirror
Bit 1	0x02	rotate 0° mirror
Bit 0	0x01	rotate 0° no mirror

Width

Maximum available picture width for encoding

Height

Maximum available picture height for encoding

Framerate In mHz

Maximum available framerate for encoding

Min Framerate In mHz (optional)

Minimum available framerate for encoding

Example

```
Rotation Options = 0x44
 -> bit 2 and bit 6 are available (of bit 0...7)
 -> rotate 90° no mirror = 2 and rotate 270° no mirror = 6 are
 supported
```

© Copyright 2001-2021 Bosch Security Systems

2.714 CONF_VIDEO_LINE_DEWARPING_MODE

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf6	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Gets the dewarping mode of a virtual line. octet[0] is the selected id. If the request is applied to a physical line, the value 0 is returned. (Do not check the length of the command in the future may additional tags will be added). If only the first Byte is set for addition parameters defaults will be used. See CONF_VIDEO_LINE_DEWARPING_MODE_ for possible values.	
Write	p_octet	minimal	Sets the dewarping mode of a virtual line octet[0] sets the selected id. (Do not check the length of the command in the future may additional tags will be added). If only the first Byte is set for addition parameters defaults will be used. See CONF_VIDEO_LINE_DEWARPING_MODE_ for possible values.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.715
CONF_VIDEO_LINE_DEWARPING_MODE_CAPS

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns a list of possible dewarping modes of a virtual line. Each mode is a 8 bit value: 0: no dewarping, 1: virtual PTZ, 2: quad view, 3: panoramic, 4: double panoramic, 5: corridor, 6: full panoramic	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.716 CONF_VIDEO_LINE_DUO

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ca9	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Duo line support (0=off, 1=on), use CONF_DEVICE_CAPABILITIES tag 41 to detect support	
Write	t_dword	service	Duo line support (0=off, 1=on), use CONF_DEVICE_CAPABILITIES tag 41 to detect support	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		yes	

© Copyright 2001-2021 Bosch Security Systems

2.717 CONF_VIDEO_OUT_CROPPING

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x070c	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Enable cropping at video output if the aspect ratio of the incoming video doesn't fit	
Write	t_octet	service	Enable cropping at video output if the aspect ratio of the incoming video doesn't fit	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.718 CONF_VIDEO_OUT_DISPLAY_FORMAT

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x070a	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	(analog) display format:	
Write	t_word	service	(analog) display format.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Read Payload Structure

8	16
flags 1 Byte	**flags2** 1 Byte
4	12

flags

AUTO/ ANAMORPH	0
4:3	1
ANAMORPH 16:9	2

flags2

crop all other	2
reserved (set to 0)	

Write Payload Structure

8	16
flags 1 Byte	**flags2** 1 Byte
4	12

© Copyright 2001-2021 Bosch Security Systems

flags

AUTO/ ANAMORPH	0
4:3	1
ANAMORPH 16:9	2

flags2

crop	2

© Copyright 2001-2021 Bosch Security Systems

2.719 CONF_VIDEO_OUT_MONITOR_SPEC

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0708	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read active monitor specification of a video out line	
Write	p_octet	service	Override default monitor timing spec	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

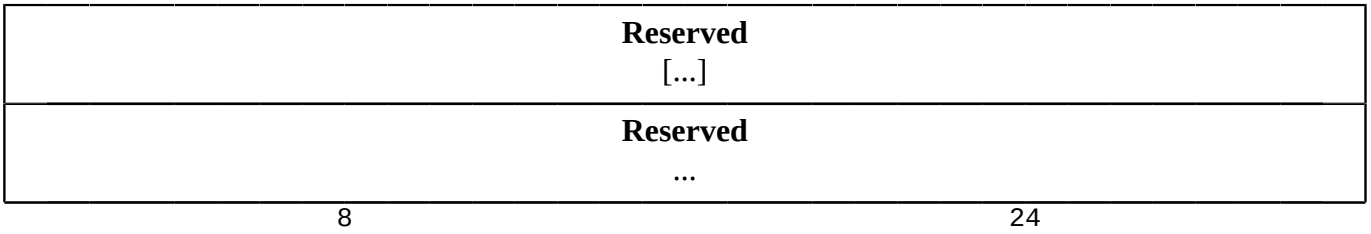
Payload Structure

16	32
width_mm 2 Bytes	**height_mm** 2 Bytes
native_width 2 Bytes	**native_height** 2 Bytes
hSyncMin 4 Bytes	
hSyncMax 4 Bytes	
vSyncMin 4 Bytes	
vSyncMax 4 Bytes	
dotClockMax 4 Bytes	

Reserved 20 Bits	**Sync Flags** 4 Bits	**compSync** 4 Bits	**sepSync** 4 Bits

Reserved... 32 Bytes

© Copyright 2001-2021 Bosch Security Systems



width_mm

Physical width of the actual display region in mm

height_mm

Physical height of the actual display region in mm

native_width

(maximum) width of the active display in pixel

native_height

(maximum) height of the active display in pixel

hSyncMin

Lowest supported scan line (hsync) frequency in Hz

hSyncMax

Highest supported scan line (hsync) frequency in Hz

vSyncMin

Lowest supported vertical refresh frequency in mHz

vSyncMax

Highest supported vertical refresh frequency in mHz

dotClockMax

Upper limit on the pixel clock

Sync Flags

	Mask	Name	Description
Bit 0	0x1	SyncOnGreen	1: Supported, 0: Not supported

compSync

Supported composite sync signals:

© Copyright 2001-2021 Bosch Security Systems

	Mask	Name	Description
Bit 3	0x8	H_Low_V_Low	h-sync active low, v-sync active low
Bit 2	0x4	H_High_V_Low	h-sync active high, v-sync active low
Bit 1	0x2	H_Low_V_High	h-sync active low, v-sync active high
Bit 0	0x1	H_High_V_High	h-sync active high, v-sync active high

sepSync

Supported separate sync signals:

	Mask	Name	Description
Bit 3	0x8	H_Low_V_Low	h-sync active low, v-sync active low
Bit 2	0x4	H_High_V_Low	h-sync active high, v-sync active low
Bit 1	0x2	H_Low_V_High	h-sync active low, v-sync active high
Bit 0	0x1	H_High_V_High	h-sync active high, v-sync active high

Note: In addition to the specified limits, default modes corresponding to the respective physical connector (e.g. standard VGA timing on a VGA connector, PAL/NTSC on BNC) are typically also assumed to be supported by the display device. The device will only allow video output timings (see CONF_VIDEO_OUT_CURRENT_SPEC, CONF_VIDEO_OUT_STANDARD) within these limits and may use the mentioned default timings as fallback values.

© Copyright 2001-2021 Bosch Security Systems

2.720 CONF_VIDEO_OUT_STANDARD

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0700	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Select video output standard: 0=AUTO; 1=PAL; 2=NTSC; 3, 4, 5, 6, 7, 8, 9, 10=VGA modes; 11=720p50; 12=720p60; 13=1080i50; 14=1080i60; 15=1080p25; 16=1080p30; 0xff=OFF	
Write	t_octet	service	Select video output standard: 0=AUTO; 1=PAL; 2=NTSC; 3, 4, 5, 6, 7, 8, 9, 10=VGA modes; 11=720p50; 12=720p60; 13=1080i50; 14=1080i60; 15=1080p25; 16=1080p30; 0xff=OFF	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.721 CONF_VIDEO_OUT_STANDARD_FORCE

[API: decoder/videoout]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x070b	output line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Force video output standard (e.g. for HDMI)	
Write	t_octet	service	Force video output standard (e.g. for HDMI)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

© Copyright 2001-2021 Bosch Security Systems

2.722 CONF_VIDEO_OVER_SSL

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b65	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	user	Returns if video over ssl is supported or not	
Write	t_dword	user	Enables/disables video over ssl	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

disabled	0
enabled	1

© Copyright 2001-2021 Bosch Security Systems

2.723 CONF_VIDEO_QUALITY

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a82	profile preset	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Gets the video quality for a selected preset (0=auto, 1(worst)..100(best))	
Write	t_dword	service	Sets the video quality for a selected preset (0=auto, 1(worst)..100(best))	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.724 CONF_VIDEO_RECOVERY

[API: recording settings]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0d04	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Recover video from a damaged span, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	no		no	

Description

This command is used to recover video data from VDP packets on a damaged span and convert it to an MP4 file. During recovery, all data but the VDP packets are ignored. Client ID, session ID, and numeric descriptor ID in the RCP header are not used. The payload contains all relevant information as a tag structure. This command is only supported on Windows.

Payload Structure

Tag-Structure [0]
...
Tag-Structure [N]

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

Length

Length of tagged value including length and tag field

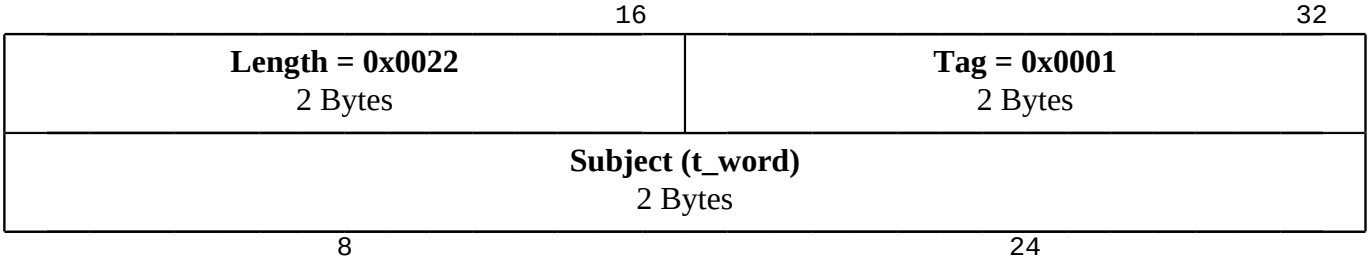
Tag

Tag specifying the encoding and meaning of the value

© Copyright 2001-2021 Bosch Security Systems

Tag 1: Subject

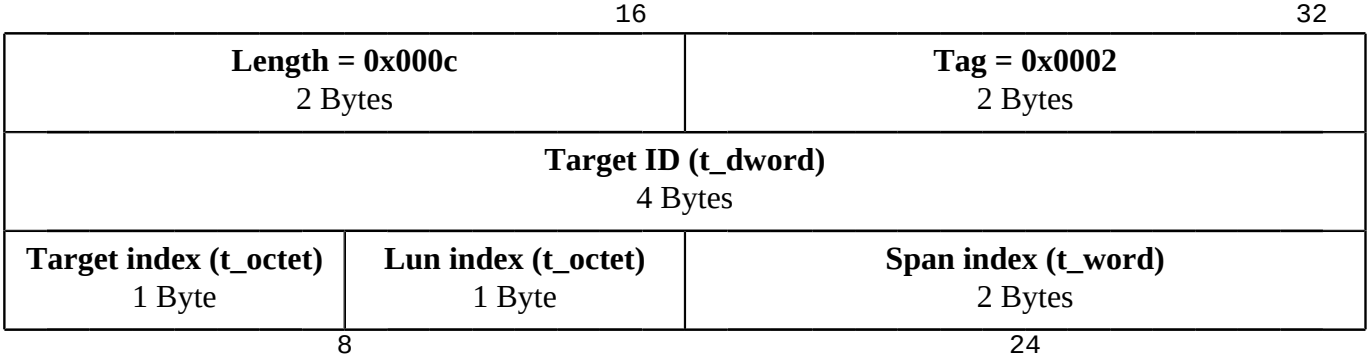
Request or event for which this command is sent.



Start	1	Sent in a request to start video recovery. The payload must contain the 'source' tag and can optionally contain the 'destination' tag. If no destination tag is sent, a default filename is used. The response will contain the 'started' subject and the 'thread ID' tag.
Started	2	Sent in a response to the 'start' subject to indicate that video recovery has been started. The payload contains the 'thread ID' tag.
Progress	3	Sent in a message to indicate the progress of video recovery. The payload contains the 'thread ID' and 'percentage' tags.
Finished	4	Sent in a message to indicate that video recovery has finished. The payload also contains the 'thread ID' tag.
Stop	5	Sent in a request to stop a video recovery thread. The payload must contain the 'thread ID' tag. The response will contain the 'stopped' subject and the 'thread ID' tag.
Stopped	6	Sent in a response to the 'stop' subject or in a message to indicate that the video recovery thread has been stopped. The payload contains the 'thread ID' tag.

Tag 2: Source

Source span from which video data is recovered. Sent with the 'start' subject.



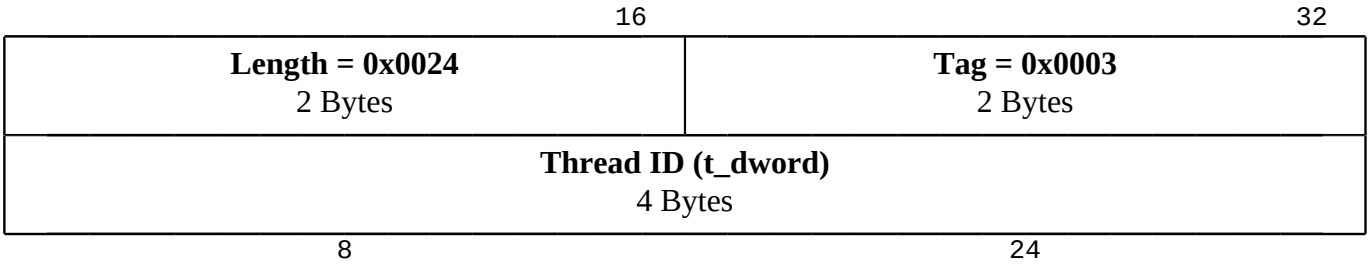
© Copyright 2001-2021 Bosch Security Systems

Target ID

IPv4 address of the target

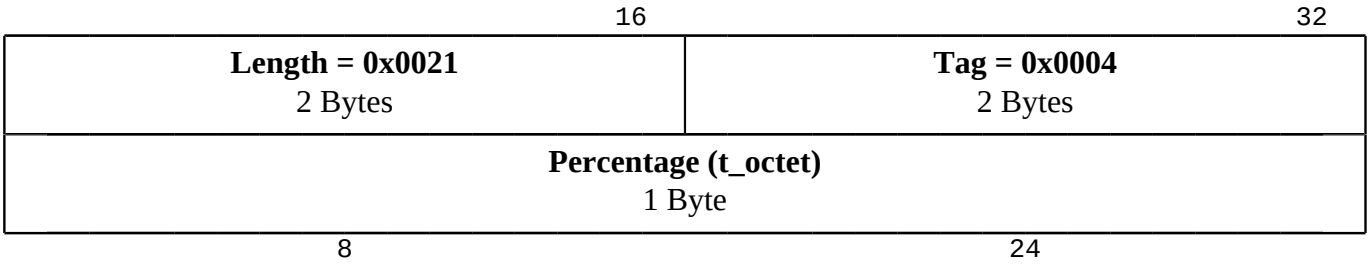
Tag 3: Thread ID

Unique ID of the video recovery thread. Sent with the 'started', 'progress', 'finished', 'stop', and 'stopped' subject.



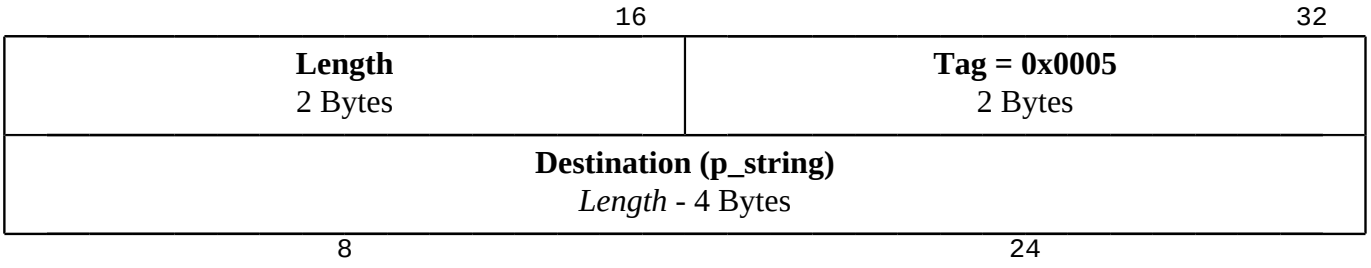
Tag 4: Percentage

Percentage ratio of storage space on the span where VDP packets could potentially exist which has already been analyzed. Sent with the 'progress' subject.



Tag 5: Destination

Destination filename to which video data is written. Can be sent with the 'start' subject.



© Copyright 2001-2021 Bosch Security Systems

2.725 CONF_VIDEO_STATIC_SCENE_REGIONS

[API: encoder]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0623	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	p_octet	service		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Description

The actual payload of this command consist of a sequence of variable length records, each specifying a region and a corresponding category to be used during encoding. The region is characterized by a mandatory rectangle and optionally a shape. Up to 8 regions and in total up to 96 32-bit data for shapes are supported. Depending on the firmware version the shape data may be ignored. In this case the rectangle is used instead. This may also happen, if the restrictions indicated below are violated by the shape data.

Normally the "background" category is used for unimportant parts of the scene and the "object" category for important ones. For this two categories, specific quality adjustments can be defined via the encoding profile of every encoder. An area not belonging to any region is implicitly assigned to the "Default" category with no specific adjustments. The forth category is currently assigned via the VIDEO_ENC_TEMP_QUANT_ADJ command separately for each encoder.

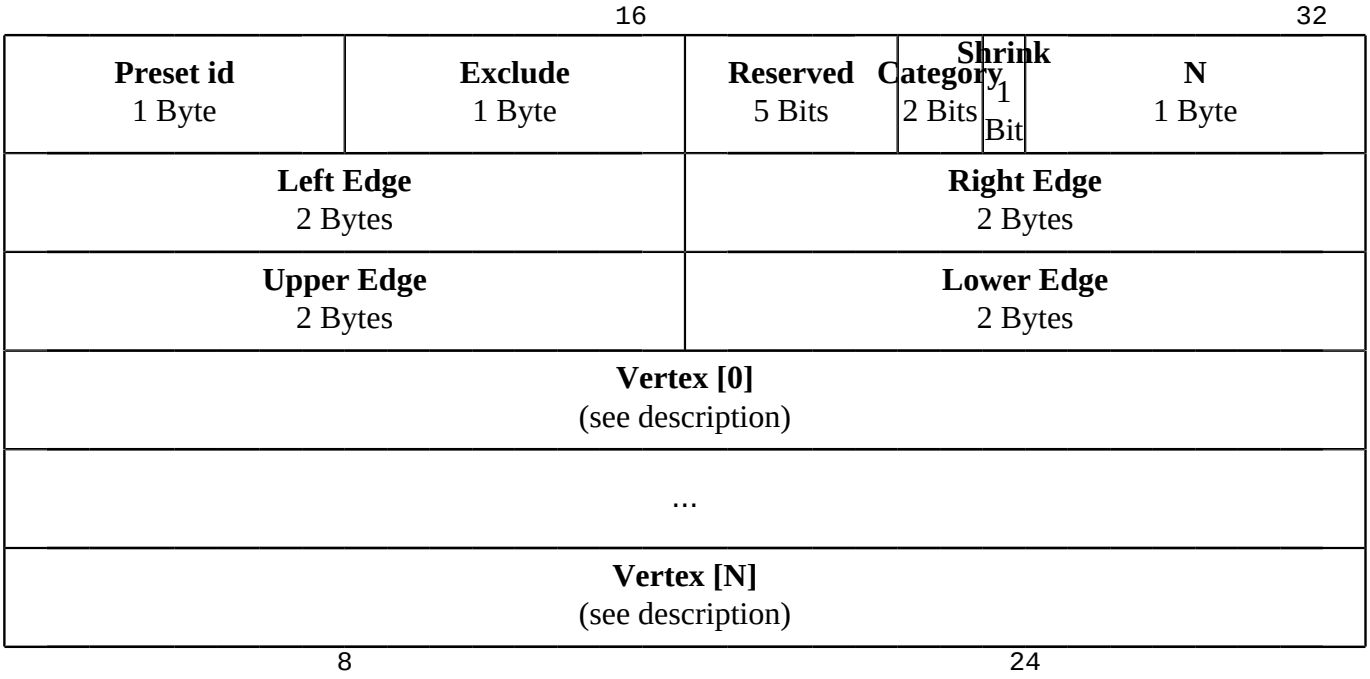
Note, that by using overlapping regions, more than one category can be assigned to some areas of the scene. In this case the category with the best quality adjustment for an encoder takes preceedence. Therefore it may be usefull to assign the "Default" category explicitly to some regions.

Payload Structure

Region Specification [0] (see description)
...
Region Specification [N] (see description)

© Copyright 2001-2021 Bosch Security Systems

Region Specification



Preset id

Only valid for a Dome

Exclude

This bitmask indicates for which encoders this area shall NOT be used. The bits are allocated according to the the relative coder number starting with the rightmost bit (LSB) for the relative coder number 1.

	Mask	Name
Bit 1	0x02	Coder 2
Bit 0	0x01	Coder 1

Category

By default the complete scene is assumed to belong to the 'default' category.

Default	0
Background	1
Objects	2
Reserved	3

Shrink

Depending on the encoding resolution, the quality may only be adjusted on quite a coarse grid. This bit allows to specifiy whether the required rounding shall be performed by shrinking or enlarging the specified region.

© Copyright 2001-2021 Bosch Security Systems

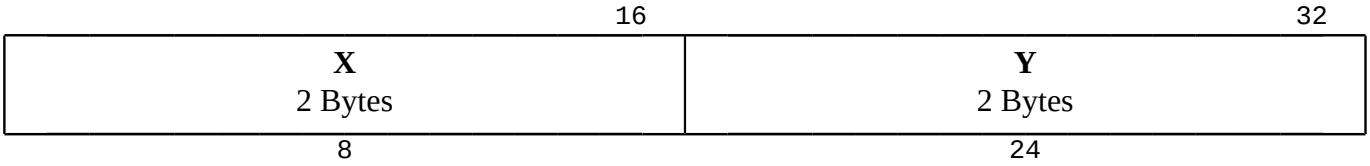
Enlarge as needed	0
Shrink as needed	1

N

This field indicates the number of 32-bit items defining the optional shape of this region and therefore also specifies indirectly the total size of this record (12 + N * 4 octets). For compatibility with future version of this RCP PLUS specification, clients SHOULD set this item to 0 for any region which they define or modify, but keep it and the corresponding shape data unmodified for regions they don't change.

Vertex

32-bit item used for shape data. The item describes a vertex of a polygonal shape boundary in the order as included in the record (the closing edge between vertex N and vertex 1 is added automatically). The vertices should all be different and the edges between them should have no point in common beside the vertex between them. (If the resulting area inside the polygon is not single connected the behaviour is undefined)



X

X Coordinate	0 - 32678

Y

Y Coordinate	0 - 32678

The rectangles and the future shapes are specified via a virtual coordinate system with a nominal range of 0..32768 x 0..32768 for the usable sensor area (or incoming video). All rectangles and shapes should be completely inside this area, otherwise, the exact behaviour is undefined.

© Copyright 2001-2021 Bosch Security Systems

2.726 CONF_VIDEO_TERMINATION_RESISTOR_OFF

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0275	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	State of the analog video termination resistor (75 Ohms on/ off)	
Write	f_flag	service	Switch the analog video termination resistor (75 Ohms on/off),	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

enabled	1	75 Ohms on
disabled	0	75 Ohms off

© Copyright 2001-2021 Bosch Security Systems

2.727 CONF_VIDEO_TERMINATION_RESISTOR_ON

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0274	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	State of the analog video termination resistor (75 Ohms on/ off)	
Write	f_flag	service	Switch the analog video termination resistor (75 Ohms on/off),	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Values:

enabled	1	75 Ohms on
disabled	0	75 Ohms off

© Copyright 2001-2021 Bosch Security Systems

2.728 CONF_VIN_BASE_FRAMERATE

[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ad6	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Obsolete use CONF_VIDEO_INPUT_FORMAT_EX_VERB	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.729 CONF_VIPROC_ALARM

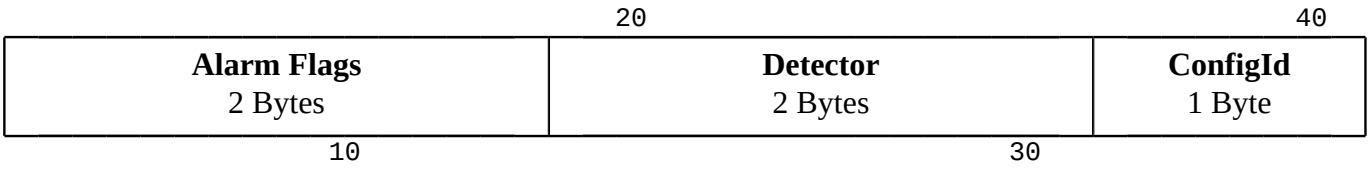
[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0807	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The message is sent whenever any of the bit values changes. Additionally it is sent once per 10 seconds when any of the bits is set.

Payload Structure



Alarm Flags

	Mask	Name
Bit 15	0x8000	default task
Bit 14	0x4000	global change flag
Bit 13	0x2000	signal too bright flag
Bit 12	0x1000	signal too dark flag
Bit 11	0x0800	reserved
Bit 10	0x0400	image too blurry flag
Bit 9	0x0200	signal loss flag
Bit 8	0x0100	reference image check failed flag
Bit 7	0x0080	invalid configuration flag
Bit 6	0x0040	flame detection flag
Bit 5	0x0020	smoke detection flag

Detector

State of the detectors. 0x0001 corresponds to detector one, 0x0002 to detector two and so on.

ConfigId

Identifcation number of the VCA profile caused this alarm.

© Copyright 2001-2021 Bosch Security Systems

2.730 CONF_VIPROC_ALARM_AGGREGATION_TIME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0aca	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Obsolete, please use VIPROC_ALARM_EXT_AGGREGATION_TI	
Write	t_octet	iva	Obsolete, please use VIPROC_ALARM_EXT_AGGREGATION_TI	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.731
CONF_VIPROC_ALARM_EXT_AGGREGATION_TIME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba7	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Obsolete, please use rule engine script. read the alarm aggregation time of the video line (unit is 0.1 seconds, thus, 16 corresponds to 1.6 seconds). The return value has to bytes is a word beginning with the upper byte and followed by the lower byte.	
Write	p_octet	iva	Obsolete, please use rule engine script. write the duration of the alarm aggregation time in 0.1 seconds (1.6 sec coresponds to a value 16). The payload consists of an upper and a lower byte resulting in a maximal time of 6553.6 seconds. Note, this parameter belongs to a viproc config.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.732 CONF_VIPROC_ALARM_MASK

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a23	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read which of the 32 bits produced by the viproc +ruleengine are considered for the CONF_VIPROC_ALARM. see CONF_VIPROC_ALARM for the meaning of the 32 bits.	
Write	p_octet	iva	Configures which of 32 bits produced by the viproc +ruleengine are considered for the CONF_VIPROC_ALARM see CONF_VIPROC_ALARM for the meaning of the 32 bits.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.733
CONF_VIPROC_CONFIG_CHANGE_IN_RECORDING

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a60	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.734 CONF_VIPROC_CONFIG_NAME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a40	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_unicode	minimal	Returns the name of the currently active viproc configuration.	
Write	p_unicode	iva	Writes the name of the currently active viproc configuration. The name can have at most 15 unicode characters.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.735 CONF_VIPROC_CUSTOM_PARAMETERS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0802	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads the video content analysis (VCA) custom parameters, payload is beyond the scope of this document	
Write	p_octet	iva	Sets the VCA custom parameters, payload is beyond the scope of this document	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.736 CONF_VIPROC_CUSTOM_PARAMETERS_TAGS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bac	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads one or more parameter tags from viproc configuration, payload is beyond the scope of this document	
Write	p_octet	iva	Sets one or more tags in the VCA custom parameters, payload is beyond the scope of this document	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.737 CONF_VIPROC_DEFAULT

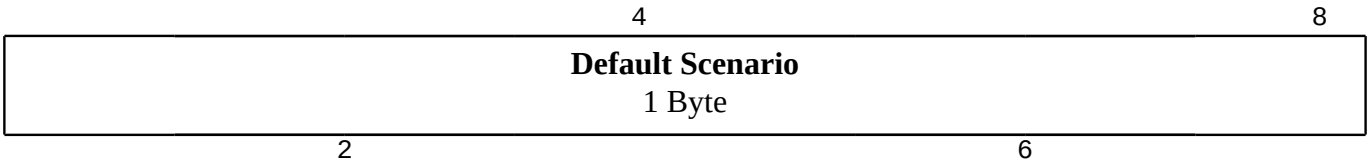
[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cbb	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	t_octet	iva		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This commands deploys a dedicated config file to set VCA config. The list of available scenario defaults can be retrieved with the command VIPROC_DEFAULT_LIST. In case of an error this command may return with a command specific error code. For a list of the specific error code see below. The value associated to scenario will be kept constant.

Payload Structure



Default Scenario

Default	0
Intrusion (one field)	1
Intrusion (two fields)	2
People counting	3
Traffic incidents	4
Traffic wrong way	5

Command Specific Errors

Default scenario not available	0x01
Calibration missing	0x02
Internal error	0xF0

© Copyright 2001-2021 Bosch Security Systems

2.738 CONF_VIPROC_DEFAULT_LIST

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cba	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	List of available IVA default configuration sets, value: string | ... eg, 0=default | 1=line_crsossing.iva | 2=inside_field.iva	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.739 CONF_VIPROC_DETECTOR_PARAMS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c96	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads VCA detector params	
Write	p_octet	iva	Sets VCA detector params	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.740 CONF_VIPROC_DLL_NAME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0804	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Reads the name of currently selected VCA component	
Write	p_string	iva	Specify the name of the VCA component	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.741 CONF_VIPROC_DLL_NAME_LIST

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0808	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	List of viproc component names	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.742 CONF_VIPROC_FIRE_PARAMETERS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c0c	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Gets the VCA fire detector settings (no vca profiles), payload is beyond the scope of this document	
Write	p_octet	iva	Sets the VCA fire detector settings (not part of the vca profiles), payload is beyond the scope of this document	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.743 CONF_VIPROC_GLOBAL_PARAMETERS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b68	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads the VCA global parameters, payload is beyond the scope of this document	
Write	p_octet	iva	Sets the VCA global parameters, payload is beyond the scope of this document	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.744 CONF_VIPROC_GLOBAL_PARAMETERS_TAGS

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c68	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Reads the VCA global parameters, payload is beyond the scope of this document	
Write	p_octet	iva	Sets the VCA global parameters, payload is beyond the scope of this document	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.745 CONF_VIPROC_HOLIDAYS_SCHEDULE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a67	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the holidays VCA schedule of the specified line. The format of the payload is the same as the one of the CONF_HD_RECORD_HOLIDAYS command.	
Write	p_octet	iva	Sets the holidays VCA schedule of the specified line. The format of the payload is the same as the one of the CONF_HD_RECORD_HOLIDAYS command.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.746 CONF_VIPROC_ID

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0803	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Get the viproc interface id (0:= no viproc), nb: message is sent in host byte order	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.747 CONF_VIPROC_MODE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a65	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Switches between silent vca(0)/ manual(1..16)/off(0xfd)/ scheduler(0xfe)/script mode(0xff) of the video content analysis (VCA). In case of a moving dome and a preset with activated VCA profile the active Id is returned. (if vca is turned off then 0xfd is returned).	
Write	t_octet	iva	Switches between silent vca(0)/ manual(1..16)/off(0xfd)/ scheduler(0xfe)/script mode(0xff) of the video content analysis (VCA). Command should not be used while configuring a profile (i.e. start_viproc_config_editing is active). On the gen4 dome, the modes 0xfe and 0xff are not supported. Any value between 0 and 16 is interpreted as on and 0xfd as off.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.748 CONF_VIPROC_MOTION_DEBOUNCE_TIME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0acb	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read the motion alarm debounce time of the video line (4, 3s == 43). Note, this parameter belongs to a viproc config.	
Write	t_octet	iva	Write the duration of the motion alarm debounce time of the video line (1, 6sec == 16). Note, this parameter belongs to a viproc config.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.749
CONF_VIPROC_OLD_AGGREGATION_TIME_USED

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf3	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Returns if old aggregation times are used or not	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.750 CONF_VIPROC_ONOFF

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0800	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	Reads the VCA status 0=off, 1=on. command is obsolete with firmware 4.0 and replaced by VIPROC_MODE.	
Write	f_flag	iva	Switches on/off the video content analysis (VCA). command is obsolete with firmware 4.0 and replaced by VIPROC_MODE.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.751 CONF_VIPROC_RE_TASK_NAMES

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b2b	video line	yes	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read the current rule engine task names which corresponds to line <num>	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command reads out the task names of the current configured rule engine.

Response Payload Structure

16	32
Algorithm type 2 Bytes	**Number of tasks** 2 Bytes
Task [0] (see description)	
...	
Task [N] (see description)	

8 24

Algorithm type

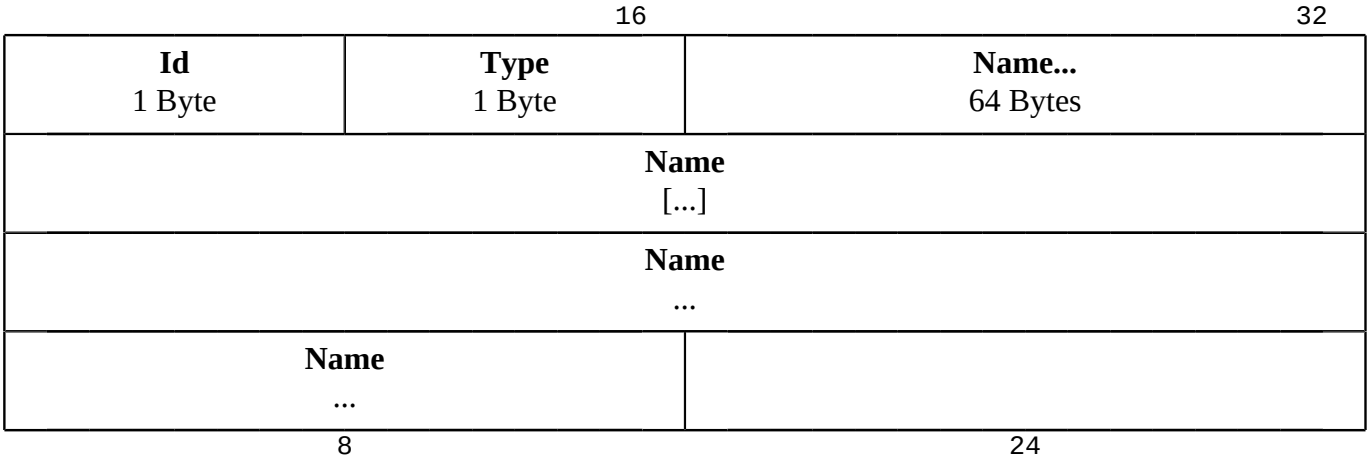
IVA algorithm	1
IVA flow algorithm	2
Motion algorithm	4

Number of tasks

Number of configured user tasks.

© Copyright 2001-2021 Bosch Security Systems

Task



Id

Id of the user task. Defines the alarm bit of the CONF_VIPROC_ALARM message.

Type

Object in field	1
Line crossing	2
Loitering	3
ConditionChange	5
FollowingRoute	6
Tampering	7
RemovedObject	8
IdleObject	9
EnteringField	10
LeavingField	11
SimilaritySearch	13
CrowdDetection	14
Counter	15
BEV people counter	16
Line Counter	17
Occupancy	18
Man over Board	19
FlowInField	40
CounterFlowInField	41
MotionInField	42

Name

The task name is UTF8 encoded and maximal 64 byte. Zero termination is not mandatory.

© Copyright 2001-2021 Bosch Security Systems

2.752 CONF_VIPROC_REF_ORIENTATION

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf2	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read tamper reference orientation	
Write	p_octet	iva	Write tamper reference orientation	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.753
CONF_VIPROC_REFERENCE_IMAGE_FILENAME

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0806	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string p_octet	minimal	Reads the current reference image filename	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.754 CONF_VIPROC_SAVE_REFERENCE_IMAGE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0805	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	f_flag	iva	Triggers the VCA to create a reference image, but only if possible (e.g. not a dome camera)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.755 CONF_VIPROC_SCENE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a3b	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	t_dword	minimal	Returns the scene for which the active config is defined.	
Write	t_dword	iva	Writes the scene number of the active viproc configuration	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.756 CONF_VIPROC_SENSITIVE_AREA

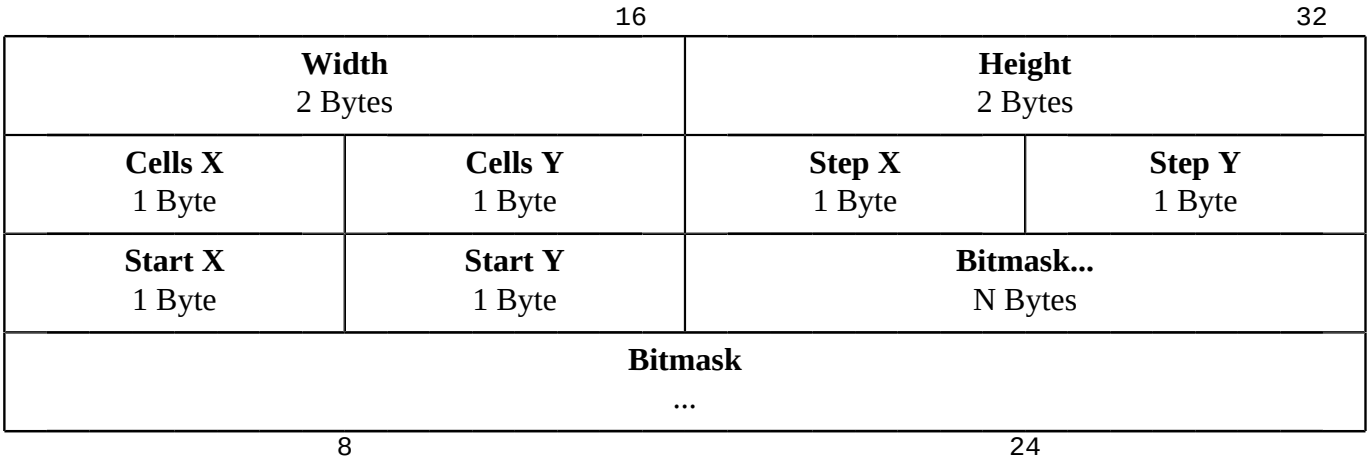
[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b78	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Sets the sensitive area of the object based VCA algorithm in the active (or last loaded) config profile, see detailed description	
Write	p_octet	iva	Sets the sensitive area of the object based VCA algorithm in the active (or last loaded) config profile	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command reads/sets the sensitivity mask of the viproc algorithm. Please use the read functionality to get the dimensions. All values are in pixel. If no viproc algorithm is running the command fails.

Payload Structure



Width

Image width of the viproc algorithm.

Height

Image height of the viproc algorithm.

Cells X

Number of cells in X direction.

© Copyright 2001-2021 Bosch Security Systems

Cells Y

Number of cells in Y direction.

Step X

Number of cell size in X direction.

Step Y

Number of cell size in Y direction.

Start X

Offset of bit mask from left edge.

Start Y

Offset of bit mask from upper edge.

Bitmask

Bitstream which signals which cell is activated or not. Upper left cell is the first going to the left and then down.

Example:

A 3x3 mask with cross activated:

oxo
xxo
oxo

has the following binary 0b010110010 or hexadecimal 0x590 structure. Alignment bits can be set to zero but will be ignored.

© Copyright 2001-2021 Bosch Security Systems

2.757 CONF_VIPROC_SET_REF_ORIENTATION

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0cf1	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	iva	Set current orientation of acceleration sensor as reference	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.758 CONF_VIPROC_TAGGED_CONFIG

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a42	yes	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read a list of viproc items in an atomar way. see detailed description	
Write	p_octet	iva	Writes any viproc commands in an atomar way. see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

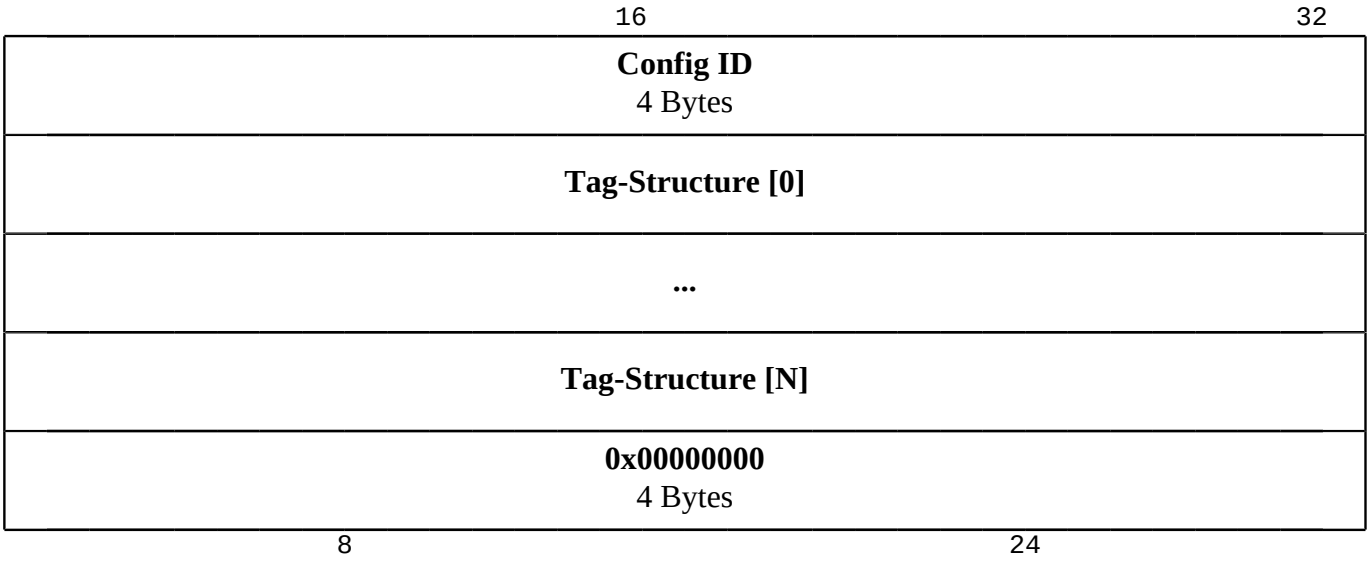
Description

This command is an extension of the already existing RCP+ commands CONF_VIPROC_DLL_NAME, CONF_VIPROC_ONOFF, CONF_VIPROC_SCENE, CONF_VIPROC_CONFIG_NAME, CONF_VIPROC_CUSTOM_PARAMETERS, CONF_VCD_OPERATOR_PARAMS, CONF_VIPROC_ALARM_MASK, CONF_VIPROC_ALARM_AGGREGATION_TIME, and CONF_VIPROC_MOTION_DEBOUNCE_TIME. Each of these commands can only access one specific item of the currently active viproc configuration. The new command allows an atomic access to an arbitrary subset of configuration items, not only of the currently active one but of any configuration of a line.

Num Descriptor Values

VideoLine	Any	The video line to which the configuration belongs

Payload Structure



© Copyright 2001-2021 Bosch Security Systems

Config ID

An integer between 1 and the maximum number of configurations available for this line (this maximum number can be retrieved by the command CONF_NUMBER_OF_VIPROC_CONFIGS).

Tag-Structure

Payload containing tagged values. Each tag structure consists of a length and a tag field defining the meaning of the following data.

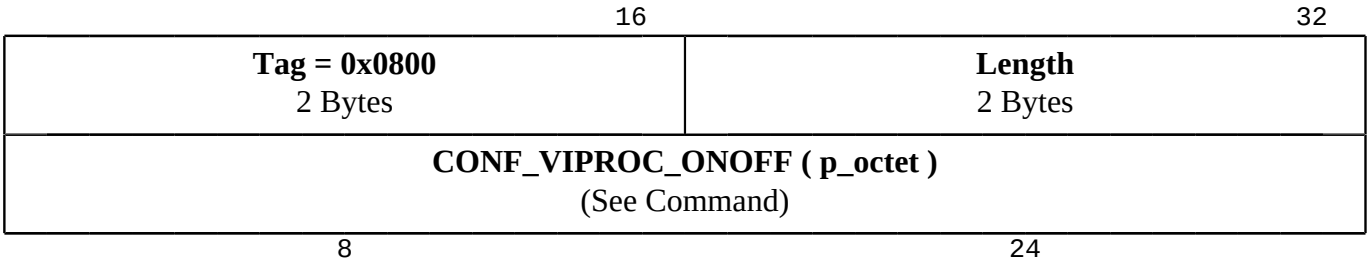
Length

Length of tagged value including length and tag field

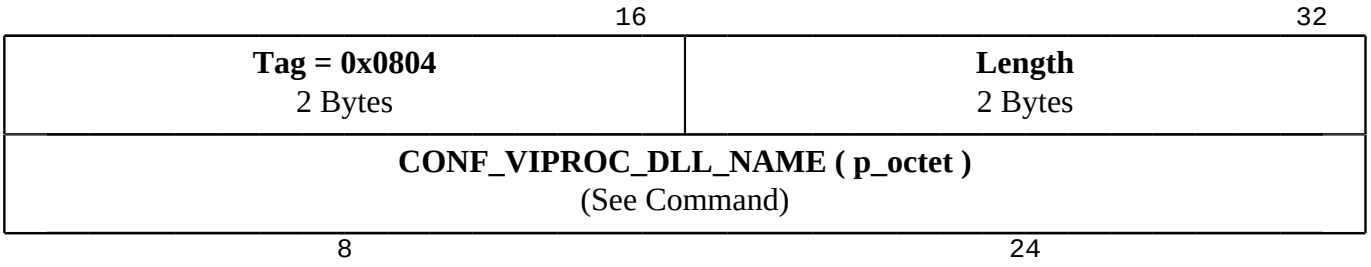
Tag

Tag specifying the encoding and meaning of the value

Tag 0x0800: CONF_VIPROC_ONOFF

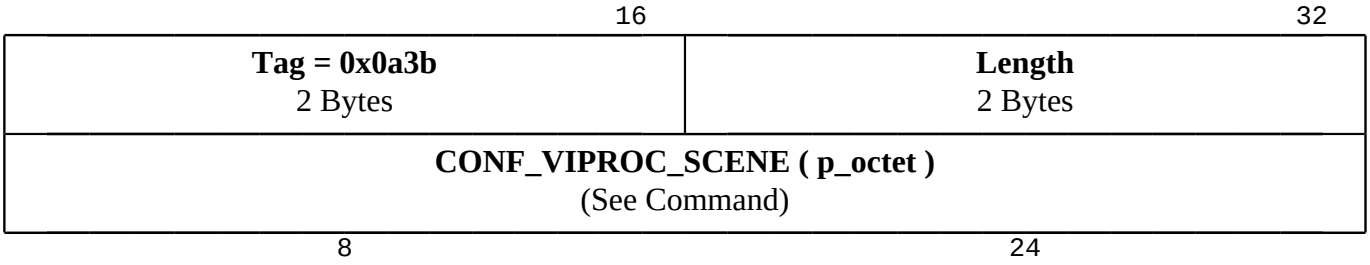


Tag 0x0804: CONF_VIPROC_DLL_NAME

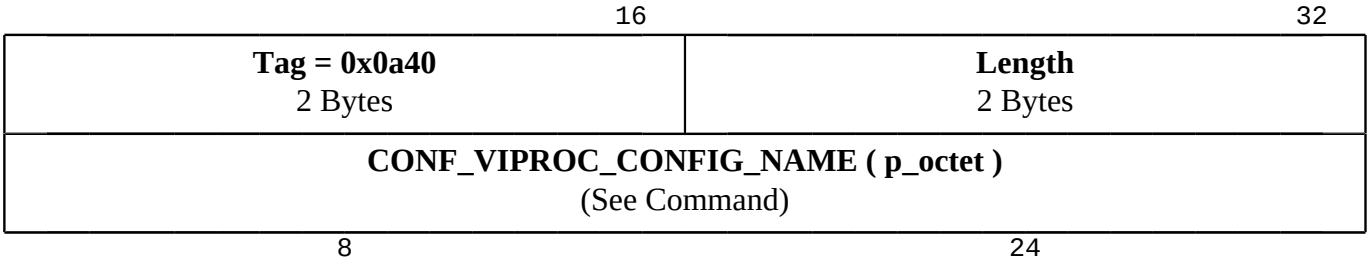


© Copyright 2001-2021 Bosch Security Systems

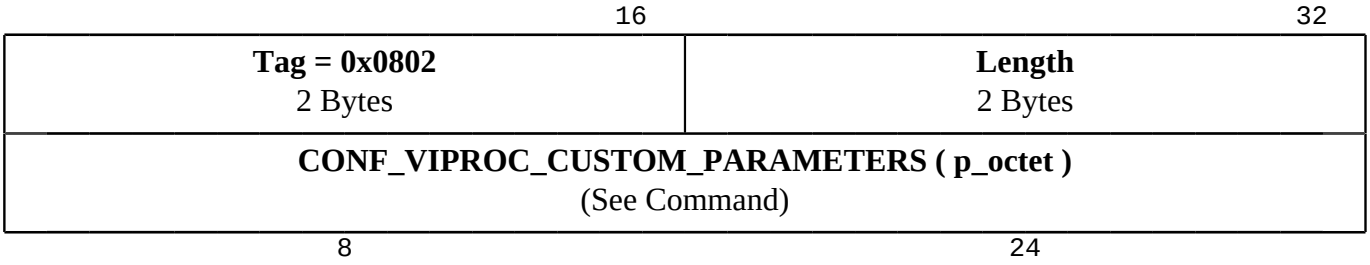
Tag 0x0a3b: CONF_VIPROC_SCENE



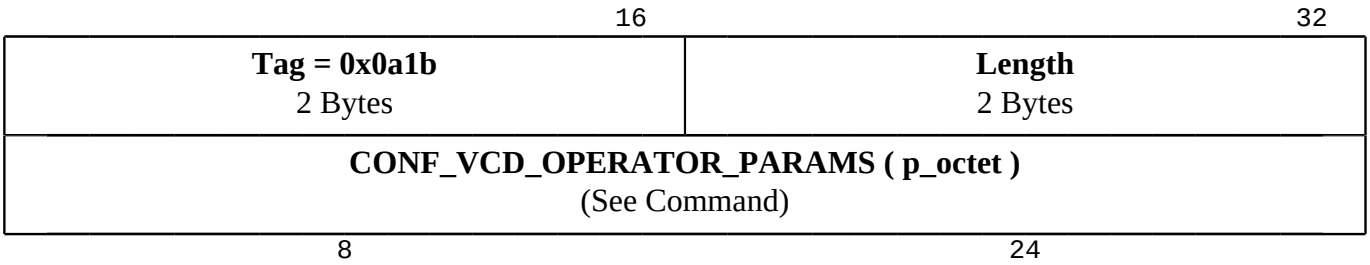
Tag 0x0a40: CONF_VIPROC_CONFIG_NAME



Tag 0x0802: CONF_VIPROC_CUSTOM_PARAMETERS

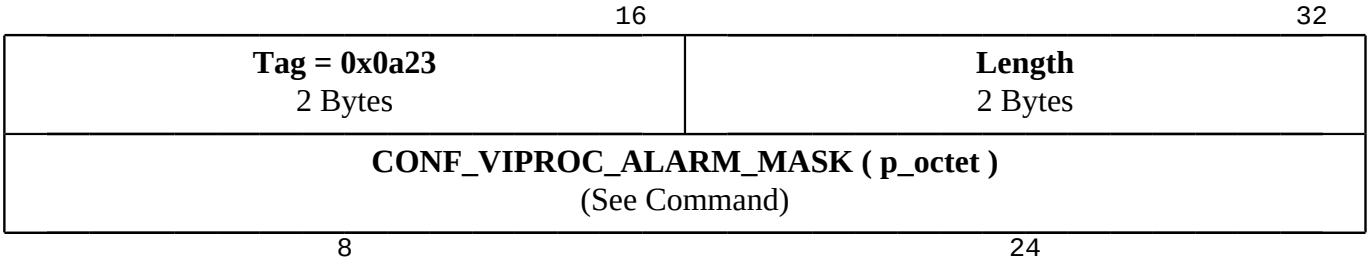


Tag 0x0a1b: CONF_VCD_OPERATOR_PARAMS

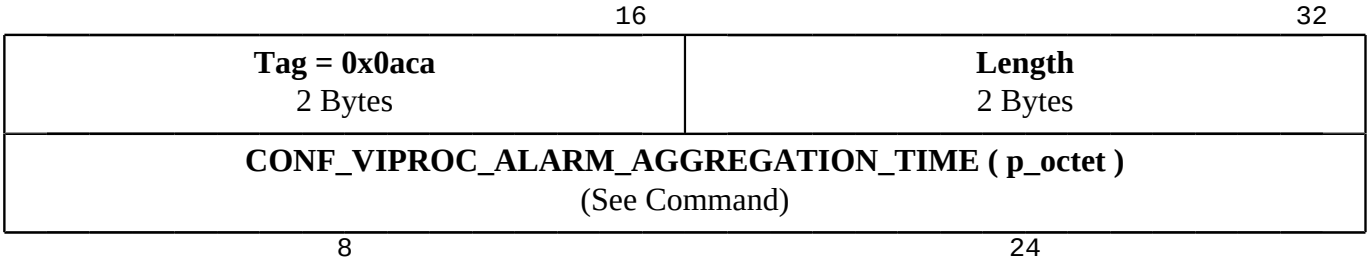


© Copyright 2001-2021 Bosch Security Systems

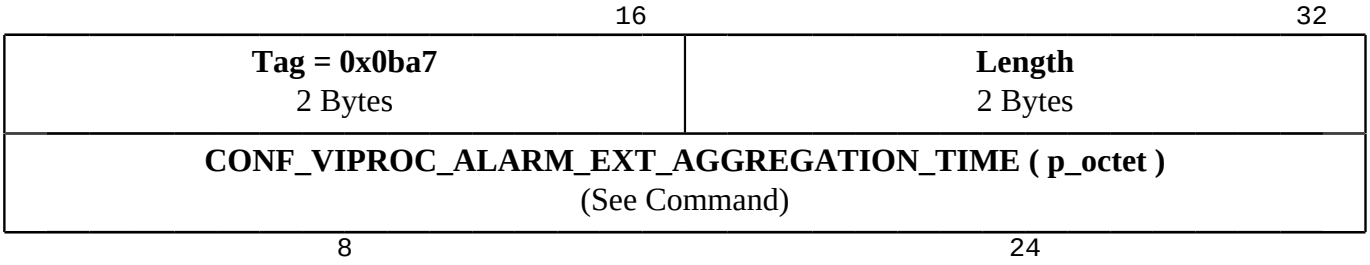
Tag 0x0a23: CONF_VIPROC_ALARM_MASK



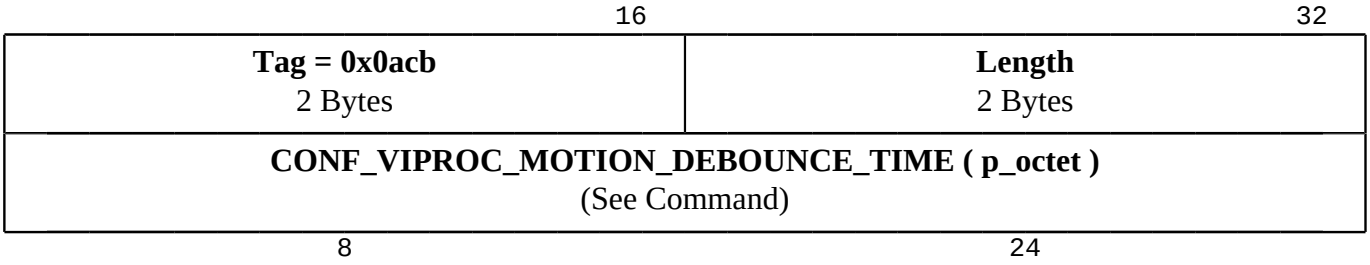
Tag 0x0aca: CONF_VIPROC_ALARM_AGGREGATION_TIME



Tag 0x0ba7: CONF_VIPROC_ALARM_EXT_AGGREGATION_TIME



Tag 0x0acb: CONF_VIPROC_MOTION_DEBOUNCE_TIME



© Copyright 2001-2021 Bosch Security Systems

2.759 CONF_VIPROC_TAGGED_CONFIG_INTERNAL

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a43	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Read a list of viproc items in an atomar way.	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.760 CONF_VIPROC_WEEKLY_SCHEDULE

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a66	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal	Returns the weekly VCA schedule of the specified line. The format of the payload is the same as the one of the CONF_HD_RECORD_SCHEDULE command.	
Write	p_octet	iva	Sets the weekly VCA schedule of the specified line. The format of the payload is the same as the one of the CONF_HD_RECORD_SCHEDULE command.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.761 CONF_VIRTUAL_ALARM_PARAMETER

[API: alarm]

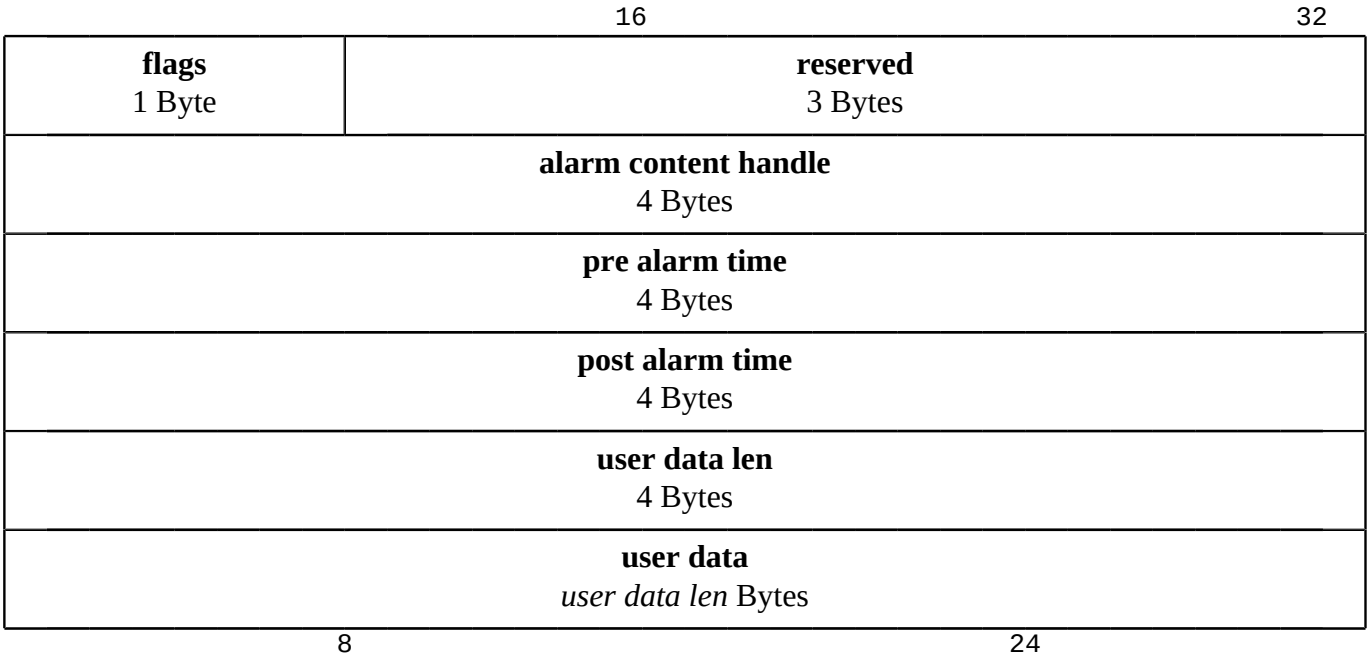
	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ba6	virtual alarm nbr	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Activate/deactivate virtual alarms with parameter, see detailed description	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

This command is used to trigger a virtual alarm along with parameters for alarm recording. The alarm can be activated, deactivated or triggered as single shot, which is an activate and deactivate within on command. Its also possible to trigger a silent alarm, by setting the activate and deactivate flag to zero. This works like a single shot alarm, the user data will be recorded and backups between the recordings will be scheduled, but the virtual alarm won't be triggered. If silent alarm is used, the num parameter doesn't choose the virtual alarm number, but it is mapped directly to the recording line. The command can have user data which can be recorded as meta data along to the video data. Also the pre and post alarm time can be specified but is only valid, if the alarm recording is done via backuped recording setup and the Flag ALARM_IGNORE_ALARM_TIME_PARAMETER isn't set. This will cause an alarm backup from the recording (primary or secondary, depending on the virtual alarm configuration and encoder index configuration) with this parameter, but the pre alarm time is limited by the max pre alarm time which is the same as the configured prealarm time of the secondary recording on the internal storage. It is also limited by the storage space of the internal storage. The max pre alarm time is also the delay for the recording. That means the recording of the alarm file triggered by an alarm will be started after this delay expires. Until that time the records of that file are only available on the internal storage. When a new alarm is triggered by activate, the reply returns an alarm content handle (only on activation, not on single shot). The limit for open content handle at the same time per line (includes also deactivated alarms or alarm shots within the post alarm time) is 32. This handle can be used to deactivate the alarm later by sending this command with the handle in the payload or it can be used to retrigger the alarm with new user data. For retrigger the playload looks the same as for activate, except a valid handle is required. Only for deactivation or retrigger of the alarm, the handle is required for in payload. A valid handle is a non zero handle, if the handle is set to zero, it will be interpreted as wild card on deactivate, which will deactivate/close all active handles or as invalid. In case of activate, a new handle will be opened and returned in the reply, if the handle is invalid/ zero. If a handle(or all handles) will be deactivated/closed, it will decay with the post alarm time. An alarm can also used on first activation to add the "protected" attribute to the file, which includes the record data of the alarm. The limit for pending intervals for protection are limited to 32 per line. Pending is an interval in the time from the occurence of the protecting alarm and the backup of the alarm record data.

Payload Structure

© Copyright 2001-2021 Bosch Security Systems



flags

	Mask	Name	Description
Bit 4	0x10	ALARM_USER_DATA_1X	user data will be recorded only once
Bit 3	0x08	ALARM_IGNORE_ALARM_TIME_PARAMETER	The pre and post alarm time has no meaning, parameter from the record profiles will be used instead
Bit 2	0x04	ALARM_FILE_PROTECTION	protects the file, which contains the alarm video data (only on activate or shot)
Bit 1	0x02	ALARM_DEACTIVATE	deactivate the alarm
Bit 0	0x01	ALARM_ACTIVATE	activate the alarm

Combined Values

Mask	Name	Description
0x03	ALARM_SINGLE_SHOT	activate and then deactivate the alarm (ALARM_ACTIVATE | ALARM_DEACTIVATE)

alarm content handle

Handle for the alarm(required for deactivation or retrigger, returned in reply on activation) non zero value or zero for wild card in case of deactivate

pre alarm time

Pre alarm time in seconds

post alarm time

Post alarm time in seconds

© Copyright 2001-2021 Bosch Security Systems

user data len

Length in bytes of the following user data, max 4000 bytes

user data

User data which will be recorded repeatedly, as long as the alarm is aktive or while within the post alarm time of the alarm

© Copyright 2001-2021 Bosch Security Systems

2.762 CONF_VIRTUAL_ALARM_STATE

[API: alarm]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0a8b	virtual alarm number	yes	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	minimal	0=virtual alarm off; 1=virtual alarm on	
Write	f_flag	user	0=virtual alarm off; 1=virtual alarm on	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.763 CONF_VIRTUAL_AUDIO_LINES

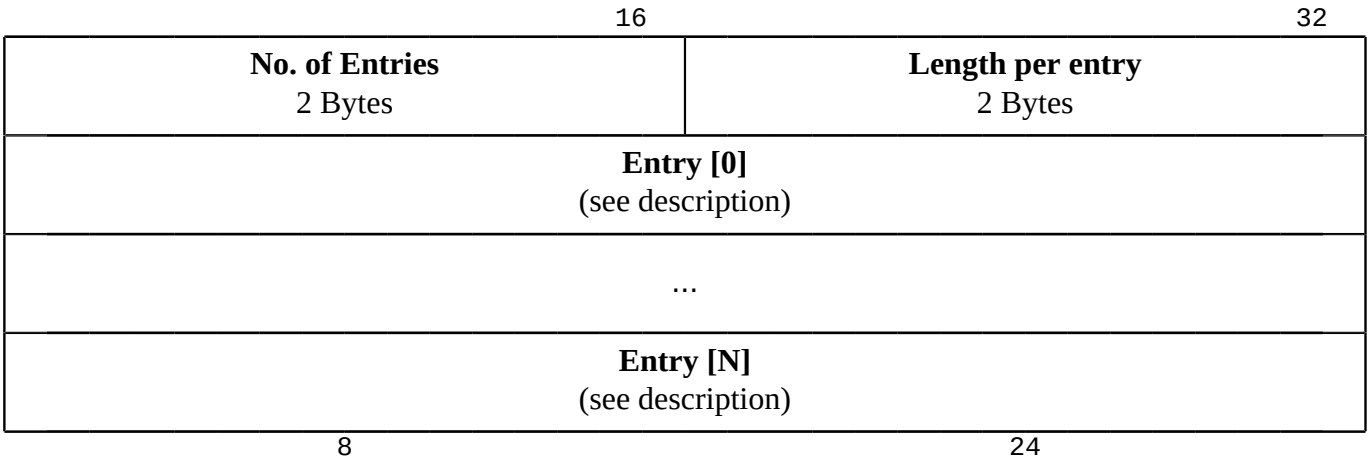
[API: audio]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf5	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command returns which audio input lines are virtual and which is the associated physical line. A physical line is phyical audio input - that's the usual case. A virtual line is just a copy of a physical line and supports only a subset of the RCP commands compared to a physical line. Some commands only make sense on physical line (e.g. input volume). Those commands fail on virtual lines and must be applied on the associated physcial line instead. The required information can be read out using this command. Virtual lines are introduced to provide a copy to an existing physical line. A single physical audio input can be associated to multiple video input lines. This keeps our known behaviour to have an audio line with the same number as the video line.

Payload Structure



No. of Entries

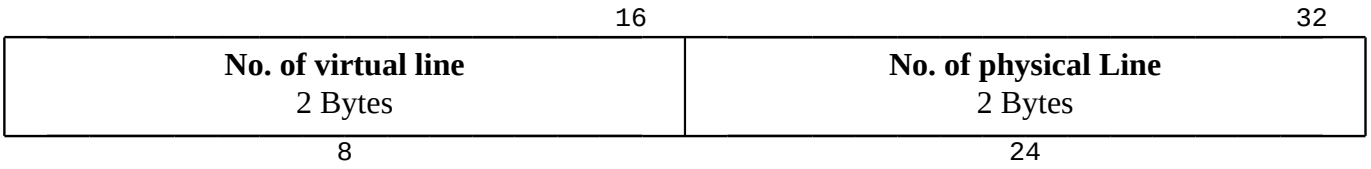
The number of virtual line descriptions.

Length per entry

The number of bytes per line description. This value is currently 4 but may be increased later when more information per entry must be provided. New fields will be appended to the entry and should be ignored by the client when unknown.

© Copyright 2001-2021 Bosch Security Systems

Entry



No. of virtual line

The number of the line which is a virtual line. Counting starts as usual from 1. Every audio input line is either a physical line (default) or a virtual line. All lines listed here are virtual lines.

No. of physical Line

The number of the physcial line which is associated to the named virtual line in the previous field. This line must be addressed for RCP commands that fail on the virtual line.

© Copyright 2001-2021 Bosch Security Systems

2.764 CONF_VIRTUAL_LINES

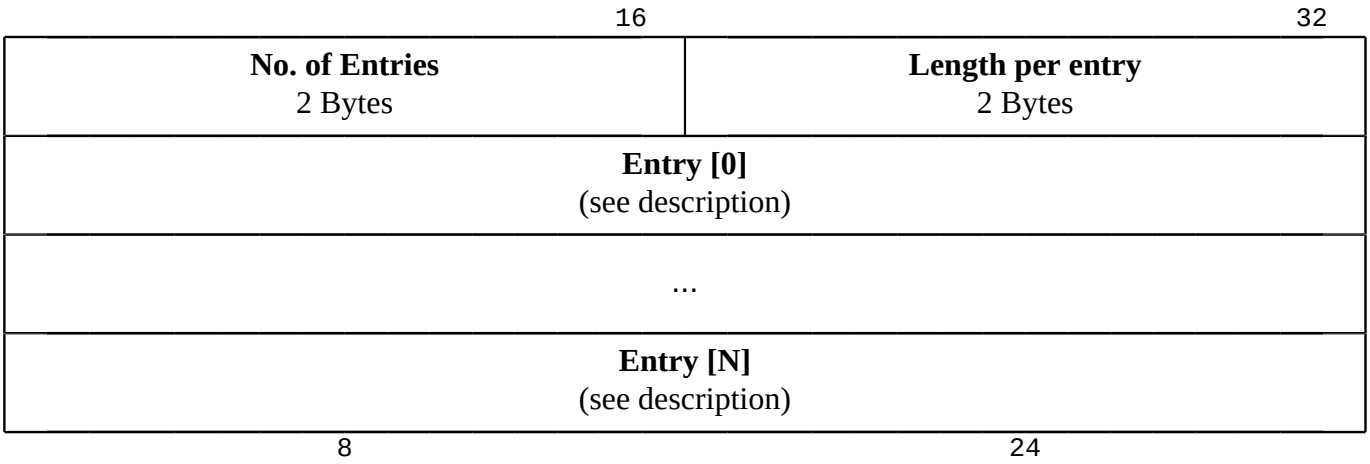
[API: video input]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bf4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	always_legacy		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

Description

The command returns which video input lines are virtual and which is the assiciated physical line. A physical line has it's own video signal - that's the usual case. A virtual line is derived from a physical line and supports only a subset of the RCP commands compared to a physical line. Some commands only make sense on physical line (e.g. brightness, contrast, ...). Those fail on virtual lines and must be applied on the associated physcial line instead. The required information can be read out using this command. Virtual lines are introduced to provide a second view on an existing physical line. This is useful e.g. to have a dewarped image on a 180 or 360 degree camera. Furthermore each virtual line has its own video analysis meta data stream (a.k.a. "VCD stream") which is also derived from the image on the physical line but already preprocessed (cropped, dewarped, ...) to fit the view on the virtual line.

Payload Structure



No. of Entries

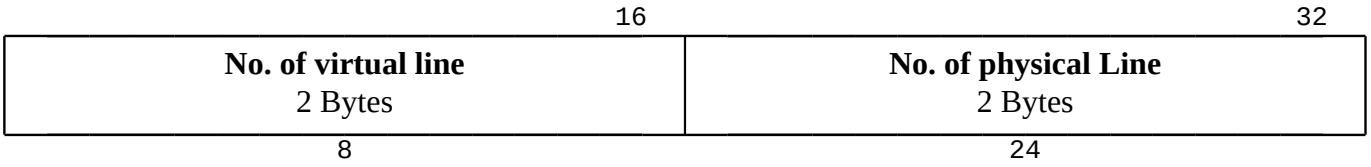
The number of virtual line descriptions.

© Copyright 2001-2021 Bosch Security Systems

Length per entry

The number of bytes per line description. This value is currently 4 but may be increased later when more information per entry must be provided. New fields will be appended to the entry and should be ignored by the client when unknown.

Entry



No. of virtual line

The number of the line which is a virtual line. Counting starts as usual from 1. Every video input line is either a physical line (default) or a virtual line. All lines listed here are virtual lines.

No. of physical Line

The number of the physcial line which is associated to the named virtual line in the previous field. This line must be addressed for RCP commands that fail on the virtual line.

© Copyright 2001-2021 Bosch Security Systems

2.765 CONF_WATERMARK

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0924	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	0=disables watermark; 1=classic watermark, 2=video authentication(low profile), 3=video authentication(mid profile), 4=video authentication(high profile)	
Write	t_octet	service	0=disables watermark; 1=classic watermark, 2=video authentication(low profile), 3=video authentication(mid profile), 4=video authentication(high profile)	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

2.766 CONF_WIFI_CONFIG_TOKEN

[API: security]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c29	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read WifiConfigToken; used for residential cloud services APP as a secret to change the Wifi network settings after initial configuration	
Write	p_string	service	Store WifiConfigToken; used for residential cloud services APP as a secret to change the Wifi network settings after initial configuration	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.767 CONF_WLAN_DEFAULT_CHANNEL

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0945	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Read the WLAN channel	
Write	t_octet	service	Write the WLAN channel	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.768 CONF_WLAN_IOCTRL

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0c4c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_octet	service	Execute ioctrl with wifi driver	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.769 CONF_WLAN_LINK_QUALITY

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b1c	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	minimal	Read WLAN link signal level	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.770 CONF_WLAN_LINK_TEST

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0b23	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_int	service	Test WLAN setup (try ability to associate to AP with current settings and return result 0=error, 1=success)	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.771 CONF_WLAN_OPERATING_MODE

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ac7	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal	Gets the operating mode of WLAN	
Write	t_octet	service	Sets the operating mode of WLAN	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

Off	0	
Auto	1	Station only
Access point	2	

© Copyright 2001-2021 Bosch Security Systems

2.772 CONF_WLAN_REGION_CODE

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0acf	None	no	no
	Datatype	**Access Level**	**Description**	
Read	t_word	minimal	Gets the region code for the WLAN adapter	
Write	t_word	service	Sets the region code for the WLAN adapter	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Values:

USA	0x10
Canada	0x20
EU	0x30
Spain	0x31
France	0x32
Japan	0x40
Japan	0x41

© Copyright 2001-2021 Bosch Security Systems

2.773 CONF_WLAN_SCAN

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ac6	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_octet	service	Performs a WLAN network scan, see detailed description	
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

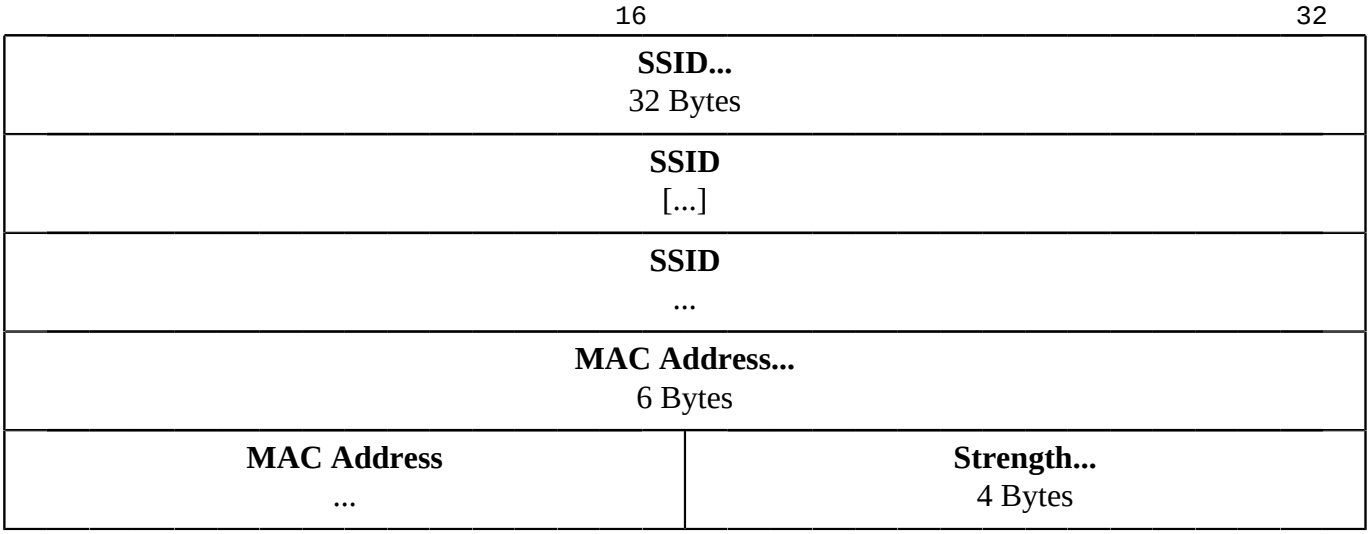
Payload Structure

Number of Entries 4 Bytes
Entry [0] (see description)
...
Entry [N] (see description)

Number of Entries

Number of scan entries

Entry



© Copyright 2001-2021 Bosch Security Systems

Strength ...	**Channel...** 4 Bytes
Channel ...	**Flags...** 4 Bytes
Flags ...	
8	24

SSID

Name of network.

MAC Address

MAC address of AccessPoint

Strength

Signal strength in range [0;5]

Channel

Channel used by AccessPoint [0;14]

Flags

	Mask	Name	Description
Bit 3	0x00000008	WPS encryption	Is set when WPS encryption is supported.
Bit 2	0x00000004	WPA2 encryption	Is set when WPA2 encryption is supported.
Bit 1	0x00000002	WPA encryption	Is set when WPA encryption is supported.
Bit 0	0x00000001	WEP encryption	Is set when WEP encryption is supported.

© Copyright 2001-2021 Bosch Security Systems

2.774 CONF_WLAN_SSID

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0943	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Read the WLAN ssid	
Write	p_string	service	Write the WLAN ssid	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.775 CONF_WLAN_WPA_PSK

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0ac3	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	service	Read the WPA pre shared key	
Write	p_string	service	Write the WPA pre shared key	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.776 CONF_WLAN_WPS_SETUP

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc4	None	no	no
	Datatype	**Access Level**	**Description**	
Read	p_string	minimal	Reports the camera WPS PIN to be entered in the access point.	
Write	p_string	service	Starts a WiFi Protected Setup (WPS) session with the provided PIN (4 or 8 digits) and SSID. - PIN: The string contains the pin, a plus sign and the SSID. E.g.: \12345678 + MyAccessPoint\" - Push-button configuration (PBC): leave string empty - If the SSID is omitted		
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.777 CONF_WLAN_WPS_STATUS

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0bc5	None	yes	no
	Datatype	**Access Level**	**Description**	
Read	t_octet	minimal		
Write	-	-	Unavailable	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

Payload Structure

Status
1 Byte

Status

Mask	Name		
Combined Values			
Mask	**Name**		
0xF0	Status		
		Not started	0
		Ongoing	1
		Successful	2
		Rejected by the access point	3
		Timeout	4
		Invalid PIN	5
0x0F	Mode		
		None	0
		PBC	1
		PIN	2

© Copyright 2001-2021 Bosch Security Systems

2.778 CONF_WPS_BUTTON_ENABLED

[API: wlan]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0be8	None	no	no
	Datatype	**Access Level**	**Description**	
Read	f_flag	user	Returns if the WPS button is enabled (1) or not (0). Default is enabled.	
Write	f_flag	service	Enables or disabled the WPS button. When the button is diabled, pressing the button has no effect.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		no	

© Copyright 2001-2021 Bosch Security Systems

2.779 CONF_WRITE_XML_TO_VCD

[API: viproc]

	Tag Code	Num Descriptor	Message	SNMP Support
	0x0baf	video line	no	no
	Datatype	**Access Level**	**Description**	
Read	-	-	Unavailable	
Write	p_string	iva	Write payload to vcd stream within a XML VCD-tag, data can be refound in VCD layer 0 and VCD layer 14. The reply is a UTC time (first 4 bytes are the seconds since 2000 followed by two bytes residual milliseconds). The payload is limitted to 1400 bytes. A very simple xml validation is performed -> tags can only have 31 bytes length and nesting is limited, CDATA is not allowed. Everything before the first xml element and after the last xml element is skipped. Only utf 8 encoding is supported.	
	CPP6/CPP7/CPP7.3		**CPP13**	
Available	yes		yes	

© Copyright 2001-2021 Bosch Security Systems

3 Appendix

© Copyright 2001-2021 Bosch Security Systems

3.1 Sample application in C

This is a sample application to interface the RCP Plus server inside the VideoJet.

This application provides six steps:

- Connecting to the RCP server over TCP port 1756
- Register at the server
- Reading the software version
- Connecting to a second unit
- Disconnect the devices
- Wait for messages

For compiling the source code we suggest Microsoft Visual C++ V5.00. Other compiler should also work fine except the winsock initialization.

Find a sample session logfile below

If you are not supplied with the source file, contact support@vcs.com and request RCPSample.c

RCPSample.c

```
/*

  RCPSample.c

  Sample RCP Plus Version 3 application

  Tested with VJ400 V6.00, VJ1000 V1.0

  Compilation tested under Microsoft Visual C++ 5.0

  -- make sure wsock32.lib is included in project->settings->linker-
>Object/Library Modules !!! --

  includes a possible output listing at the end of this file

*/


#include <stdio.h>
#include <winsock.h>

#define RCP_NETWORK_PORT         (1756)


//definable IP addresses for this test

//the VJ for testing the RCP connection - this should be an video
 encoder
#define REMOTE_VJ_IP             0x0a000001 //is "10.0.0.1"
```

© Copyright 2001-2021 Bosch Security Systems

```
//a second VJ with video decoder for testing interconnect via the RCP
 'connect to' command
#define DECODER_IP              0x0a000002 //is "10.0.0.2"


//password and length
//this is a just a default password; replace this if a password is set
 on the VJ
//if no password ist set, all password will be accepted
#define LOGIN_STRING            "+service:pass+"
#define LOGIN_STRING_LENGTH     14

//some datatypes
typedef unsigned long           DWORD;
typedef unsigned short          WORD;
typedef unsigned char           OCTET;




//structures for the RCP header and commands

//the TPKT header
struct TPKT_header
{
   OCTET    version;
   OCTET    reserved;
   WORD     size;
};


//the RCP header
struct RCPPlus_header
{
   WORD     command;
   OCTET    datatype;
   OCTET    rd_wr    :4, //Note swap these lines on big endian
 machines !!
            version  :4; //Note swap these lines on big endian
 machines !!
   OCTET    method;
   OCTET    reserved;
   WORD     client_id;
   DWORD    session_id;
   WORD     num;
   WORD     len;
};
```

© Copyright 2001-2021 Bosch Security Systems

```
//the fix part of the RCP Register command
struct h_RCP_ClientRegister
{
   OCTET    RegType;
   OCTET    Reserved;
   WORD     ClientID;

   OCTET    PwdEncryption;
   OCTET    PwdLen;
   WORD     NbrOfMessages;

   //followed by
   //WORD   Messages[n];
   //OCTET  Password[nn];
};

//the replay to the RCP register command
struct h_RCP_ClientRegisterReply
{
   OCTET    Result;
   OCTET    Level;
   WORD     ClientID;
};


//the RCP 'connect to' command
struct connect_to_header
{
   DWORD    Dest_IP;
   WORD     Reserved1;
   WORD     Flags;

   DWORD    PutChannels;
   DWORD    GetChannels;
};

//the RCP disconnect reply command
struct discon_prim_reply_header
{
   OCTET    status;
   OCTET    reserved1;
   WORD     reserved2;
   DWORD    remote_host;
};
```

© Copyright 2001-2021 Bosch Security Systems

```
int      SendPacket( SOCKET Socket, OCTET *TPKT_hdr, int TPKT_hdr_len,
 OCTET *RCP_hdr, int RCP_hdr_len,
                     OCTET *Payload, int Payload_len);
int      GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen);
void     view_hex(void *pv, int l);
WORD     Register(   SOCKET Socket);
DWORD    GetSoftwareVersion(  SOCKET Socket, WORD ClientID);
DWORD    ConnectTo(SOCKET Socket, WORD ClientID, DWORD RemoteIP);
int      Disconnect(SOCKET Socket, WORD ClientID, DWORD SessionID);
```

```
void main(int argc, char* argv[])
{
   WORD                  wVersionRequested;   //for initial access
 to the winsocks
   WSADATA               wsaData;             //for initial access
 to the winsocks
   SOCKET                Socket;              //socket for the TCP
 connection
   struct sockaddr_in    remote_sin;          //remote address

   WORD                  ClientID;            //to be assgined after
 RCP registration
   DWORD                 SoftwareVersion;     //demo access
   DWORD                 SessionID;           //to be assgined after
 RCP 'connect to'

   char                  RX_Buf[2048];        //buffer for data from
 socket
   struct RCPPlus_header *prcp_hdr;           //pointer to access
 the RCP header inside a buffer
   int                   len;                 //length of a received
 packet
   OCTET                 *Payload;            //pointer to the
 payload section


   //init winsock
   wVersionRequested = MAKEWORD( 2, 2 );
```

© Copyright 2001-2021 Bosch Security Systems

```
   if ( WSAStartup( wVersionRequested, &wsaData ) != 0 )
   {
      printf("Error starting winsock\n");
      exit(0);
   }


   //create a socket and connect to the RCP server
   Socket = socket (AF_INET, SOCK_STREAM, 0);
   if (Socket != INVALID_SOCKET)
   {
      remote_sin.sin_family         = AF_INET;
      remote_sin.sin_port           = htons (RCP_NETWORK_PORT);
      remote_sin.sin_addr.s_addr    = htonl (REMOTE_VJ_IP);

      connect(Socket, (struct sockaddr*)&remote_sin, sizeof(struct
sockaddr));
   }
   else
   {
      exit(0);
   }

   printf("============================================\n");
   printf("\n\nTry to register...\n");
   Sleep(2000);
   ClientID = Register(Socket);
   Sleep(3000);

   printf("============================================\n");
   printf("\n\nTry to read out the software version...\n");
   Sleep(2000);
   SoftwareVersion = GetSoftwareVersion(Socket, ClientID);
   printf("Software version %08x\n", SoftwareVersion);
   Sleep(3000);

   printf("============================================\n");
   printf("\n\nNow try to connect to a second VJ...\n");
   Sleep(2000);
   SessionID = ConnectTo(Socket, ClientID, DECODER_IP);
   printf("connected with SessionID %08x\n",SessionID);
   Sleep(5000);

   printf("============================================\n");
   printf("\n\nEnough for today; close connection...\n");
   Sleep(2000);
   Disconnect(Socket, ClientID, SessionID);
   Sleep(3000);
```

© Copyright 2001-2021 Bosch Security Systems

```
   printf("=========================================\n");
   printf("\n\nNow waiting for messages...\n");

   //waiting for messages
   while(1)
   {
      len = GetPacket(Socket, RX_Buf, sizeof(RX_Buf));

      //position to RCP header
      prcp_hdr = (struct RCPPlus_header*)RX_Buf;

      //check for valid message
      if(prcp_hdr->method == 0x02)
      {
         printf("Message %04x num %04x type %02x\n"   ,ntohs(prcp_hdr-
>command), ntohs(prcp_hdr->num)
                                                      , prcp_hdr-
>datatype);

         len -= sizeof(struct RCPPlus_header);

         //position to payload section
         Payload = (OCTET *)(RX_Buf + sizeof(struct RCPPlus_header));
         printf("Payload (%d bytes)\n", len);
         view_hex(Payload, len);

      }
   }
}
```

```
WORD Register(SOCKET Socket)
{
   struct TPKT_header              tpkt_hdr;                    //
the TPKT header
   struct RCPPlus_header           rcp_hdr, *prcp_hdr;          //
the RCP header and a pointer for
                                                                //
access inside an RX buffer
   struct h_RCP_ClientRegister     *prcp_reg_hdr;               //
pointer to the RCP registration struct
   struct h_RCP_ClientRegisterReply *rcp_reg_reply_hdr;         //
pointer to access the RCP registration
```

© Copyright 2001-2021 Bosch Security Systems

```
                                                                    //
reply inside a buffer
   WORD                             NbrOfMessages, n;               //
counts the number of messages in the
                                                                    //
regitration command
   WORD                             Messages[]={0xFFC3, 0x01C0};    //
the messages to be registerd
   OCTET                            *Payload , *pw;                 //
pointer to access the payload section
   WORD                             PayloadLen, wTPKTsize;          //
length descriptors
   char                             RX_Buf[2048];                   //
buffer for data from socket
   WORD                             ClientID;                       //
assigned client IP; to be taken
                                                                    //
from the reply


   //fill out the RCP registration
   PayloadLen = sizeof(struct h_RCP_ClientRegister) + sizeof(Messages)
 + LOGIN_STRING_LENGTH;
   Payload = malloc(PayloadLen);
   if(!Payload)
   {
      return 0;
   }

   //use pw as a write pointer
   pw = Payload;

   //write fix header
   prcp_reg_hdr = (struct h_RCP_ClientRegister*)pw;
   prcp_reg_hdr->RegType         =        0x01;
        //Normal registration
   prcp_reg_hdr->Reserved        =        0;
   prcp_reg_hdr->ClientID        = htons( 0x0000);
        //no ClientID in normal

        //registration
   prcp_reg_hdr->PwdEncryption   =        0x00;
        //Plain text password encrytion
   prcp_reg_hdr->PwdLen          =        LOGIN_STRING_LENGTH;
        //Length of password

        //string (without '\0')
```

© Copyright 2001-2021 Bosch Security Systems

```
  NbrOfMessages                 =       sizeof(Messages) /
sizeof(WORD);
  prcp_reg_hdr->NbrOfMessages   = htons( NbrOfMessages);
       //Nbr of messages that follow
  pw += sizeof(struct h_RCP_ClientRegister);

  //write message array
  for(n=0;n<NbrOfMessages;n++)
  {
     ((WORD*)pw)[n]             = htons( Messages[n]);
       //fill in message requests
  }
  pw+=sizeof(Messages);

  //write password string
  memcpy(pw,                            LOGIN_STRING,
LOGIN_STRING_LENGTH); //fill the password string itself


  //fill out the RCP header
  rcp_hdr.command           = htons( 0xFF00);
       //Command 'RCP register'
  rcp_hdr.datatype          =       0x0c;
       //Datatype P_OCTET
  rcp_hdr.rd_wr             =       0x1;
       //WRITE
  rcp_hdr.version           =       0x3;
       //Version 3
  rcp_hdr.method            =       0x00;
       //REQUEST
  rcp_hdr.reserved          =       0;
  rcp_hdr.client_id         = htons( 0x0000);
       //currently no ClientID available
  rcp_hdr.session_id        = htonl( 0x00000000);
       //currently no SessionID available
  rcp_hdr.num               = htons( 0x0000);
       //no num parameter needed
  rcp_hdr.len               = htons( PayloadLen);
       //length of the payload section

  //fill out TPKT
  tpkt_hdr.version          = 3;
       //TPKT version 3
  tpkt_hdr.reserved         = 0;
  wTPKTsize                 = PayloadLen + sizeof(rcp_hdr) +
sizeof(tpkt_hdr);
  tpkt_hdr.size             = htons( wTPKTsize);
       //overall length
```

© Copyright 2001-2021 Bosch Security Systems

```
        //(including TPKT itself)



  //send to packet to the network
  if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                               (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                               Payload,             PayloadLen))
  {
     printf("Send packet error\n");
     exit(0);
  }

  //get the reply
  if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
RCPPlus_header) + sizeof(struct h_RCP_ClientRegisterReply))
  {
     //position to RCP header
     prcp_hdr = (struct RCPPlus_header*)RX_Buf;

     //check for error message
     if(prcp_hdr->method == 0x03)
     {
        OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
        printf("An error was returned code %02x\n",*ErrorCode);
        return 0;
     }

     //position to payload section
     rcp_reg_reply_hdr = (struct h_RCP_ClientRegisterReply *)(RX_Buf
+ sizeof(struct RCPPlus_header));

     //get assigned ClientID
     ClientID = ntohs(rcp_reg_reply_hdr->ClientID);

     printf("Register reply:\n");
     printf("Result:     %02x\n",rcp_reg_reply_hdr->Result);
     printf("Level:      %02x\n",rcp_reg_reply_hdr->Level);
     printf("ClientID    %04x\n",ClientID);
  }
  else
  {
     printf("Register failed\n");
  }

  return ClientID;
```

© Copyright 2001-2021 Bosch Security Systems

```
}
```

```
DWORD GetSoftwareVersion(SOCKET Socket, WORD ClientID)
{
   struct TPKT_header                tpkt_hdr;            //the TPKT
 header
   struct RCPPlus_header             rcp_hdr, *prcp_hdr;  //the RCP
 header and a pointer for access
                                                          //inside
 an RX buffer
   char                              RX_Buf[2048];        //buffer
 for data from socket
   DWORD                             *pDWORD;             //pointer
 to the reply payload
   DWORD                             version=0;


   //Next task; read out the software version number
   //fill out the RCP header
   rcp_hdr.command           = htons( 0x002F);            //Command
 'Software version'
   rcp_hdr.datatype          =        0x08;               //Datatype
 T_DWORD
   rcp_hdr.rd_wr             =        0x0;                //READ
   rcp_hdr.version           =        0x3;                //Version
 3
   rcp_hdr.method            =        0x00;               //REQUEST
   rcp_hdr.reserved          =        0;
   rcp_hdr.client_id         = htons( ClientID);          //use
 clientID from register reply
   rcp_hdr.session_id        = htonl( 0x00000000);        //
currently no SessionID available
   rcp_hdr.num               = htons( 0x0000);            //no num
 parameter needed
   rcp_hdr.len               = 0;                         //no
 payload in READ command
```

© Copyright 2001-2021 Bosch Security Systems

```
    //fill out TPKT
    tpkt_hdr.version           = 3;                          //TPKT
  version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(rcp_hdr) +
  sizeof(tpkt_hdr));

    //send to packet to the network
    if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
  sizeof(tpkt_hdr),
                                  (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                  NULL,                0))
    {
       printf("Send packet error\n");
       exit(0);
    }

    //get the reply
    if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
  RCPPlus_header) + sizeof(DWORD))
    {
       //position to RCP header
       prcp_hdr = (struct RCPPlus_header*)RX_Buf;

       //check for error message
       if(prcp_hdr->method == 0x03)
       {
          OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
          printf("An error was returned code %02x\n",*ErrorCode);
          return 0;;
       }

       //position to payload section
       pDWORD = (DWORD *)(RX_Buf + sizeof(struct RCPPlus_header));

       version = ntohl(*pDWORD);
    }
    else
    {
       printf("failed to read software version\n");
    }


    return version;
  }
```

© Copyright 2001-2021 Bosch Security Systems

```
DWORD ConnectTo(SOCKET Socket, WORD ClientID, DWORD RemoteIP)
{
   struct TPKT_header                  tpkt_hdr;
  //the TPKT header
   struct RCPPlus_header               rcp_hdr, *prcp_hdr;
  //the RCP header and a pointer for

  //access inside an RX buffer
   char                                RX_Buf[2048];
  //buffer for data from socket
   struct connect_to_header            rcp_con_hdr,
 *rcp_con_reply_hdr; //the connect command and a pointer

  //for access inside an RX buffer
   DWORD                               sessionID=0;


   //fill out the RCP connect to header
   rcp_con_hdr.Dest_IP        = htonl( RemoteIP);
  //the remote IP address
   rcp_con_hdr.Reserved1      = htons( 0);
   rcp_con_hdr.Flags          = htons( 0);
  //use no flags
   rcp_con_hdr.PutChannels    = htonl( 0x00000001);
  //put video
   rcp_con_hdr.GetChannels    = htonl( 0x00000000);
  //no get channels


   //fill out the RCP header
   rcp_hdr.command            = htons( 0xFFCC);
  //Command 'RCP connect to'
   rcp_hdr.datatype           =        0x0c;
  //Datatype P_OCTET
   rcp_hdr.rd_wr              =        0x1;
  //WRITE
   rcp_hdr.version            =        0x3;
  //Version 3
```

© Copyright 2001-2021 Bosch Security Systems

```
   rcp_hdr.method             =        0x00;
  //REQUEST
   rcp_hdr.reserved           =        0;
   rcp_hdr.client_id          = htons( ClientID);
  //use clientID from register reply
   rcp_hdr.session_id         = htonl( 0x00000000);
  //currently no SessionID available
   rcp_hdr.num                = htons( 0x0000);
  //no num parameter needed
   rcp_hdr.len                = htons( sizeof(rcp_con_hdr));
  //length of the payload section

   //fill out TPKT
   tpkt_hdr.version           = 3;
  //TPKT version 3
   tpkt_hdr.reserved          = 0;
   //overall length(including TPKT itself)
   tpkt_hdr.size              = htons( sizeof(rcp_con_hdr) +
sizeof(rcp_hdr) + sizeof(tpkt_hdr));

   //send to packet to the network
   if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                                    (OCTET*)&rcp_hdr,
sizeof(rcp_hdr),
                                    (OCTET*)&rcp_con_hdr,
sizeof(rcp_con_hdr)))
   {
      printf("Send packet error\n");
      exit(0);
   }

   //get the reply
   if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
RCPPlus_header) + sizeof(struct connect_to_header))
   {
      //position to RCP header
      prcp_hdr = (struct RCPPlus_header*)RX_Buf;

      //check for error message
      if(prcp_hdr->method == 0x03)
      {
         OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
         printf("An error was returned code %02x\n",*ErrorCode);
         return 0;
      }

      //position to payload section
```

© Copyright 2001-2021 Bosch Security Systems

```
      rcp_con_reply_hdr = (struct connect_to_header *)(RX_Buf +
 sizeof(struct RCPPlus_header));

      printf("connect to reply:\n");
      printf("Dest IP:     %08x\n",ntohl(rcp_con_reply_hdr-
>Dest_IP));
      printf("PutChannels  %08x\n",ntohl(rcp_con_reply_hdr-
>PutChannels));
      printf("GetChannels  %08x (Note VJ1000 V1.0 may return invaild
 data here)\n",ntohl(rcp_con_reply_hdr->GetChannels));

      sessionID = ntohl(prcp_hdr->session_id);
   }
   else
   {
      printf("Connect to failed\n");
   }

   return sessionID;
}
```

```
int Disconnect(SOCKET Socket, WORD ClientID, DWORD SessionID)
{
   struct TPKT_header                tpkt_hdr;              //the
 TPKT header
   struct RCPPlus_header             rcp_hdr, *prcp_hdr;    //the
 RCP header and a pointer for
                                                            //
access inside an RX buffer
   char                              RX_Buf[2048];          //
buffer for data from socket
   struct discon_prim_reply_header   *rcp_discon_reply_hdr; //the
 pointer for  access inside
                                                            //an RX
 buffer


   //Next task; read out the software version number
```

© Copyright 2001-2021 Bosch Security Systems

```
    //fill out the RCP header
    rcp_hdr.command            = htons( 0xFF0D);                //
Command 'disconnect'
    rcp_hdr.datatype           =        0x0c;                   //
Datatype P_OCTET
    rcp_hdr.rd_wr              =        0x1;                    //WRITE
    rcp_hdr.version            =        0x3;                    //
Version 3
    rcp_hdr.method             =        0x00;                   //
REQUEST
    rcp_hdr.reserved           =        0;
    rcp_hdr.client_id          = htons( ClientID);              //use
 clientID from register reply
    rcp_hdr.session_id         = htonl( SessionID);             //use
 clientID from connect to reply
    rcp_hdr.num                = htons( 0x0000);                //no
 num parameter needed
    rcp_hdr.len                = 0;                             //no
 payload needed here

    //fill out TPKT
    tpkt_hdr.version           = 3;                             //TPKT
 version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(rcp_hdr) +
 sizeof(tpkt_hdr));

    //send to packet to the network
    if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                  (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                  NULL,                0))
    {
       printf("Send packet error\n");
       exit(0);
    }

    //get the reply
    if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
 RCPPlus_header) + sizeof(struct discon_prim_reply_header))
    {
       //position to RCP header
       prcp_hdr = (struct RCPPlus_header*)RX_Buf;

       //check for valid error message
       if(prcp_hdr->method == 0x03)
       {
```

© Copyright 2001-2021 Bosch Security Systems

```
         OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
         printf("An error was returned code %02x\n",*ErrorCode);
         exit(0);
      }

      //position to payload section
      rcp_discon_reply_hdr = (struct discon_prim_reply_header*)(RX_Buf
 + sizeof(struct RCPPlus_header));

      printf("Disconnect reply:\n");
      printf("Status      %02x\n",rcp_discon_reply_hdr->status);
      printf("Remote Host %08x\n",ntohl(rcp_discon_reply_hdr-
>remote_host));
   }
   else
   {
      printf("Disconnect failed\n");
   }

   return 0;
}
```

```
int SendPacket (
                  SOCKET Socket,
                  OCTET *TPKT_hdr,  int TPKT_hdr_len,
                  OCTET *RCP_hdr,   int RCP_hdr_len,
                  OCTET *Payload,   int Payload_len
               )
{
   int err=0;

   printf("\n");

   //send in 3 parts (this is allowed)

   //send the TPKT header
   err += send(Socket, (char*)TPKT_hdr,   TPKT_hdr_len,  0);
   printf("snd TPKT hdr:\n"); view_hex(TPKT_hdr, TPKT_hdr_len);
```

© Copyright 2001-2021 Bosch Security Systems

```
    //send the RCP header
    err += send(Socket, (char*)RCP_hdr,    RCP_hdr_len,   0);
    printf("snd RCP hdr:\n"); view_hex(RCP_hdr, RCP_hdr_len);

    //send the payload, if one
    if(Payload_len)
    {
       err += send(Socket, (char*)Payload,    Payload_len,   0);
       printf("snd Payload:\n"); view_hex(Payload, Payload_len);
    }

    printf("\n");

    //have all bytes been sent ?
    if (err != TPKT_hdr_len + RCP_hdr_len + Payload_len )
    {
       return -1;
    }
    return 0;
}
```

```
int GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen)
{
    int len, wait_bytes;
    struct TPKT_header                  *ptpkt_hdr;

    printf("\n");

    if(RX_BufMaxLen < sizeof(struct TPKT_header))
    {
       return -1;
    }

    //wait for reply

    //wait for the TPKT header to be arrived completely
    do
```

© Copyright 2001-2021 Bosch Security Systems

```
    {
       //just check; do not empty TCP buffer
       len = recv(Socket, RX_Buf, sizeof(struct TPKT_header),
 MSG_PEEK);
    } while(len != sizeof(struct TPKT_header));
    //all bytes arrived - empty TCP buffer
    recv(Socket, RX_Buf, sizeof(struct TPKT_header), 0);

    printf("rcv TPKT hdr:\n"); view_hex(RX_Buf, len);


    //Now the length of the entire RCP packet can be determined
    ptpkt_hdr = (struct TPKT_header*)RX_Buf;
    wait_bytes = ntohs(ptpkt_hdr->size) - sizeof(struct TPKT_header);

    if(RX_BufMaxLen < wait_bytes)
    {
       return -1;
    }

    do
    {
       //just check; do not empty TCP buffer
       len = recv(Socket, RX_Buf, wait_bytes, MSG_PEEK);
    } while(len != wait_bytes);
    //all bytes arrived - empty TCP buffer
    recv(Socket, RX_Buf, wait_bytes, 0);

    printf("rcv Payload:\n"); view_hex(RX_Buf, len);
    printf("\n");

    return wait_bytes;
}
```

```
void view_hex(void *pv, int l)
{
    //improved display of hex strings
    OCTET *p = (OCTET *)pv;
```

© Copyright 2001-2021 Bosch Security Systems

```
   int i;

   for (i=0;i<l;i++)
   {
      if(!(i%16) && i) printf("\n");

      printf("%02x ",p[i]);
   }
   printf("\n");
}
```

```
/*


//Possible output when running this test program

(<<-- Note: xxxxx...xxx are additional comments)


==========================================


Try to register...

snd TPKT hdr:
03 00 00 2e
snd RCP hdr:
ff 00 0c 31 00 00 00 00 00 00 00 00 00 00 1a
snd Payload:
01 00 00 00 00 0e 00 02 ff c3 01 c0 2b 73 65 72
76 69 63 65 3a 70 61 73 73 2b


rcv TPKT hdr:
03 00 00 18
rcv Payload:
ff 00 0c 31 01 00 00 77 00 00 00 00 00 00 00 04
01 02 00 77

Register reply:
Result:     01
Level:      02
```

© Copyright 2001-2021 Bosch Security Systems

```
ClientID    0077 <<-- Note: this may vary
=========================================


Try to read out the software version...

snd TPKT hdr:
03 00 00 14
snd RCP hdr:
00 2f 08 30 00 00 00 77 00 00 00 00 00 00 00 00


rcv TPKT hdr:
03 00 00 18
rcv Payload:
00 2f 08 30 01 00 00 77 00 00 00 00 00 00 00 04
11 00 06 00

Software version 11000600
=========================================


Now try to connect to a second VJ...

snd TPKT hdr:
03 00 00 24
snd RCP hdr:
ff cc 0c 31 00 00 00 77 00 00 00 00 00 00 00 10
snd Payload:
0a 00 00 02 00 00 00 00 00 00 00 01 00 00 00 00


rcv TPKT hdr:
03 00 00 24
rcv Payload:
ff cc 0c 31 01 00 00 77 c8 28 00 26 00 00 00 10
0a 00 00 02 00 00 00 00 00 00 00 01 00 00 00 00

connect to reply:
Dest IP:     0a000002
PutChannels  00000001
GetChannels  00000000 (Note VJ1000 V1.0 may return invaild data here)
connected with SessionID c8280026 <<-- Note: this may vary
=========================================


Enough for today; close connection...
```

© Copyright 2001-2021 Bosch Security Systems

```
snd TPKT hdr:
03 00 00 14
snd RCP hdr:
ff 0d 0c 31 00 00 00 77 c8 28 00 26 00 00 00 00


rcv TPKT hdr:
03 00 00 1c
rcv Payload:
ff 0d 0c 31 01 00 00 77 c8 28 00 26 00 00 00 08
01 00 ca e4 0a 00 00 02

Disconnect reply:
Status      01
Remote Host 0a000002
=========================================

<<-- Note: these messages will only be received if an input event on
 the VJ takes place


Now waiting for messages...

rcv TPKT hdr:
03 00 00 15
rcv Payload:
01 c0 00 30 02 00 00 77 00 00 00 00 00 01 00 01
00

Message 01c0 num 0001 type 00
Payload (1 bytes)
00

rcv TPKT hdr:
03 00 00 15
rcv Payload:
01 c0 00 30 02 0b 00 77 00 00 00 00 00 01 00 01
01

Message 01c0 num 0001 type 00
Payload (1 bytes)
01

rcv TPKT hdr:
03 00 00 15
rcv Payload:
01 c0 00 30 02 00 00 77 00 00 00 00 00 01 00 01
00
```

© Copyright 2001-2021 Bosch Security Systems

```
Message 01c0 num 0001 type 00
Payload (1 bytes)
00

rcv TPKT hdr:
03 00 00 15
rcv Payload:
01 c0 00 30 02 00 00 77 00 00 00 00 00 01 00 01
01

Message 01c0 num 0001 type 00
Payload (1 bytes)
01

*/
```

© Copyright 2001-2021 Bosch Security Systems

3.2 Sample code for connecting to a VideoJet using the UDP protocol

This is a sample application to interface the RCP Plus server inside the VideoJet.

This application provides six steps:

- Connecting to the RCP server over TCP port 1756
- Register at the server
- Open a UDP socket for receiving video data
- Send a connect primitive
- Receives video data
- Send keep-alives to the VideoJet

For compiling the source code we suggest Microsoft Visual C++ V5.00. Other compiler should also work fine except the winsock initialization.

If you are not supplied with the source file, contact support@vcs.com and request RCPConnect_UDP.c

RCPConnect_UDP.c

```
 /*

  RCPConnect_UDP.c

  Sample RCP Plus Version 3 application

  Tested with VJ400 V6.00, VJ1000 V1.0

  Compilation tested under Microsoft Visual C++ 5.0

  -- make sure wsock32.lib is included in project->settings->linker-
>Object/Library Modules !!! --

  includes a possible output listing at the end of this file

*/


#include <stdio.h>
#include <winsock.h>

#define RCP_NETWORK_PORT         (1756)


//definable IP addresses for this test

//the VJ for testing the RCP connection - this should be an video
 encoder
```

© Copyright 2001-2021 Bosch Security Systems

```
#define REMOTE_VJ_IP            0x0a000001 //is "10.0.0.1"



//password and length
//this is a just a default password; replace this if a password is set
 on the VJ
//if no password ist set, all password will be accepted
#define LOGIN_STRING            "+service:pass+"
#define LOGIN_STRING_LENGTH     14


//port for UDP video receiving; this could also be a dynamically
 assignd port from the stack
//see socket binding
#define VIDEO_RX_PORT           1234

//some datatypes
typedef unsigned long           DWORD;
typedef unsigned short          WORD;
typedef unsigned char           OCTET;



//structures for the RCP header and commands

//the TPKT header
struct TPKT_header
{
   OCTET    version;
   OCTET    reserved;
   WORD     size;
};


//the RCP header
struct RCPPlus_header
{
   WORD     command;
   OCTET    datatype;
   OCTET    rd_wr    :4, //Note swap these lines on big endian
 machines !!
            version  :4; //Note swap these lines on big endian
 machines !!
   OCTET    method;
   OCTET    reserved;
   WORD     client_id;
```

© Copyright 2001-2021 Bosch Security Systems

```
    DWORD    session_id;
    WORD     num;
    WORD     len;
};


//the fix part of the RCP Register command
struct h_RCP_ClientRegister
{
    OCTET    RegType;
    OCTET    Reserved;
    WORD     ClientID;

    OCTET    PwdEncryption;
    OCTET    PwdLen;
    WORD     NbrOfMessages;

    //followed by
    //WORD   Messages[n];
    //OCTET  Password[nn];
};

//the replay to the RCP register command
struct h_RCP_ClientRegisterReply
{
    OCTET    Result;
    OCTET    Level;
    WORD     ClientID;
};



//structures and defines for requesting video
struct con_prim_header
{
    OCTET    method;
    OCTET    media;
    OCTET    status;
    OCTET    reverse_login;
    DWORD    reserved2;
};


#define METHOD_GET              0x00
#define METHOD_PUT              0x01

#define MEDIA_CHANNEL_VIDEO     0x01
#define MEDIA_CHANNEL_AUDIO     0x02
```

© Copyright 2001-2021 Bosch Security Systems

```
#define MEDIA_CHANNEL_DATA       0x03

#define CODING_MPEG4             0x0004
#define CODING_MPEG2             0x0008
#define CODING_META              0x0010


#define RESOLUTION_QCIF          0x0001
#define RESOLUTION_CIF           0x0002
#define RESOLUTION_2CIF          0x0004
#define RESOLUTION_4CIF          0x0008
#define RESOLUTION_CUSTOM        0x0010

#define MEP_RTP                  0x01
#define MEP_TCP_OUTBAND          0x04

struct media_desc_video
{
   OCTET    media_encapsulation;
   OCTET    flags;
   WORD     media_port;
   DWORD    media_host;

   OCTET    coder;
   OCTET    line;
   WORD     media_ctrl_port;
   DWORD    media_ctrl_host;

   WORD     coding;
   WORD     resolution;
   WORD     reserved6;
   WORD     bandwidth;
};



int      SendPacket( SOCKET Socket, OCTET *TPKT_hdr, int TPKT_hdr_len,
 OCTET *RCP_hdr, int RCP_hdr_len,
                     OCTET *Payload, int Payload_len);
int      GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen);
void     view_hex(void *pv, int l);
WORD     Register(   SOCKET Socket);
DWORD    SendConnectPrimitive(SOCKET Socket, WORD ClientID, WORD
 local_UDP_Port);
int      SendConnectionKeepAlive(SOCKET Socket, WORD ClientID, DWORD
 SessionID);
int      Disconnect(SOCKET Socket, WORD ClientID, DWORD SessionID);
```

© Copyright 2001-2021 Bosch Security Systems

```
void main(int argc, char* argv[])
{
   WORD                  wVersionRequested;   //for initial access
 to the winsocks
   WSADATA               wsaData;             //for initial access
 to the winsocks
   SOCKET                Socket;              //socket for the TCP
 connection
   struct sockaddr_in    remote_sin;          //remote address

   WORD                  ClientID;            //to be assgined after
 RCP registration
   DWORD                 SessionID;           //to be assgined after
 RCP 'connect to'

   char                  RX_Buf[2048];        //buffer for data from
 socket
   int                   len;                 //length of a received
 packet
   DWORD                 cnt=0;               //used for a simple
 timeout mechanism
   unsigned long int     nNoBlock=1;          //for setting sockets
 to non-blocking
   struct sockaddr_in    stFrom;              //binding structure
 for UDP rx socket
   SOCKET                VideoDataSocket;     //the video receivinbg
 socket


   //init winsock
   wVersionRequested = MAKEWORD( 2, 2 );
   if ( WSAStartup( wVersionRequested, &wsaData ) != 0 )
   {
      printf("Error starting winsock\n");
      exit(0);
   }
```

© Copyright 2001-2021 Bosch Security Systems

```
    //create a socket and connect to the RCP server
    Socket = socket (AF_INET, SOCK_STREAM, 0);
    if (Socket != INVALID_SOCKET)
    {
       remote_sin.sin_family         = AF_INET;
       remote_sin.sin_port           = htons (RCP_NETWORK_PORT);
       remote_sin.sin_addr.s_addr    = htonl (REMOTE_VJ_IP);

       connect(Socket, (struct sockaddr*)&remote_sin, sizeof(struct
  sockaddr));
    }
    else
    {
       exit(0);
    }

    printf("=============================================\n");
    printf("\n\nTry to register...\n");
    Sleep(2000);
    ClientID = Register(Socket);
    Sleep(3000);

    printf("=============================================\n");
    printf("\n\nOpen UDP socket for receiving video data...\n");

    //open a UDP port for receiving the video data
    VideoDataSocket = socket(AF_INET,SOCK_DGRAM,0);
    if (VideoDataSocket == SOCKET_ERROR)
    {
       exit(0);
    }

    // Bind a receive port to the socket
    stFrom.sin_family       = AF_INET;
    stFrom.sin_port         = htons(VIDEO_RX_PORT);
    stFrom.sin_addr.s_addr  = htonl(INADDR_ANY);
    if (SOCKET_ERROR == bind (VideoDataSocket, (struct sockaddr *)
 (&stFrom), sizeof (stFrom)))
    {
       exit(0);
    }



    //make UDP socket non-blocking
    if (SOCKET_ERROR == ioctlsocket(VideoDataSocket, FIONBIO,
  &nNoBlock))
```

© Copyright 2001-2021 Bosch Security Systems

```
    {
       exit(0);
    }



    SessionID = SendConnectPrimitive(Socket, ClientID, VIDEO_RX_PORT);

    //make RCP socket non-blocking
    if (SOCKET_ERROR == ioctlsocket(Socket, FIONBIO, &nNoBlock))
    {
       exit(0);
    }



 while(1)
    {
       cnt++;

       //a very, very simple timeout mechanism; a timeout retrigger
  should be sent every 2-5 seconds
       if(!(cnt%100000))
       {
          SendConnectionKeepAlive(Socket, ClientID, SessionID);
       }


       GetPacket(Socket, RX_Buf, sizeof(RX_Buf));
       //don't check responses for the keep-alive at the moment;


       //receive the video data
       len = recvfrom(VideoDataSocket, RX_Buf, sizeof(RX_Buf), 0, NULL,
  NULL);
       if(len > 0)
       {
          printf("rx %d\n",len);
       }

       Sleep(1);


    }
 }
```

© Copyright 2001-2021 Bosch Security Systems

```
WORD Register(SOCKET Socket)
{
   struct TPKT_header               tpkt_hdr;                    //
the TPKT header
   struct RCPPlus_header            rcp_hdr, *prcp_hdr;          //
the RCP header and a pointer for
                                                                 //
access inside an RX buffer
   struct h_RCP_ClientRegister      *prcp_reg_hdr;               //
pointer to the RCP registration struct
   struct h_RCP_ClientRegisterReply *rcp_reg_reply_hdr;          //
pointer to access the RCP registration
                                                                 //
reply inside a buffer
   WORD                             NbrOfMessages, n;           //
counts the number of messages in the
                                                                 //
regitration command
   WORD                             Messages[]={0xFFC3, 0x01C0}; //
the messages to be registerd
   OCTET                            *Payload , *pw;             //
pointer to access the payload section
   WORD                             PayloadLen, wTPKTsize;      //
length descriptors
   char                             RX_Buf[2048];               //
buffer for data from socket
   WORD                             ClientID;                   //
assigned client IP; to be taken
                                                                 //
from the reply


   //fill out the RCP registration
   PayloadLen = sizeof(struct h_RCP_ClientRegister) + sizeof(Messages)
 + LOGIN_STRING_LENGTH;
   Payload = malloc(PayloadLen);
   if(!Payload)
   {
      return 0;
   }
```

© Copyright 2001-2021 Bosch Security Systems

```
  //use pw as a write pointer
  pw = Payload;

  //write fix header
  prcp_reg_hdr = (struct h_RCP_ClientRegister*)pw;
  prcp_reg_hdr->RegType         =        0x01;
       //Normal registration
  prcp_reg_hdr->Reserved        =        0;
  prcp_reg_hdr->ClientID        = htons( 0x0000);
       //no ClientID in normal

       //registration
  prcp_reg_hdr->PwdEncryption   =        0x00;
       //Plain text password encrytion
  prcp_reg_hdr->PwdLen          =        LOGIN_STRING_LENGTH;
       //Length of password

       //string (without '\0')
  NbrOfMessages                 =        sizeof(Messages) /
sizeof(WORD);
  prcp_reg_hdr->NbrOfMessages   = htons( NbrOfMessages);
       //Nbr of messages that follow
  pw += sizeof(struct h_RCP_ClientRegister);

  //write message array
  for(n=0;n<NbrOfMessages;n++)
  {
     ((WORD*)pw)[n]             = htons( Messages[n]);
       //fill in message requests
  }
  pw+=sizeof(Messages);

  //write password string
  memcpy(pw,                             LOGIN_STRING,
LOGIN_STRING_LENGTH); //fill the password string itself


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF00);
       //Command 'RCP register'
  rcp_hdr.datatype           =        0x0c;
       //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
       //WRITE
  rcp_hdr.version            =        0x3;
       //Version 3
```

© Copyright 2001-2021 Bosch Security Systems

```
  rcp_hdr.method             =        0x00;
       //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( 0x0000);
       //currently no ClientID available
  rcp_hdr.session_id         = htonl( 0x00000000);
       //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
       //no num parameter needed
  rcp_hdr.len                = htons( PayloadLen);
       //length of the payload section

  //fill out TPKT
  tpkt_hdr.version           = 3;
       //TPKT version 3
  tpkt_hdr.reserved          = 0;
  wTPKTsize                  = PayloadLen + sizeof(rcp_hdr) +
sizeof(tpkt_hdr);
  tpkt_hdr.size              = htons( wTPKTsize);
       //overall length

       //(including TPKT itself)



  //send to packet to the network
  if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                               (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                               Payload,             PayloadLen))
  {
     printf("Send packet error\n");
     exit(0);
  }

  //get the reply
  if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
RCPPlus_header) + sizeof(struct h_RCP_ClientRegisterReply))
  {
     //position to RCP header
     prcp_hdr = (struct RCPPlus_header*)RX_Buf;

     //check for error message
     if(prcp_hdr->method == 0x03)
     {
        OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
        printf("An error was returned code %02x\n",*ErrorCode);
        return 0;
```

© Copyright 2001-2021 Bosch Security Systems

```
      }

      //position to payload section
      rcp_reg_reply_hdr = (struct h_RCP_ClientRegisterReply *)(RX_Buf
 + sizeof(struct RCPPlus_header));

      //get assigned ClientID
      ClientID = ntohs(rcp_reg_reply_hdr->ClientID);

      printf("Register reply:\n");
      printf("Result:     %02x\n",rcp_reg_reply_hdr->Result);
      printf("Level:      %02x\n",rcp_reg_reply_hdr->Level);
      printf("ClientID    %04x\n",ClientID);
   }
   else
   {
      printf("Register failed\n");
   }

   return ClientID;

}
```

```
DWORD SendConnectPrimitive(SOCKET Socket, WORD ClientID, WORD
 local_UDP_Port)
{

   struct TPKT_header                  tpkt_hdr;
  //the TPKT header
   struct RCPPlus_header               rcp_hdr, *prcp_hdr;
  //the RCP header and a pointer for

  //access inside an RX buffer
   char                                TX_Buf[256], RX_Buf[2048], *po;
       //buffer for data from socket
   struct con_prim_header              *h_req;
   struct media_desc_video             *md_video;
   WORD                                payload_len;

   DWORD                               sessionID=0;
```

© Copyright 2001-2021 Bosch Security Systems

```
  po = TX_Buf;

  h_req = (struct con_prim_header*)po;

  //fill out the RCP connect primitive
  h_req->method              =  METHOD_GET;
  h_req->media               =  MEDIA_CHANNEL_VIDEO;
  h_req->reverse_login       =  FALSE;
  po += sizeof(struct con_prim_header);

  md_video = (struct media_desc_video*)po;
  md_video->media_encapsulation =        MEP_RTP;
  md_video->media_port          = htons( local_UDP_Port );
  md_video->media_host          = htonl( 0 ); //transmitter shall
take the destination address from the received packet
  md_video->media_ctrl_port     = htons( 0 );
  md_video->media_ctrl_host     = htonl( 0 );
  md_video->coding              = htons( CODING_MPEG4 );
  md_video->resolution          = htons( RESOLUTION_CIF );
  md_video->bandwidth           = htons( 1000 );
  md_video->coder               = 0;
  md_video->line                = 1;
  md_video->flags               = 0;
  po += sizeof(struct media_desc_video);
  payload_len = (WORD)(po - TX_Buf);


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF0C);
 //Command 'RCP connect primitive'
  rcp_hdr.datatype           =        0x0c;
 //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
 //WRITE
  rcp_hdr.version            =        0x3;
 //Version 3
  rcp_hdr.method             =        0x00;
 //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( ClientID);
 //use clientID from register reply
  rcp_hdr.session_id         = htonl( 0x00000000);
 //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
 //no num parameter needed
```

© Copyright 2001-2021 Bosch Security Systems

```
    rcp_hdr.len                = htons( payload_len);
   //length of the payload section

    //fill out TPKT
    tpkt_hdr.version           = 3;
   //TPKT version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( (WORD)(payload_len +
 sizeof(rcp_hdr) + sizeof(tpkt_hdr)));

    //send to packet to the network
    if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                     (OCTET*)&rcp_hdr,
 sizeof(rcp_hdr),
                                     TX_Buf,   payload_len))
    {
       printf("Send packet error\n");
       exit(0);
    }

    //get the reply
    if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == (int)
(sizeof(struct RCPPlus_header) + (payload_len)))
    {
       //position to RCP header
       prcp_hdr = (struct RCPPlus_header*)RX_Buf;

       //check for error message
       if(prcp_hdr->method == 0x03)
       {
          OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
          printf("An error was returned code %02x\n",*ErrorCode);
          return 0;
       }


       sessionID = ntohl(prcp_hdr->session_id);
    }
    else
    {
       printf("Connect to failed\n");
    }

    return sessionID;
}
```

© Copyright 2001-2021 Bosch Security Systems

```
int SendConnectionKeepAlive(SOCKET Socket, WORD ClientID, DWORD
 SessionID)
{
   struct TPKT_header                tpkt_hdr;           //the TPKT
 header
   struct RCPPlus_header             rcp_hdr;            //the RCP
 header
                                                         //inside
 an RX buffer

   //fill out the RCP header
   rcp_hdr.command           = htons( 0xFFC2);           //Command
 'Retrigger'
   rcp_hdr.datatype          =        0x08;              //Datatype
 T_DWORD
   rcp_hdr.rd_wr             =        0x0;               //READ
   rcp_hdr.version           =        0x3;               //Version
 3
   rcp_hdr.method            =        0x00;              //REQUEST
   rcp_hdr.reserved          =        0;
   rcp_hdr.client_id         = htons( ClientID);         //use
 clientID from register reply
   rcp_hdr.session_id        = htonl( SessionID);
   rcp_hdr.num               = htons( 0x0000);           //no num
 parameter needed
   rcp_hdr.len               = 0;                        //no
 payload in READ command

   //fill out TPKT
   tpkt_hdr.version          = 3;                        //TPKT
 version 3
   tpkt_hdr.reserved         = 0;
   //overall length(including TPKT itself)
   tpkt_hdr.size             = htons( sizeof(rcp_hdr) +
 sizeof(tpkt_hdr));

   //send to packet to the network
   if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                 (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                 NULL,                0))
   {
      printf("Send packet error\n");
      exit(0);
   }
```

© Copyright 2001-2021 Bosch Security Systems

```
   return 0;
}



int SendPacket (
                  SOCKET Socket,
                  OCTET *TPKT_hdr,  int TPKT_hdr_len,
                  OCTET *RCP_hdr,   int RCP_hdr_len,
                  OCTET *Payload,   int Payload_len
               )
{
   int err=0;

   printf("\n");

   //send in 3 parts (this is allowed)

   //send the TPKT header
   err += send(Socket, (char*)TPKT_hdr,   TPKT_hdr_len,  0);
   printf("snd TPKT hdr:\n"); view_hex(TPKT_hdr, TPKT_hdr_len);

   //send the RCP header
   err += send(Socket, (char*)RCP_hdr,    RCP_hdr_len,   0);
   printf("snd RCP hdr:\n"); view_hex(RCP_hdr, RCP_hdr_len);

   //send the payload, if one
   if(Payload_len)
   {
      err += send(Socket, (char*)Payload,    Payload_len,   0);
      printf("snd Payload:\n"); view_hex(Payload, Payload_len);
   }

   printf("\n");

   //have all bytes been sent ?
   if (err != TPKT_hdr_len + RCP_hdr_len + Payload_len )
   {
      return -1;
   }
   return 0;
}
```

© Copyright 2001-2021 Bosch Security Systems

```
int GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen)
{
   int len, wait_bytes;
   struct TPKT_header                  *ptpkt_hdr;



   if(RX_BufMaxLen < sizeof(struct TPKT_header))
   {
      return -1;
   }


   //wait for reply

   //wait for the TPKT header to be arrived completely
   do
   {
      //just check; do not empty TCP buffer
      len = recv(Socket, RX_Buf, sizeof(struct TPKT_header),
 MSG_PEEK);

      if(len == SOCKET_ERROR)
      {
         //nothing to do
         return -1;
      }

   } while(len != sizeof(struct TPKT_header));
   //all bytes arrived - empty TCP buffer
   recv(Socket, RX_Buf, sizeof(struct TPKT_header), 0);

   printf("rcv TPKT hdr:\n"); view_hex(RX_Buf, len);


   //Now the length of the entire RCP packet can be determined
   ptpkt_hdr = (struct TPKT_header*)RX_Buf;
   wait_bytes = ntohs(ptpkt_hdr->size) - sizeof(struct TPKT_header);

   if(RX_BufMaxLen < wait_bytes)
   {
```

© Copyright 2001-2021 Bosch Security Systems

```
      return -1;
   }

   do
   {
      //just check; do not empty TCP buffer
      len = recv(Socket, RX_Buf, wait_bytes, MSG_PEEK);
   } while(len != wait_bytes);
   //all bytes arrived - empty TCP buffer
   recv(Socket, RX_Buf, wait_bytes, 0);

   printf("rcv Payload:\n"); view_hex(RX_Buf, len);
   printf("\n");

   return wait_bytes;
}
```

```
void view_hex(void *pv, int l)
{
   //improved display of hex strings
   OCTET *p = (OCTET *)pv;

   int i;

   for (i=0;i<l;i++)
   {
      if(!(i%16) && i) printf("\n");

      printf("%02x ",p[i]);
   }
   printf("\n");
}
```

© Copyright 2001-2021 Bosch Security Systems

3.3 Sample code for connecting to a VideoJet using the TCP protocol

This is a sample application to interface the RCP Plus server inside the VideoJet.

This application provides eight steps:

- Connecting to the RCP server over TCP port 80
- Pass over socket control to the RCP server
- Register at the server
- Send a connect primitive
- Open a TCP socket for receiving video data
- Pass over socket control to the RCP server
- Receives video data
- Send keep-alives to the VideoJet

For compiling the source code we suggest Microsoft Visual C++ V5.00. Other compiler should also work fine except the winsock initialization.

If you are not supplied with the source file, contact support@vcs.com and request request RCPConnect_TCP.c

RCPConnect_TCP.c

```
/*

  RCPConnect_TCP80.c

  Sample RCP Plus Version 3 application

  Tested with VJ400 V6.00, VJ1000 V1.0

  Compilation tested under Microsoft Visual C++ 5.0

  -- make sure wsock32.lib is included in project->settings->linker-
>Object/Library Modules !!! --

  includes a possible output listing at the end of this file

*/


#include <stdio.h>
#include <winsock.h>

#define RCP_NETWORK_PORT         (80) //the HTTP port


//definable IP addresses for this test
```

© Copyright 2001-2021 Bosch Security Systems

```
//the VJ for testing the RCP connection - this should be an video
 encoder
#define REMOTE_VJ_IP            0x0a000001 //is "10.0.0.1"



//password and length
//this is a just a default password; replace this if a password is set
 on the VJ
//if no password ist set, all password will be accepted
#define LOGIN_STRING            "+service:pass+"
#define LOGIN_STRING_LENGTH     14



//some datatypes
typedef unsigned long           DWORD;
typedef unsigned short          WORD;
typedef unsigned char           OCTET;



//structures for the RCP header and commands

//the TPKT header
struct TPKT_header
{
   OCTET    version;
   OCTET    reserved;
   WORD     size;
};


//the RCP header
struct RCPPlus_header
{
   WORD     command;
   OCTET    datatype;
   OCTET    rd_wr    :4, //Note swap these lines on big endian
 machines !!
            version  :4; //Note swap these lines on big endian
 machines !!
   OCTET    method;
   OCTET    reserved;
   WORD     client_id;
   DWORD    session_id;
```

© Copyright 2001-2021 Bosch Security Systems

```
   WORD     num;
   WORD     len;
};


//the fix part of the RCP Register command
struct h_RCP_ClientRegister
{
   OCTET    RegType;
   OCTET    Reserved;
   WORD     ClientID;

   OCTET    PwdEncryption;
   OCTET    PwdLen;
   WORD     NbrOfMessages;

   //followed by
   //WORD   Messages[n];
   //OCTET  Password[nn];
};

//the replay to the RCP register command
struct h_RCP_ClientRegisterReply
{
   OCTET    Result;
   OCTET    Level;
   WORD     ClientID;
};



//structures and defines for requesting video
struct con_prim_header
{
   OCTET    method;
   OCTET    media;
   OCTET    status;
   OCTET    reverse_login;
   DWORD    reserved2;
};


#define METHOD_GET              0x00
#define METHOD_PUT              0x01

#define MEDIA_DIRECTION_TX      0x00
#define MEDIA_DIRECTION_RX      0x01
```

© Copyright 2001-2021 Bosch Security Systems

```
#define MEDIA_CHANNEL_VIDEO      0x01
#define MEDIA_CHANNEL_AUDIO      0x02
#define MEDIA_CHANNEL_DATA       0x03

#define CODING_MPEG4             0x0004
#define CODING_MPEG2             0x0008

#define RESOLUTION_QCIF          0x0001
#define RESOLUTION_CIF           0x0002
#define RESOLUTION_2CIF          0x0004
#define RESOLUTION_4CIF          0x0008
#define RESOLUTION_CUSTOM        0x0010

#define MEP_RTP                  0x01
#define MEP_TCP_OUTBAND          0x04

struct media_desc_video
{
   OCTET    media_encapsulation;
   OCTET    flags;
   WORD     media_port;
   DWORD    media_host;

   OCTET    coder;
   OCTET    line;
   WORD     media_ctrl_port;
   DWORD    media_ctrl_host;

   WORD     coding;
   WORD     resolution;
   WORD     reserved6;
   WORD     bandwidth;
};
```

```
int      SendPacket( SOCKET Socket, OCTET *TPKT_hdr, int TPKT_hdr_len,
 OCTET *RCP_hdr, int RCP_hdr_len,
                     OCTET *Payload, int Payload_len);
int      GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen, int
 doprint);
void     view_hex(void *pv, int l);
WORD     Register(   SOCKET Socket);
DWORD    SendConnectPrimitive(SOCKET Socket, WORD ClientID);
```

© Copyright 2001-2021 Bosch Security Systems

```
int      SendConnectionKeepAlive(SOCKET Socket, WORD ClientID, DWORD
 SessionID);
int      Disconnect(SOCKET Socket, WORD ClientID, DWORD SessionID);
```

```
void main(int argc, char* argv[])
{
   WORD                  wVersionRequested;   //for initial access
 to the winsocks
   WSADATA               wsaData;             //for initial access
 to the winsocks
   SOCKET                Socket;              //socket for the TCP
 connection
   struct sockaddr_in    remote_sin;          //remote address

   WORD                  ClientID;            //to be assgined after
 RCP registration
   DWORD                 SessionID;           //to be assgined after
 RCP 'connect to'

   char                  RX_Buf[2048];        //buffer for data from
 socket
   int                   len;                 //length of a received
 packet
   DWORD                 cnt=0;               //used for a simple
 timeout mechanism
   unsigned long int     nNoBlock=1;          //for setting sockets
 to non-blocking
   SOCKET                VideoDataSocket;     //the video receivinbg
 socket

   OCTET                 PassOverToRCP[] = "GET /rcp_tunnel
 HTTP1.0\r\n\r\n";
   OCTET                 PassVideoDataSocketToRCP[128];
   int                   buffsize = 128 * 1024;



   //init winsock
```

© Copyright 2001-2021 Bosch Security Systems

```
  wVersionRequested = MAKEWORD( 2, 2 );
  if ( WSAStartup( wVersionRequested, &wsaData ) != 0 )
  {
     printf("Error starting winsock\n");
     exit(0);
  }


  //create a socket and connect to the RCP server
  Socket = socket (AF_INET, SOCK_STREAM, 0);
  if (Socket != INVALID_SOCKET)
  {
     remote_sin.sin_family         = AF_INET;
     remote_sin.sin_port           = htons (RCP_NETWORK_PORT);
     remote_sin.sin_addr.s_addr    = htonl (REMOTE_VJ_IP);

     connect(Socket, (struct sockaddr*)&remote_sin, sizeof(struct
sockaddr));
  }
  else
  {
     exit(0);
  }


  //the connect of the the socket goes to the HTTP server on the
VideoJet
  //advice the HTTP server to pass the socket internally to the RCP
server

  send(Socket, PassOverToRCP, strlen(PassOverToRCP), 0);




  printf("=========================================\n");
  printf("\n\nTry to register...\n");
  Sleep(2000);
  ClientID = Register(Socket);
  Sleep(3000);

  printf("=========================================\n");
  printf("\n\nSend connect primitive...\n");

  SessionID = SendConnectPrimitive(Socket, ClientID);

  //make RCP socket non-blocking
  if (SOCKET_ERROR == ioctlsocket(Socket, FIONBIO, &nNoBlock))
```

© Copyright 2001-2021 Bosch Security Systems

```
    {
       exit(0);
    }


    printf("=========================================\n");
    printf("\n\nOpen TCP socket for receiving video data...\n");

    //open a TCP port for receiving the video data
    VideoDataSocket = socket(AF_INET,SOCK_STREAM,0);
    if (VideoDataSocket != INVALID_SOCKET)
    {
       remote_sin.sin_family         = AF_INET;
       remote_sin.sin_port           = htons (RCP_NETWORK_PORT);
       remote_sin.sin_addr.s_addr    = htonl (REMOTE_VJ_IP);

       connect(VideoDataSocket, (struct sockaddr*)&remote_sin,
  sizeof(struct sockaddr));
    }
    else
    {
       exit(0);
    }

    if (SOCKET_ERROR == setsockopt(VideoDataSocket, SOL_SOCKET,
  SO_RCVBUF, (char *)&buffsize, sizeof(buffsize)))
    {
       exit(0);
    }

    //the connect of the the socket goes to the HTTP server on the
  VideoJet
    //advice the HTTP server to pass the socket internally to the RCP
  server
    //signal that this is a video receive socket to the corresponding
  session ID

    sprintf(PassVideoDataSocketToRCP, "GET /media_tunnel/%08lx/
%02x/%02x HTTP 1.0\r\n\r\n", SessionID, MEDIA_CHANNEL_VIDEO,
  MEDIA_DIRECTION_RX);

    if ((int)strlen(PassVideoDataSocketToRCP) != send(VideoDataSocket,
  PassVideoDataSocketToRCP, strlen(PassVideoDataSocketToRCP), 0))
    {
       exit(0);
    }
```

© Copyright 2001-2021 Bosch Security Systems

```
    //make TCP socket non-blocking
    if (SOCKET_ERROR == ioctlsocket(VideoDataSocket, FIONBIO,
  &nNoBlock))
    {
       exit(0);
    }



    while(1)
    {
       cnt++;

       //a very, very simple timeout mechanism; a timeout retrigger
  should be sent every 2-5 seconds
       if(!(cnt%100000))
       {
          SendConnectionKeepAlive(Socket, ClientID, SessionID);
       }


       GetPacket(Socket, RX_Buf, sizeof(RX_Buf), TRUE);
       //don't check responses for the keep-alive at the moment;


       //receive the video data
       len = GetPacket(VideoDataSocket, RX_Buf, sizeof(RX_Buf), FALSE);
       if(len > 0)
       {
          printf("rx %d\n",len);
       }

       Sleep(1);


    }
  }
```

© Copyright 2001-2021 Bosch Security Systems

```
WORD Register(SOCKET Socket)
{
   struct TPKT_header               tpkt_hdr;                    //
the TPKT header
   struct RCPPlus_header            rcp_hdr, *prcp_hdr;          //
the RCP header and a pointer for
                                                                 //
access inside an RX buffer
   struct h_RCP_ClientRegister      *prcp_reg_hdr;               //
pointer to the RCP registration struct
   struct h_RCP_ClientRegisterReply *rcp_reg_reply_hdr;          //
pointer to access the RCP registration
                                                                 //
reply inside a buffer
   WORD                             NbrOfMessages, n;           //
counts the number of messages in the
                                                                 //
regitration command
   WORD                             Messages[]={0xFFC3, 0x01C0}; //
the messages to be registerd
   OCTET                            *Payload , *pw;             //
pointer to access the payload section
   WORD                             PayloadLen, wTPKTsize;      //
length descriptors
   char                             RX_Buf[2048];               //
buffer for data from socket
   WORD                             ClientID;                   //
assigned client IP; to be taken
                                                                 //
from the reply


   //fill out the RCP registration
   PayloadLen = sizeof(struct h_RCP_ClientRegister) + sizeof(Messages)
 + LOGIN_STRING_LENGTH;
   Payload = malloc(PayloadLen);
   if(!Payload)
   {
      return 0;
   }

   //use pw as a write pointer
   pw = Payload;

   //write fix header
   prcp_reg_hdr = (struct h_RCP_ClientRegister*)pw;
   prcp_reg_hdr->RegType         =        0x01;
        //Normal registration
```

© Copyright 2001-2021 Bosch Security Systems

```
  prcp_reg_hdr->Reserved        =        0;
  prcp_reg_hdr->ClientID        = htons( 0x0000);
       //no ClientID in normal

       //registration
  prcp_reg_hdr->PwdEncryption   =        0x00;
       //Plain text password encrytion
  prcp_reg_hdr->PwdLen          =        LOGIN_STRING_LENGTH;
       //Length of password

       //string (without '\0')
  NbrOfMessages                 =        sizeof(Messages) /
sizeof(WORD);
  prcp_reg_hdr->NbrOfMessages   = htons( NbrOfMessages);
       //Nbr of messages that follow
  pw += sizeof(struct h_RCP_ClientRegister);

  //write message array
  for(n=0;n<NbrOfMessages;n++)
  {
     ((WORD*)pw)[n]             = htons( Messages[n]);
       //fill in message requests
  }
  pw+=sizeof(Messages);

  //write password string
  memcpy(pw,                             LOGIN_STRING,
LOGIN_STRING_LENGTH); //fill the password string itself


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF00);
       //Command 'RCP register'
  rcp_hdr.datatype           =        0x0c;
       //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
       //WRITE
  rcp_hdr.version            =        0x3;
       //Version 3
  rcp_hdr.method             =        0x00;
       //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( 0x0000);
       //currently no ClientID available
  rcp_hdr.session_id         = htonl( 0x00000000);
       //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
       //no num parameter needed
```

© Copyright 2001-2021 Bosch Security Systems

```
  rcp_hdr.len                = htons( PayloadLen);
       //length of the payload section

  //fill out TPKT
  tpkt_hdr.version           = 3;
       //TPKT version 3
  tpkt_hdr.reserved          = 0;
  wTPKTsize                  = PayloadLen + sizeof(rcp_hdr) +
sizeof(tpkt_hdr);
  tpkt_hdr.size              = htons( wTPKTsize);
       //overall length

       //(including TPKT itself)



  //send to packet to the network
  if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                               (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                               Payload,             PayloadLen))
  {
     printf("Send packet error\n");
     exit(0);
  }

  //get the reply
  if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf), TRUE)
== sizeof(struct RCPPlus_header) + sizeof(struct
h_RCP_ClientRegisterReply))
  {
     //position to RCP header
     prcp_hdr = (struct RCPPlus_header*)RX_Buf;

     //check for error message
     if(prcp_hdr->method == 0x03)
     {
        OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
        printf("An error was returned code %02x\n",*ErrorCode);
        return 0;
     }

     //position to payload section
     rcp_reg_reply_hdr = (struct h_RCP_ClientRegisterReply *)(RX_Buf
+ sizeof(struct RCPPlus_header));

     //get assigned ClientID
     ClientID = ntohs(rcp_reg_reply_hdr->ClientID);
```

© Copyright 2001-2021 Bosch Security Systems

```
      printf("Register reply:\n");
      printf("Result:     %02x\n",rcp_reg_reply_hdr->Result);
      printf("Level:      %02x\n",rcp_reg_reply_hdr->Level);
      printf("ClientID    %04x\n",ClientID);
   }
   else
   {
      printf("Register failed\n");
   }

   return ClientID;

}
```

```
DWORD SendConnectPrimitive(SOCKET Socket, WORD ClientID)
{

   struct TPKT_header                  tpkt_hdr;
  //the TPKT header
   struct RCPPlus_header               rcp_hdr, *prcp_hdr;
  //the RCP header and a pointer for

  //access inside an RX buffer
   char                                TX_Buf[256], RX_Buf[2048], *po;
       //buffer for data from socket
   struct con_prim_header              *h_req;
   struct media_desc_video             *md_video;
   WORD                                payload_len;

   DWORD                               sessionID=0;


   po = TX_Buf;

   h_req = (struct con_prim_header*)po;

   //fill out the RCP connect primitive
   h_req->method              =  METHOD_GET;
   h_req->media               =  MEDIA_CHANNEL_VIDEO;
```

© Copyright 2001-2021 Bosch Security Systems

```
  h_req->reverse_login       =  FALSE;
  po += sizeof(struct con_prim_header);

  md_video = (struct media_desc_video*)po;
  md_video->media_encapsulation =        MEP_TCP_OUTBAND;
  md_video->media_port          = 0;
  md_video->media_host          = htonl( 0 ); //transmitter shall
take the destination address from the received packet
  md_video->media_ctrl_port     = htons( 0 );
  md_video->media_ctrl_host     = htonl( 0 );
  md_video->coding              = htons( CODING_MPEG4 );
  md_video->resolution          = htons( RESOLUTION_CIF );
  md_video->bandwidth           = htons( 1000 );
  md_video->coder               = 0;
  md_video->line                = 1;
  md_video->flags               = 0;
  po += sizeof(struct media_desc_video);
  payload_len = (WORD)(po - TX_Buf);


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF0C);
 //Command 'RCP connect primitive'
  rcp_hdr.datatype           =        0x0c;
 //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
 //WRITE
  rcp_hdr.version            =        0x3;
 //Version 3
  rcp_hdr.method             =        0x00;
 //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( ClientID);
 //use clientID from register reply
  rcp_hdr.session_id         = htonl( 0x00000000);
 //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
 //no num parameter needed
  rcp_hdr.len                = htons( payload_len);
 //length of the payload section

  //fill out TPKT
  tpkt_hdr.version           = 3;
 //TPKT version 3
  tpkt_hdr.reserved          = 0;
  //overall length(including TPKT itself)
  tpkt_hdr.size              = htons( (WORD)(payload_len +
sizeof(rcp_hdr) + sizeof(tpkt_hdr)));
```

© Copyright 2001-2021 Bosch Security Systems

```
   //send to packet to the network
   if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                    (OCTET*)&rcp_hdr,
 sizeof(rcp_hdr),
                                    TX_Buf,   payload_len))
   {
      printf("Send packet error\n");
      exit(0);
   }

   //get the reply
   if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf), TRUE) == (int)
(sizeof(struct RCPPlus_header) + (payload_len)))
   {
      //position to RCP header
      prcp_hdr = (struct RCPPlus_header*)RX_Buf;

      //check for error message
      if(prcp_hdr->method == 0x03)
      {
         OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
         printf("An error was returned code %02x\n",*ErrorCode);
         return 0;
      }


      sessionID = ntohl(prcp_hdr->session_id);
   }
   else
   {
      printf("Connect to failed\n");
   }

   return sessionID;
}



int SendConnectionKeepAlive(SOCKET Socket, WORD ClientID, DWORD
 SessionID)
{
   struct TPKT_header                tpkt_hdr;           //the TPKT
 header
   struct RCPPlus_header             rcp_hdr;            //the RCP
 header
```

© Copyright 2001-2021 Bosch Security Systems

```
                                                        //inside
  an RX buffer
    //Next task; read out the software version number
    //fill out the RCP header
    rcp_hdr.command             = htons( 0xFFC2);           //Command
  'Retrigger'
    rcp_hdr.datatype            =        0x08;              //Datatype
  T_DWORD
    rcp_hdr.rd_wr               =        0x0;               //READ
    rcp_hdr.version             =        0x3;               //Version
  3
    rcp_hdr.method              =        0x00;              //REQUEST
    rcp_hdr.reserved            =        0;
    rcp_hdr.client_id           = htons( ClientID);         //use
  clientID from register reply
    rcp_hdr.session_id          = htonl( SessionID);
    rcp_hdr.num                 = htons( 0x0000);           //no num
  parameter needed
    rcp_hdr.len                 = 0;                        //no
  payload in READ command

    //fill out TPKT
    tpkt_hdr.version            = 3;                        //TPKT
  version 3
    tpkt_hdr.reserved           = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size               = htons( sizeof(rcp_hdr) +
  sizeof(tpkt_hdr));

    //send to packet to the network
    if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
  sizeof(tpkt_hdr),
                                  (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                  NULL,                0))
    {
       printf("Send packet error\n");
       exit(0);
    }

    return 0;
  }



  int SendPacket (
                    SOCKET Socket,
                    OCTET *TPKT_hdr,  int TPKT_hdr_len,
```

© Copyright 2001-2021 Bosch Security Systems

```
                  OCTET *RCP_hdr,   int RCP_hdr_len,
                  OCTET *Payload,   int Payload_len
               )
{
   int err=0;

   printf("\n");

   //send in 3 parts (this is allowed)

   //send the TPKT header
   err += send(Socket, (char*)TPKT_hdr,   TPKT_hdr_len,  0);
   printf("snd TPKT hdr:\n"); view_hex(TPKT_hdr, TPKT_hdr_len);

   //send the RCP header
   err += send(Socket, (char*)RCP_hdr,    RCP_hdr_len,   0);
   printf("snd RCP hdr:\n"); view_hex(RCP_hdr, RCP_hdr_len);

   //send the payload, if one
   if(Payload_len)
   {
      err += send(Socket, (char*)Payload,    Payload_len,   0);
      printf("snd Payload:\n"); view_hex(Payload, Payload_len);
   }

   printf("\n");

   //have all bytes been sent ?
   if (err != TPKT_hdr_len + RCP_hdr_len + Payload_len )
   {
      return -1;
   }
   return 0;
}
```

```
int GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen, int
 doprint)
{
```

© Copyright 2001-2021 Bosch Security Systems

```
   int len, wait_bytes;
   struct TPKT_header                  *ptpkt_hdr;



   if(RX_BufMaxLen < sizeof(struct TPKT_header))
   {
      return -1;
   }


   //wait for reply

   //wait for the TPKT header to be arrived completely
   do
   {
      //just check; do not empty TCP buffer
      len = recv(Socket, RX_Buf, sizeof(struct TPKT_header),
MSG_PEEK);

      if(len == SOCKET_ERROR)
      {
         //nothing to do
         return -1;
      }

   } while(len != sizeof(struct TPKT_header));
   //all bytes arrived - empty TCP buffer
   recv(Socket, RX_Buf, sizeof(struct TPKT_header), 0);

   if(doprint)
   {
      printf("rcv TPKT hdr:\n"); view_hex(RX_Buf, len);
   }


   //Now the length of the entire RCP packet can be determined
   ptpkt_hdr = (struct TPKT_header*)RX_Buf;
   wait_bytes = ntohs(ptpkt_hdr->size) - sizeof(struct TPKT_header);

   if(RX_BufMaxLen < wait_bytes)
   {
      return -1;
   }

   do
   {
      //just check; do not empty TCP buffer
```

© Copyright 2001-2021 Bosch Security Systems

```
      len = recv(Socket, RX_Buf, wait_bytes, MSG_PEEK);
   } while(len != wait_bytes);
   //all bytes arrived - empty TCP buffer
   recv(Socket, RX_Buf, wait_bytes, 0);

   if(doprint)
   {
      printf("rcv Payload:\n"); view_hex(RX_Buf, len);
      printf("\n");
   }

   return wait_bytes;
}
```

```
void view_hex(void *pv, int l)
{
   //improved display of hex strings
   OCTET *p = (OCTET *)pv;

   int i;

   for (i=0;i<l;i++)
   {
      if(!(i%16) && i) printf("\n");

      printf("%02x ",p[i]);
   }
   printf("\n");
}
```

© Copyright 2001-2021 Bosch Security Systems

3.4 Sample code for connecting to VideoJet’s internal HDD and start replay

This is a sample application to interface the RCP Plus server inside the VideoJet.

This application provides eight steps:

- Connecting to the RCP server over TCP port 80
- Pass over socket control to the RCP server
- Register at the server
- Send a connect primitive
- Open a TCP socket for receiving video data
- Pass over socket control to the RCP server
- Receives video data
- Send keep-alives to the VideoJet

For compiling the source code we suggest Microsoft Visual C++ V5.00. Other compiler should also work fine except the winsock initialization.

If you are not supplied with the source file, contact support@vcs.com and request RCPConnect_REC.c

RCPConnect_REC.c

```
 /*

  RCPConnect_UDP.c

  Sample RCP Plus Version 3 application

  Tested with VJ400 V6.00, VJ1000 V1.0

  Compilation tested under Microsoft Visual C++ 5.0

  -- make sure wsock32.lib is included in project->settings->linker-
>Object/Library Modules !!! --

  includes a possible output listing at the end of this file

*/


#include <stdio.h>
#include <winsock.h>

#define RCP_NETWORK_PORT         (1756)


//definable IP addresses for this test
```

© Copyright 2001-2021 Bosch Security Systems

```
//the VJ for testing the RCP connection - this should be an video
 encoder
#define REMOTE_VJ_IP            0x0a000001 //is "10.0.0.1"



//password and length
//this is a just a default password; replace this if a password is set
 on the VJ
//if no password ist set, all password will be accepted
#define LOGIN_STRING            "+service:pass+"
#define LOGIN_STRING_LENGTH     14


//port for UDP video receiving; this could also be a dynamically
 assignd port from the stack
//see socket binding
#define VIDEO_RX_PORT           1234

//some datatypes
typedef unsigned long           DWORD;
typedef unsigned short          WORD;
typedef unsigned char           OCTET;




//structures for the RCP header and commands

//the TPKT header
struct TPKT_header
{
   OCTET    version;
   OCTET    reserved;
   WORD     size;
};


//the RCP header
struct RCPPlus_header
{
   WORD     command;
   OCTET    datatype;
   OCTET    rd_wr    :4, //Note swap these lines on big endian
 machines !!
            version  :4; //Note swap these lines on big endian
 machines !!
   OCTET    method;
```

© Copyright 2001-2021 Bosch Security Systems

```
    OCTET    reserved;
    WORD     client_id;
    DWORD    session_id;
    WORD     num;
    WORD     len;
};


//the fix part of the RCP Register command
struct h_RCP_ClientRegister
{
    OCTET    RegType;
    OCTET    Reserved;
    WORD     ClientID;

    OCTET    PwdEncryption;
    OCTET    PwdLen;
    WORD     NbrOfMessages;

    //followed by
    //WORD   Messages[n];
    //OCTET  Password[nn];
};

//the replay to the RCP register command
struct h_RCP_ClientRegisterReply
{
    OCTET    Result;
    OCTET    Level;
    WORD     ClientID;
};



//structures and defines for requesting video
struct con_prim_header
{
    OCTET    method;
    OCTET    media;
    OCTET    status;
    OCTET    reverse_login;
    DWORD    reserved2;
};


#define METHOD_GET              0x00
#define METHOD_PUT              0x01
```

© Copyright 2001-2021 Bosch Security Systems

```
#define MEDIA_CHANNEL_VIDEO      0x01
#define MEDIA_CHANNEL_AUDIO      0x02
#define MEDIA_CHANNEL_DATA       0x03

#define CODING_MPEG4             0x0004
#define CODING_MPEG2             0x0008

#define CODING_RECORDED          0x4000

#define RESOLUTION_QCIF          0x0001
#define RESOLUTION_CIF           0x0002
#define RESOLUTION_2CIF          0x0004
#define RESOLUTION_4CIF          0x0008
#define RESOLUTION_CUSTOM        0x0010

#define MEP_RTP                  0x01
#define MEP_TCP_OUTBAND          0x04

struct media_desc_video
{
   OCTET    media_encapsulation;
   OCTET    flags;
   WORD     media_port;
   DWORD    media_host;

   OCTET    coder;
   OCTET    line;
   WORD     media_ctrl_port;
   DWORD    media_ctrl_host;

   WORD     coding;
   WORD     resolution;
   WORD     reserved6;
   WORD     bandwidth;
};



//structures for replay
typedef DWORD SEC_SINCE_2000;

struct HdPartitionFileInfo
{
   SEC_SINCE_2000 secBeg;
   SEC_SINCE_2000 secEnd;
   DWORD          flags;
   DWORD          ID;
```

© Copyright 2001-2021 Bosch Security Systems

```
};


#define CONNECTED                0
#define WAIT_FOR_FILE_LIST       1
#define WAIT_FOR_STOPTIME_ACK    2
#define WAIT_FOR_STARTTIME_ACK   3
#define REPLAY_RUNNING           4



int                              SendPacket( SOCKET Socket, OCTET
 *TPKT_hdr, int TPKT_hdr_len, OCTET *RCP_hdr, int RCP_hdr_len,
                                             OCTET *Payload, int
 Payload_len);
int                              GetPacket(SOCKET Socket, OCTET *RX_Buf,
 int RX_BufMaxLen);
void                             view_hex(void *pv, int l);
WORD                             Register(   SOCKET Socket);
DWORD                            SendConnectPrimitive(SOCKET Socket, WORD
 ClientID, WORD local_UDP_Port);
int                              SendConnectionKeepAlive(SOCKET Socket,
 WORD ClientID, DWORD SessionID);
int                              RequestFileList(SOCKET Socket, WORD
 ClientID);
struct HdPartitionFileInfo *     ReadFileList(OCTET *list, DWORD len);
int                              RequestReplayStoptime(SOCKET Socket,
 WORD ClientID, DWORD SessionID, SEC_SINCE_2000 Stoptime);
int                              RequestReplayStartime(SOCKET Socket,
 WORD ClientID, DWORD SessionID, SEC_SINCE_2000 Starttime);
int                              RequestReplayGo(SOCKET Socket, WORD
 ClientID, DWORD SessionID, int pitch);
int                              WaitForResponse(WORD cmd, OCTET *reply,
 DWORD len);
void                             DisplayCurrentReplayTime(OCTET *RX_Buf,
 DWORD len);




void main(int argc, char* argv[])
{
   WORD                    wVersionRequested;   //for initial access
 to the winsocks
```

© Copyright 2001-2021 Bosch Security Systems

```
  WSADATA                 wsaData;            //for initial access
to the winsocks
  SOCKET                  Socket;             //socket for the TCP
connection
  struct sockaddr_in      remote_sin;         //remote address

  WORD                    ClientID;           //to be assgined after
RCP registration
  DWORD                   SessionID;          //to be assgined after
RCP 'connect to'

  char                    RX_Buf[8096];       //buffer for data from
socket
  int                     len;                //length of a received
packet
  DWORD                   cnt=0;              //used for a simple
timeout mechanism
  unsigned long int       nNoBlock=1;         //for setting sockets
to non-blocking
  struct sockaddr_in      stFrom;             //binding structure
for UDP rx socket
  SOCKET                  VideoDataSocket;    //the video receivinbg
socket
  int                     replay_state;       //status of replay
sequence
  struct HdPartitionFileInfo *p;


  //init winsock
  wVersionRequested = MAKEWORD( 2, 2 );
  if ( WSAStartup( wVersionRequested, &wsaData ) != 0 )
  {
     printf("Error starting winsock\n");
     exit(0);
  }


  //create a socket and connect to the RCP server
  Socket = socket (AF_INET, SOCK_STREAM, 0);
  if (Socket != INVALID_SOCKET)
  {
     remote_sin.sin_family         = AF_INET;
     remote_sin.sin_port           = htons (RCP_NETWORK_PORT);
     remote_sin.sin_addr.s_addr    = htonl (REMOTE_VJ_IP);

     connect(Socket, (struct sockaddr*)&remote_sin, sizeof(struct
sockaddr));
  }
```

© Copyright 2001-2021 Bosch Security Systems

```
    else
    {
       exit(0);
    }

    printf("=========================================\n");
    printf("\n\nTry to register...\n");
    Sleep(2000);
    ClientID = Register(Socket);
    Sleep(3000);

    printf("=========================================\n");
    printf("\n\nOpen UDP socket for receiving video data...\n");

    //open a UDP port for receiving the video data
    VideoDataSocket = socket(AF_INET,SOCK_DGRAM,0);
    if (VideoDataSocket == SOCKET_ERROR)
    {
       exit(0);
    }

    // Bind a receive port to the socket
    stFrom.sin_family       = AF_INET;
    stFrom.sin_port         = htons(VIDEO_RX_PORT);
    stFrom.sin_addr.s_addr  = htonl(INADDR_ANY);
    if (SOCKET_ERROR == bind (VideoDataSocket, (struct sockaddr *)
(&stFrom), sizeof (stFrom)))
    {
       exit(0);
    }

    //make UDP socket non-blocking
    if (SOCKET_ERROR == ioctlsocket(VideoDataSocket, FIONBIO,
 &nNoBlock))
    {
       exit(0);
    }

    printf("=========================================\n");
    printf("Send connect primitive\n");
    SessionID = SendConnectPrimitive(Socket, ClientID, VIDEO_RX_PORT);

    //make RCP socket non-blocking
    if (SOCKET_ERROR == ioctlsocket(Socket, FIONBIO, &nNoBlock))
    {
       exit(0);
    }
```

© Copyright 2001-2021 Bosch Security Systems

```
    replay_state = CONNECTED;


while(1)
   {
      cnt++;

      switch(replay_state)
      {
         case CONNECTED:
            //first of all, request the current file list from the
 VideoJet
            printf("=========================================\n");
            printf("Request file list\n");
            RequestFileList(Socket, ClientID);
            replay_state = WAIT_FOR_FILE_LIST;
            break;

         case WAIT_FOR_FILE_LIST:
            //when a packet arrives,...
            len = GetPacket(Socket, RX_Buf, sizeof(RX_Buf));
            if(len > 0)
            {
               //check wheather this is the reply to the file list
 command,...
               if (WaitForResponse(0x0901, RX_Buf, len))
               {
                  //if so, extrace the first entry,...
                  if ( (p = ReadFileList(RX_Buf, len)) )
                  {
                     //and request setting the stop time for the next
 replay.

 printf("=========================================\n");
                     printf("Set the stop time for replay\n");
                     RequestReplayStoptime(Socket, ClientID,
 SessionID, p->secEnd);
                     replay_state = WAIT_FOR_STOPTIME_ACK;
                  }
               }
            }
            break;

         case WAIT_FOR_STOPTIME_ACK:
            //when a packet arrives,...
            len = GetPacket(Socket, RX_Buf, sizeof(RX_Buf));
            if(len > 0)
            {
```

© Copyright 2001-2021 Bosch Security Systems

```
                //check wheather this is the reply to the stop time
  command,...
                if (WaitForResponse(0x0904, RX_Buf, len))
                {
                   //if so, request setting the start time of the next
  replay.
                   printf("=========================================
\n");
                   printf("Set the start time for replay\n");
                   RequestReplayStartime(Socket, ClientID, SessionID,
  p->secBeg);
                   replay_state = WAIT_FOR_STARTTIME_ACK;
                }
             }
             break;

          case WAIT_FOR_STARTTIME_ACK:
             //when a packet arrives,...
             len = GetPacket(Socket, RX_Buf, sizeof(RX_Buf));
             if(len > 0)
             {
                //check wheather this is the reply to the start time
  command,...
                if (WaitForResponse(0x0905, RX_Buf, len))
                {
                   //if so, request starting the replay with 100% speed
                   printf("=========================================
\n");
                   printf("Start replay with 100%% speed\n");

                   RequestReplayGo(Socket, ClientID, SessionID, 100);
                   //NOTE: the reply to the start command ist not
  checked.

                   replay_state = REPLAY_RUNNING;
                }
             }
             break;

          case REPLAY_RUNNING:

             //a very, very simple timeout mechanism; a timeout
  retrigger should be sent every 2-5 seconds
             if(!(cnt%100))
             {
                SendConnectionKeepAlive(Socket, ClientID, SessionID);
             }
```

© Copyright 2001-2021 Bosch Security Systems

```
            len = GetPacket(Socket, RX_Buf, sizeof(RX_Buf));
            //don't check responses for the keep-alive at the moment;
            if(len > 0)
            {
               //check wheather this is the reply to the start time
 command,...
               if (WaitForResponse(0x0905, RX_Buf, len))
               {
                  DisplayCurrentReplayTime(RX_Buf, len);
               }
            }



            //receive the video data
            len = recvfrom(VideoDataSocket, RX_Buf, sizeof(RX_Buf), 0,
 NULL, NULL);
            if(len > 0)
            {
               printf("rx %d\n",len);
            }

            break;
      }


      Sleep(1);


   }
}
```

```
WORD Register(SOCKET Socket)
{
   struct TPKT_header              tpkt_hdr;                    //
the TPKT header
```

© Copyright 2001-2021 Bosch Security Systems

```
   struct RCPPlus_header            rcp_hdr, *prcp_hdr;           //
the RCP header and a pointer for
                                                                  //
access inside an RX buffer
   struct h_RCP_ClientRegister      *prcp_reg_hdr;                //
pointer to the RCP registration struct
   struct h_RCP_ClientRegisterReply *rcp_reg_reply_hdr;           //
pointer to access the RCP registration
                                                                  //
reply inside a buffer
   WORD                             NbrOfMessages, n;             //
counts the number of messages in the
                                                                  //
regitration command
   WORD                             Messages[]={0xFFC3, 0x0905};  //
the messages to be registerd
   OCTET                            *Payload , *pw;               //
pointer to access the payload section
   WORD                             PayloadLen, wTPKTsize;        //
length descriptors
   char                             RX_Buf[2048];                 //
buffer for data from socket
   WORD                             ClientID;                     //
assigned client IP; to be taken
                                                                  //
from the reply


   //fill out the RCP registration
   PayloadLen = sizeof(struct h_RCP_ClientRegister) + sizeof(Messages)
 + LOGIN_STRING_LENGTH;
   Payload = malloc(PayloadLen);
   if(!Payload)
   {
      return 0;
   }

   //use pw as a write pointer
   pw = Payload;

   //write fix header
   prcp_reg_hdr = (struct h_RCP_ClientRegister*)pw;
   prcp_reg_hdr->RegType         =        0x01;
        //Normal registration
   prcp_reg_hdr->Reserved        =        0;
   prcp_reg_hdr->ClientID        = htons( 0x0000);
        //no ClientID in normal
```

© Copyright 2001-2021 Bosch Security Systems

```
      //registration
  prcp_reg_hdr->PwdEncryption   =        0x00;
      //Plain text password encrytion
  prcp_reg_hdr->PwdLen          =        LOGIN_STRING_LENGTH;
      //Length of password

      //string (without '\0')
  NbrOfMessages                 =        sizeof(Messages) /
sizeof(WORD);
  prcp_reg_hdr->NbrOfMessages   = htons( NbrOfMessages);
      //Nbr of messages that follow
  pw += sizeof(struct h_RCP_ClientRegister);

  //write message array
  for(n=0;n<NbrOfMessages;n++)
  {
     ((WORD*)pw)[n]             = htons( Messages[n]);
      //fill in message requests
  }
  pw+=sizeof(Messages);

  //write password string
  memcpy(pw,                             LOGIN_STRING,
LOGIN_STRING_LENGTH); //fill the password string itself


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF00);
      //Command 'RCP register'
  rcp_hdr.datatype           =        0x0c;
      //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
      //WRITE
  rcp_hdr.version            =        0x3;
      //Version 3
  rcp_hdr.method             =        0x00;
      //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( 0x0000);
      //currently no ClientID available
  rcp_hdr.session_id         = htonl( 0x00000000);
      //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
      //no num parameter needed
  rcp_hdr.len                = htons( PayloadLen);
      //length of the payload section
```

© Copyright 2001-2021 Bosch Security Systems

```
  //fill out TPKT
  tpkt_hdr.version           = 3;
       //TPKT version 3
  tpkt_hdr.reserved          = 0;
  wTPKTsize                  = PayloadLen + sizeof(rcp_hdr) +
sizeof(tpkt_hdr);
  tpkt_hdr.size              = htons( wTPKTsize);
       //overall length

       //(including TPKT itself)



  //send to packet to the network
  if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                               (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                               Payload,             PayloadLen))
  {
     printf("Send packet error\n");
     exit(0);
  }

  //get the reply
  if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == sizeof(struct
RCPPlus_header) + sizeof(struct h_RCP_ClientRegisterReply))
  {
     //position to RCP header
     prcp_hdr = (struct RCPPlus_header*)RX_Buf;

     //check for error message
     if(prcp_hdr->method == 0x03)
     {
        OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
        printf("An error was returned code %02x\n",*ErrorCode);
        return 0;
     }

     //position to payload section
     rcp_reg_reply_hdr = (struct h_RCP_ClientRegisterReply *)(RX_Buf
+ sizeof(struct RCPPlus_header));

     //get assigned ClientID
     ClientID = ntohs(rcp_reg_reply_hdr->ClientID);

     printf("Register reply:\n");
     printf("Result:     %02x\n",rcp_reg_reply_hdr->Result);
     printf("Level:      %02x\n",rcp_reg_reply_hdr->Level);
```

© Copyright 2001-2021 Bosch Security Systems

```
      printf("ClientID    %04x\n",ClientID);
   }
   else
   {
      printf("Register failed\n");
   }

   return ClientID;

}
```

```
DWORD SendConnectPrimitive(SOCKET Socket, WORD ClientID, WORD
 local_UDP_Port)
{

   struct TPKT_header                  tpkt_hdr;
  //the TPKT header
   struct RCPPlus_header               rcp_hdr, *prcp_hdr;
  //the RCP header and a pointer for

  //access inside an RX buffer
   char                                TX_Buf[256], RX_Buf[2048], *po;
  //buffer for data from socket
   struct con_prim_header              *h_req;
   struct media_desc_video             *md_video;
   WORD                                payload_len;

   DWORD                               sessionID=0;


   po = TX_Buf;

   h_req = (struct con_prim_header*)po;

   //fill out the RCP connect primitive
   h_req->method              =  METHOD_GET;
   h_req->media               =  MEDIA_CHANNEL_VIDEO;
   h_req->reverse_login       =  FALSE;
   po += sizeof(struct con_prim_header);
```

© Copyright 2001-2021 Bosch Security Systems

```
  md_video = (struct media_desc_video*)po;
  md_video->media_encapsulation =        MEP_RTP;
  md_video->media_port          = htons( local_UDP_Port );
  md_video->media_host          = htonl( 0 ); //transmitter shall
take the destination address from the received packet
  md_video->media_ctrl_port     = htons( 0 );
  md_video->media_ctrl_host     = htonl( 0 );
  md_video->coding              = htons( CODING_MPEG4 |
CODING_RECORDED);
  md_video->resolution          = htons( RESOLUTION_CIF );
  md_video->bandwidth           = htons( 1000 );
  md_video->coder               = 0;
  md_video->line                = 1;
  md_video->flags               = 0;
  po += sizeof(struct media_desc_video);
  payload_len = (WORD)(po - TX_Buf);


  //fill out the RCP header
  rcp_hdr.command            = htons( 0xFF0C);
 //Command 'RCP connect primitive'
  rcp_hdr.datatype           =        0x0c;
 //Datatype P_OCTET
  rcp_hdr.rd_wr              =        0x1;
 //WRITE
  rcp_hdr.version            =        0x3;
 //Version 3
  rcp_hdr.method             =        0x00;
 //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( ClientID);
 //use clientID from register reply
  rcp_hdr.session_id         = htonl( 0x00000000);
 //currently no SessionID available
  rcp_hdr.num                = htons( 0x0000);
 //no num parameter needed
  rcp_hdr.len                = htons( payload_len);
 //length of the payload section

  //fill out TPKT
  tpkt_hdr.version           = 3;
 //TPKT version 3
  tpkt_hdr.reserved          = 0;
  //overall length(including TPKT itself)
  tpkt_hdr.size              = htons( (WORD)(payload_len +
sizeof(rcp_hdr) + sizeof(tpkt_hdr)));

  //send to packet to the network
```

© Copyright 2001-2021 Bosch Security Systems

```
    if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
  sizeof(tpkt_hdr),
                                     (OCTET*)&rcp_hdr,
  sizeof(rcp_hdr),
                                     TX_Buf,   payload_len))
    {
       printf("Send packet error\n");
       exit(0);
    }

    //get the reply
    if( GetPacket(Socket, RX_Buf, sizeof(RX_Buf)) == (int)
(sizeof(struct RCPPlus_header) + (payload_len)))
    {
       //position to RCP header
       prcp_hdr = (struct RCPPlus_header*)RX_Buf;

       //check for error message
       if(prcp_hdr->method == 0x03)
       {
          OCTET *ErrorCode = RX_Buf + sizeof(struct RCPPlus_header);
          printf("An error was returned code %02x\n",*ErrorCode);
          return 0;
       }

       sessionID = ntohl(prcp_hdr->session_id);
    }
    else
    {
       printf("Connect to failed\n");
    }

    return sessionID;
}



int SendConnectionKeepAlive(SOCKET Socket, WORD ClientID, DWORD
 SessionID)
{
    struct TPKT_header                tpkt_hdr;           //the TPKT
 header
    struct RCPPlus_header             rcp_hdr;            //the RCP
 header
                                                          //inside
 an RX buffer

    //fill out the RCP header
```

© Copyright 2001-2021 Bosch Security Systems

```
    rcp_hdr.command            = htons( 0xFFC2);             //Command
  'Retrigger'
    rcp_hdr.datatype           =        0x08;                //Datatype
  T_DWORD
    rcp_hdr.rd_wr              =        0x0;                 //READ
    rcp_hdr.version            =        0x3;                 //Version
  3
    rcp_hdr.method             =        0x00;                //REQUEST
    rcp_hdr.reserved           =        0;
    rcp_hdr.client_id          = htons( ClientID);           //use
  clientID from register reply
    rcp_hdr.session_id         = htonl( SessionID);
    rcp_hdr.num                = htons( 0x0000);             //no num
  parameter needed
    rcp_hdr.len                = 0;                          //no
  payload in READ command

    //fill out TPKT
    tpkt_hdr.version           = 3;                          //TPKT
  version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(rcp_hdr) +
  sizeof(tpkt_hdr));

    //send to packet to the network
    if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
  sizeof(tpkt_hdr),
                                  (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                  NULL,                0))
    {
       printf("Send packet error\n");
       exit(0);
    }

    return 0;
}



int RequestFileList(SOCKET Socket, WORD ClientID)
{
    struct TPKT_header                tpkt_hdr;           //the TPKT
  header
    struct RCPPlus_header             rcp_hdr;            //the RCP
  header
                                                          //inside
  an RX buffer
```

© Copyright 2001-2021 Bosch Security Systems

```
    //fill out the RCP header
    rcp_hdr.command            = htons( 0x0901);            //Command
  'File info'
    rcp_hdr.datatype           =        0x0c;               //Datatype
  P_OCTET
    rcp_hdr.rd_wr              =        0x0;                //READ
    rcp_hdr.version            =        0x3;                //Version
  3
    rcp_hdr.method             =        0x00;               //REQUEST
    rcp_hdr.reserved           =        0;
    rcp_hdr.client_id          = htons( ClientID);          //use
  clientID from register reply
    rcp_hdr.session_id         = htonl( 0);                 //session
  ID is not needed for this command
    rcp_hdr.num                = htons( 0x0001);            //num
  parameter for partition number
    rcp_hdr.len                = 0;                         //no
  payload in READ command

    //fill out TPKT
    tpkt_hdr.version           = 3;                         //TPKT
  version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(rcp_hdr) +
  sizeof(tpkt_hdr));

    //send to packet to the network
    if( -1 == SendPacket(Socket,  (OCTET*)&tpkt_hdr,
  sizeof(tpkt_hdr),
                                  (OCTET*)&rcp_hdr,    sizeof(rcp_hdr),
                                  NULL,                0))
    {
       printf("Send packet error\n");
       exit(0);
    }

    return 0;

}



struct HdPartitionFileInfo *ReadFileList(OCTET *list, DWORD len)
{
    struct RCPPlus_header *prcp_hdr;
    struct HdPartitionFileInfo *pfileinfo;
```

© Copyright 2001-2021 Bosch Security Systems

```
    DWORD entries;

    //interpret the reply from the file list command, the list is a
  array of type 'struct HdPartitionFileInfo'
    //we do only read the first entry if there is one

    prcp_hdr = (struct RCPPlus_header*)list;

    //we want to replay the first entry

    pfileinfo = (struct HdPartitionFileInfo *) (list + sizeof(struct
  RCPPlus_header));

    entries = (len - sizeof(struct RCPPlus_header)) / sizeof(struct
  HdPartitionFileInfo);

    if(entries > 0)
    {
       pfileinfo->secBeg = ntohl(pfileinfo->secBeg);
       pfileinfo->secEnd = ntohl(pfileinfo->secEnd);
       pfileinfo->flags  = ntohl(pfileinfo->flags);
       pfileinfo->ID     = ntohl(pfileinfo->ID);

       printf("=========================================\n");
       printf("Select first entry from file list\n");
       printf("Recording from %lu ... %lu\n",pfileinfo->secBeg,
  pfileinfo->secEnd);

       return pfileinfo;
    }
    else
    {
       printf("no recording available for replay\n");
       exit(0);
    }


    return NULL;
  }


  int WaitForResponse(WORD cmd, OCTET *reply, DWORD len)
  {
    struct RCPPlus_header *prcp_hdr;

    //check the received packet if this is a vaild reply to a given
  command
```

© Copyright 2001-2021 Bosch Security Systems

```
    prcp_hdr = (struct RCPPlus_header*)reply;

    if( ntohs(prcp_hdr->command) == cmd )
    {
       if(prcp_hdr->method == 0x03)
       {
          printf("error in response\n");
          exit(0);
       }
       else
       {
          return TRUE;
       }
    }

    return FALSE;
}



int RequestReplayStoptime(SOCKET Socket, WORD ClientID, DWORD
 SessionID, SEC_SINCE_2000 Stoptime)
{
    struct TPKT_header                  tpkt_hdr;             //the
 TPKT header
    struct RCPPlus_header               rcp_hdr;              //the RCP
 header
    DWORD                               Payload[2];

    //send the stop time, the first dword carries the stop seconds from
 the file list,
    //the seconds dword is currently not used
    Payload[0] = htonl(Stoptime);
    Payload[1] = htonl(0);


    //fill out the RCP header
    rcp_hdr.command            = htons( 0x0904);
   //Command 'RCP stop time'
    rcp_hdr.datatype           =        0x0c;
   //Datatype P_OCTET
    rcp_hdr.rd_wr              =        0x1;
   //WRITE
    rcp_hdr.version            =        0x3;
   //Version 3
    rcp_hdr.method             =        0x00;
   //REQUEST
```

© Copyright 2001-2021 Bosch Security Systems

```
    rcp_hdr.reserved           =        0;
    rcp_hdr.client_id          = htons( ClientID);
   //use clientID from register reply
    rcp_hdr.session_id         = htonl( SessionID);
    rcp_hdr.num                = htons( 0x0000);
   //no num parameter needed
    rcp_hdr.len                = htons( sizeof(Payload));
   //length of the payload section

    //fill out TPKT
    tpkt_hdr.version           = 3;
   //TPKT version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(Payload) +
 sizeof(rcp_hdr) + sizeof(tpkt_hdr));

    //send to packet to the network
    if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                     (OCTET*)&rcp_hdr,
 sizeof(rcp_hdr),
                                     (OCTET*)&Payload,
 sizeof(Payload)))
    {
       printf("Send packet error\n");
       exit(0);
    }

    return 0;
}


int RequestReplayStartime(SOCKET Socket, WORD ClientID, DWORD
 SessionID, SEC_SINCE_2000 Starttime)
{
    struct TPKT_header                tpkt_hdr;
   //the TPKT header
    struct RCPPlus_header             rcp_hdr;
    DWORD                             Payload[2];

    //send the stop time, the first dword carries the start seconds
 from the file list,
    //the seconds dword is currently not used
    Payload[0] = htonl(Starttime);
    Payload[1] = htonl(0);
```

© Copyright 2001-2021 Bosch Security Systems

```
    //fill out the RCP header
    rcp_hdr.command            = htons( 0x0905);
   //Command 'RCP stop time'
    rcp_hdr.datatype           =        0x0c;
   //Datatype P_OCTET
    rcp_hdr.rd_wr              =        0x1;
   //WRITE
    rcp_hdr.version            =        0x3;
   //Version 3
    rcp_hdr.method             =        0x00;
   //REQUEST
    rcp_hdr.reserved           =        0;
    rcp_hdr.client_id          = htons( ClientID);
   //use clientID from register reply
    rcp_hdr.session_id         = htonl( SessionID);
    rcp_hdr.num                = htons( 0x0000);
   //no num parameter needed
    rcp_hdr.len                = htons( sizeof(Payload));
   //length of the payload section

    //fill out TPKT
    tpkt_hdr.version           = 3;
   //TPKT version 3
    tpkt_hdr.reserved          = 0;
    //overall length(including TPKT itself)
    tpkt_hdr.size              = htons( sizeof(Payload) +
 sizeof(rcp_hdr) + sizeof(tpkt_hdr));

    //send to packet to the network
    if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
 sizeof(tpkt_hdr),
                                     (OCTET*)&rcp_hdr,
 sizeof(rcp_hdr),
                                     (OCTET*)&Payload,
 sizeof(Payload)))
    {
       printf("Send packet error\n");
       exit(0);
    }

    return 0;
}


int RequestReplayGo(SOCKET Socket, WORD ClientID, DWORD SessionID, int
 pitch)
{
```

© Copyright 2001-2021 Bosch Security Systems

```
  struct TPKT_header                  tpkt_hdr;
 //the TPKT header
  struct RCPPlus_header               rcp_hdr;
  int                                 Payload;

  //send the replay start command with the selected speed
  Payload = htonl(pitch);


  //fill out the RCP header
  rcp_hdr.command            = htons( 0x0902);
 //Command 'RCP stop time'
  rcp_hdr.datatype           =        0x04;
 //Datatype T_INT
  rcp_hdr.rd_wr              =        0x1;
 //WRITE
  rcp_hdr.version            =        0x3;
 //Version 3
  rcp_hdr.method             =        0x00;
 //REQUEST
  rcp_hdr.reserved           =        0;
  rcp_hdr.client_id          = htons( ClientID);
 //use clientID from register reply
  rcp_hdr.session_id         = htonl( SessionID);
  rcp_hdr.num                = htons( 0x0000);
 //no num parameter needed
  rcp_hdr.len                = htons( sizeof(Payload));
 //length of the payload section

  //fill out TPKT
  tpkt_hdr.version           = 3;
 //TPKT version 3
  tpkt_hdr.reserved          = 0;
  //overall length(including TPKT itself)
  tpkt_hdr.size              = htons( sizeof(Payload) +
sizeof(rcp_hdr) + sizeof(tpkt_hdr));

  //send to packet to the network
  if ( -1 == SendPacket(Socket,    (OCTET*)&tpkt_hdr,
sizeof(tpkt_hdr),
                                   (OCTET*)&rcp_hdr,
sizeof(rcp_hdr),
                                   (OCTET*)&Payload,
sizeof(Payload)))
  {
     printf("Send packet error\n");
     exit(0);
  }
```

© Copyright 2001-2021 Bosch Security Systems

```
   return 0;
}

void DisplayCurrentReplayTime(OCTET *RX_Buf, DWORD len)
{
   DWORD *p = (DWORD*)(RX_Buf + sizeof(struct RCPPlus_header));

   *p = ntohl(*p);

   printf("Current replay at %lu\n", *p);

}




int SendPacket (
                  SOCKET Socket,
                  OCTET *TPKT_hdr,  int TPKT_hdr_len,
                  OCTET *RCP_hdr,   int RCP_hdr_len,
                  OCTET *Payload,   int Payload_len
               )
{
   int err=0;

   printf("\n");

   //send in 3 parts (this is allowed)

   //send the TPKT header
   err += send(Socket, (char*)TPKT_hdr,   TPKT_hdr_len,  0);
   printf("snd TPKT hdr:\n"); view_hex(TPKT_hdr, TPKT_hdr_len);

   //send the RCP header
   err += send(Socket, (char*)RCP_hdr,    RCP_hdr_len,   0);
   printf("snd RCP hdr:\n"); view_hex(RCP_hdr, RCP_hdr_len);

   //send the payload, if one
   if(Payload_len)
   {
      err += send(Socket, (char*)Payload,    Payload_len,   0);
      printf("snd Payload:\n"); view_hex(Payload, Payload_len);
   }

   printf("\n");
```

© Copyright 2001-2021 Bosch Security Systems

```
   //have all bytes been sent ?
   if (err != TPKT_hdr_len + RCP_hdr_len + Payload_len )
   {
      return -1;
   }
   return 0;
}
```

```
int GetPacket(SOCKET Socket, OCTET *RX_Buf, int RX_BufMaxLen)
{
   int len, wait_bytes;
   struct TPKT_header                  *ptpkt_hdr;


   if(RX_BufMaxLen < sizeof(struct TPKT_header))
   {
      return -1;
   }


   //wait for reply

   //wait for the TPKT header to be arrived completely
   do
   {
      //just check; do not empty TCP buffer
      len = recv(Socket, RX_Buf, sizeof(struct TPKT_header),
 MSG_PEEK);

      if(len == SOCKET_ERROR)
      {
         //nothing to do
         return -1;
      }

   } while(len != sizeof(struct TPKT_header));
```

© Copyright 2001-2021 Bosch Security Systems

```
    //all bytes arrived - empty TCP buffer
    recv(Socket, RX_Buf, sizeof(struct TPKT_header), 0);

    printf("rcv TPKT hdr:\n"); view_hex(RX_Buf, len);


    //Now the length of the entire RCP packet can be determined
    ptpkt_hdr = (struct TPKT_header*)RX_Buf;
    wait_bytes = ntohs(ptpkt_hdr->size) - sizeof(struct TPKT_header);

    if(RX_BufMaxLen < wait_bytes)
    {
       printf("pkt len exceeds\n");
       return -1;
    }

    do
    {
       //just check; do not empty TCP buffer
       len = recv(Socket, RX_Buf, wait_bytes, MSG_PEEK);
    } while(len != wait_bytes);
    //all bytes arrived - empty TCP buffer
    recv(Socket, RX_Buf, wait_bytes, 0);

    printf("rcv Payload:\n"); view_hex(RX_Buf, len);
    printf("\n");

    return wait_bytes;
}









void view_hex(void *pv, int l)
{
    //improved display of hex strings
    OCTET *p = (OCTET *)pv;

    int i;

    for (i=0;i<l;i++)
```

© Copyright 2001-2021 Bosch Security Systems

```
    {
       if(!(i%16) && i) printf("\n");

       printf("%02x ",p[i]);
    }
    printf("\n");
}
```

© Copyright 2001-2021 Bosch Security Systems

3.5 Starting a replay service on the VCS VideoJet using the VCS MPEG-Active-X

The VCS Active-X control is a software which provides

- RCP+ registration at the VideoJet
- The establishing of video, audio and data channels to the VideoJet
- A network connection to the VideoJets for sending and receiving RCP+ commands

The VCS Active-X has its own API. Please also refer to this documentation. The VCS Active-X is not essential for starting a reply service; all commands can be send using a socket communication (see sample code in appendix).

The VCS VideoJets with the HDD option are capable to record live video and audio stream to their internal HDD. The recording is scheduled by the build-in HDD recording manager which can be configured using the Webinterface. Normally the replay can also be started using the Webinterface, or alternatively by the VCS VIDOS software.

This document describes the scenario for starting a replay service step by step.

Enable the reception of RCP messages via the ActiveX.

```
m_mpeg.EnableRCPEvents(short sEnable);
```

with sEnable = 1.

Establish a RCP connection to the VideoJet for VCS MPEG Active-X.

```
m_mpeg.RegisterDeviceEx(LPCTSTR strDestination, short sClient, LPCTSTR
 strUserPass);
```

with strDestination being the IP address of the requested unit. Set sClient = 0 and strUserPass = NULL as long as the referenced unit is not password-protected.

The response of the unit will be forwarded via the event method.

```
RCPStateEx(short sState, short sClient);
```

If the registration succeeded, sState will be set to 4 and sClient specifies the client ID provided by the remote unit. This client ID has to be used subsequently in all RCP+ commands.

Connect the VCS MPEG Active-X with the internal HDD of the VideoJet

```
m_mpeg.ConnectVideoEx(short type, short bandwidth, short resolution,
 short line, bool forceUnicast)
```

with

- type = 0x80FF to accept both MPEG2 and MPEG4 recordings
- bandwith has be unequal to 0, but is otherwise not evaluated

© Copyright 2001-2021 Bosch Security Systems

- resolution = 0xFFFF to accept any resolution
- line specifies the harddisk partition number to be accessed
- forceUnicast should be set to true.

The state of the video connection will be forwarded via event method

```
VideoState(short sState, long session);
```

If the connection succeeded, sState will be set to 2 and ssession specifies the session ID provided by the remote unit. This session ID has to be used subsequently in all RCP+ commands referencing this connection.

In the callback reply from the command, a session ID, assigned by the VideoJet will be retrieved. Using this session ID, all replay related commands will affect this connection to the HDD of the VideoJet.

To retrieve the current available recordings on the VideoJet, a list can be loaded from the HDD.

The RCP command is HD_PARTITION_FILE_INFO (0x0901):

```
unsigned short clientID;
unsigned char cmd[64];
cmd[0] = 0x09;
cmd[1] = 0x01;
cmd[2] = 0x0c;
cmd[3] = 0x03;
cmd[4] = 0x00;
cmd[5] = 0x00;
cmd[6] = (clientID & 0xFF00) >> 8;
cmd[7] = (clientID & 0x00FF);
cmd[8] = 0x00;
cmd[9] = 0x00;
cmd[10] = 0x00;
cmd[11] = 0x00;
cmd[12] = 0x00;
cmd[13] = 0x01;
cmd[14] = 0x00;
cmd[15] = 0x00;
```

Any RCP commands will be sent via

```
m_mpeg.SendRCP(VARIANT rcp);
```

with rcp being a VARIANT representation of the RCP command. From the above byte array the VARIANT can be built as follows:

```
VARIANT* pVar = 0;
SAFEARRAY FAR* psa;
SAFEARRAYBOUND rgsabound[1];

rgsabound[0].lLbound = 0;
```

© Copyright 2001-2021 Bosch Security Systems

```
rgsabound[0].cElements = size; //the number of bytes to transmit (16
 in this example)
psa = SafeArrayCreate(VT_UI1, 1, rgsabound);
if(psa != NULL)
{
 char* psafeData;
 if(SafeArrayAccessData(psa,(void HUGEP* FAR*)&psafeData) == S_OK)
 {
    memcpy(psafeData,cmd,size);
    SafeArrayUnaccessData(psa);

    pVar = new VARIANT;
    if(pVar)
    {
     VariantInit(pVar);
     pVar->vt = VT_ARRAY | VT_UI1;
     pVar->parray = psa;

     //send the command
     m_mpeg.SendRCP(*pVar);

     //free the variant
     VariantClear(pVar);
     delete pVar;
    }
 }
}
```

Any RCP responses from the unit will be forwarded via event method

```
ReceivedRCP(VARIANT rcp)
```

with rcp being a VARIANT representation of the received response. The contained byte array can be extracted as follows:

```
long i;
SAFEARRAY FAR* psa = NULL;
BYTE * pbytes;
HRESULT hr;
LONG cElements, lLBound, lUBound;
VARIANT* pVariant;

// Type check VARIANT parameter. It should contain a BSTR array
// passed by reference. The array must be passed by reference it is
// an in-out-parameter.
if (RCP.vt == (VT_ARRAY | VT_UI1))
```

© Copyright 2001-2021 Bosch Security Systems

```
{
 psa = RCP.parray;
 if(psa == NULL)
      return FALSE;
 // Check dimensions of the array.
 if (SafeArrayGetDim(psa) != 1)
    return FALSE;

 // Get array bounds.
 hr = SafeArrayGetLBound(psa, 1, &lLBound);
 if (FAILED(hr) || lLBound != 0)
    return FALSE;
 hr = SafeArrayGetUBound(psa, 1, &lUBound);
 if (FAILED(hr))
    return FALSE;

 // Get a pointer to the elements of the array.
 hr = SafeArrayAccessData(psa, (void HUGEP* FAR*)&pbytes);
 if (FAILED(hr))
      return FALSE;

 cElements = lUBound-lLBound+1;

 BYTE * pMess = new BYTE[cElements];
 if(pMess)
 {
    for (i = 0; i < cElements; i++)
    {
       *(pMess+i) = pbytes[i];
    }

    SafeArrayUnaccessData(psa);

    //here process the received byte array contained in pMess
    …

    // free the byte array
    delete[] pMess;
 }
```

For the above command the VideoJet will return a list of all current recordings. The structure of this list follows this data structure:

```
struct HdPartitionFileInfo
{
 DWORD secBeg;
```

© Copyright 2001-2021 Bosch Security Systems

```
 DWORD secEnd;
 DWORD flags;
 DWORD ID;
};
```

secBeg: Start of the recording (counted in seconds since the 1.1.2000 00:00)

secEnd: End of the recording (counted in seconds since the 1.1.2000 00:00)

flags:

	Mask	**Name**	**Description**
Bit 29	0x20000000	FLAG_TIME_ZONE_SIGN	The time zone sign of this file
Bit 21	0x00200000	FLAG_VIDEO_MUX	when set, multiple cameras are recorded in this file
Bit 7	0x00000080	FLAG_ALARM_RECORDING 0x00000080 when set, this is a alarm recording	
Bit 6	0x00000040	FLAG_TIME_RECORDING 0x00000040 when set, this is a time recording	
Bit 5	0x00000020	FLAG_VIDEO_LOSS 0x00000020 when set, a video loss alarm has been asserted since the beginning of this recording	
Bit 3	0x00000008	FLAG_ALARM_MOTION	When set, a motion alarm has been asserted since the beginning of this recording
Bit 2	0x00000004	FLAG_ALARM_INPUT	When set, a input alarm has been asserted since the beginning of this recording
Bit 1	0x00000002	FLAG_OVERWRITING	When set, the recording of this file is currently overwriting a previous recording
Bit 0	0x00000001	FLAG_RECORDING	when set, the recording of this file is currently running

Combined Values

Mask	**Name**	**Description**
0x1F800000	MASK_TIME_ZONE_QH	the time zone offset in quarter hours of this file

ID: for internal use only

The total number of recordings can be calculated by the size of the reply divided by the size of the structure (16).

The application can now decide which file to replay. In addition to that, the VideoJets are capable only to replay a specified period of time.

To select the period (or the entire file), the RCP command HD_REPLAY_STOP_TIME (0x0904) will set the end marker for the replay and the command HD_REPLAY_SEEK_TIME (0x0905) will set the start marker. The time is specified by the same format as in the list (counted in seconds since 1.1.2000 00:00). There is seconds DWORD for future extensions. The structure for setting the start and stop time is:

© Copyright 2001-2021 Bosch Security Systems

```
struct TimeSet
{
 DWORD sec;
 DWORD unused;
};
```

Now set the stop marker:

```
unsigned long sessionID, StopTime;
unsigned short clientID;
unsigned char cmd[64];

cmd[0] = 0x09;
cmd[1] = 0x04;
cmd[2] = 0x0c;
cmd[3] = 0x13;
cmd[4] = 0x00;
cmd[5] = 0x00;
cmd[6] = (clientID & 0xFF00) >> 8;
cmd[7] = (clientID & 0x00FF);
cmd[8] = (sessiondID & 0xFF000000) >> 24;
cmd[9] = (sessiondID & 0x00FF0000) >> 16;
cmd[10] = (sessiondID & 0x0000FF00) >> 8;
cmd[11] = (sessiondID & 0x000000FF);
cmd[12] = 0x00;
cmd[13] = 0x00;
cmd[14] = 0x00;
cmd[15] = 0x08;

cmd[16] = (StopTime & 0xFF000000) >> 24;
cmd[17] = (StopTime & 0x00FF0000) >> 16;
cmd[18] = (StopTime & 0x0000FF00) >> 8;
cmd[19] = (StopTime & 0x000000FF);
cmd[20] = 0x00;
cmd[21] = 0x00;
cmd[22] = 0x00;
cmd[23] = 0x00;
```

Transfer the command into a VARIANT and send it as described above. A corresponding response via ReceivedRCP() should be waited for.

The send the start marker:

© Copyright 2001-2021 Bosch Security Systems

```
unsigned long sessionID, StartTime;
unsigned short clientID;
unsigned char cmd[64];

cmd[0] = 0x09;
cmd[1] = 0x05;
cmd[2] = 0x0c;
cmd[3] = 0x13;
cmd[4] = 0x00;
cmd[5] = 0x00;
cmd[6] = (clientID & 0xFF00) >> 8;
cmd[7] = (clientID & 0x00FF);
cmd[8] = (sessiondID & 0xFF000000) >> 24;
cmd[9] = (sessiondID & 0x00FF0000) >> 16;
cmd[10] = (sessiondID & 0x0000FF00) >> 8;
cmd[11] = (sessiondID & 0x000000FF);
cmd[12] = 0x00;
cmd[13] = 0x00;
cmd[14] = 0x00;
cmd[15] = 0x08;

cmd[16] = (StartTime & 0xFF000000) >> 24;
cmd[17] = (StartTime & 0x00FF0000) >> 16;
cmd[18] = (StartTime & 0x0000FF00) >> 8;
cmd[19] = (StartTime & 0x000000FF);
cmd[20] = 0x00;
cmd[21] = 0x00;
cmd[22] = 0x00;
cmd[23] = 0x00;
```

Transfer the command into a VARIANT and send it as described above. A corresponding response via ReceivedRCP() should be waited for.

Now the replay can be started by sending RCP command HD_REPLAY_START (0x0902). The integer parameter specifies the speed of the replay. Normally 100% is live speed, slow motion is less than 100% and fast forward is more than 100%. For backup purposes, the speed can be increased up to 99999. this signals the VideoJet to send the recorded data as fast as possible. A value of 0% will pause the replay.

```
int Speed=100;
unsigned long sessionID;
unsigned short clientID;
unsigned char cmd[64];

cmd[0] = 0x09;
cmd[1] = 0x02;
cmd[2] = 0x04;
cmd[3] = 0x13;
```

© Copyright 2001-2021 Bosch Security Systems

```
cmd[4] = 0x00;
cmd[5] = 0x00;
cmd[6] = (clientID & 0xFF00) >> 8;
cmd[7] = (clientID & 0x00FF);
cmd[8] = (sessiondID & 0xFF000000) >> 24;
cmd[9] = (sessiondID & 0x00FF0000) >> 16;
cmd[10] = (sessiondID & 0x0000FF00) >> 8;
cmd[11] = (sessiondID & 0x000000FF);
cmd[12] = 0x00;
cmd[13] = 0x00;
cmd[14] = 0x00;
cmd[15] = 0x04;

cmd[16] = (Speed & 0xFF000000) >> 24;
cmd[17] = (Speed & 0x00FF0000) >> 16;
cmd[18] = (Speed & 0x0000FF00) >> 8;
cmd[19] = (Speed & 0x000000FF);
```

Transfer the command into a VARIANT and send it as described above. A corresponding response via ReceivedRCP() should be waited for.

To jump to a certain time, the HD_REPLAY_SEEK_TIME (0x0905) can also be used during a replay service. In paused mode, the VideoJet will perform jumps to next frames or previous intra frames. The command for this is HD_REPLAY_SEEK_IFRAME (0x0907). The integer parameter specifies the number of frames to go. Positive values will lead to the next frames; negative values direct to previous intra frames. Due to the limitation of compressed video data, backward skips will lead to intra frames only. The intra frame distance is 5 seconds on VideoJet X00 and is scalable on VideoJet X000. To terminate a replay, the RCP command HD_REPLAY_STOP (0x0903) may be send.

```
unsigned long sessionID;
unsigned short clientID;
unsigned char cmd[64];

cmd[0] = 0x09;
cmd[1] = 0x03;
cmd[2] = 0x00;
cmd[3] = 0x31;
cmd[4] = 0x00;
cmd[5] = 0x00;
cmd[6] = (clientID & 0xFF00) >> 8;
cmd[7] = (clientID & 0x00FF);
cmd[8] = (sessiondID & 0xFF000000) >> 24;
cmd[9] = (sessiondID & 0x00FF0000) >> 16;
cmd[10] = (sessiondID & 0x0000FF00) >> 8;
cmd[11] = (sessiondID & 0x000000FF);
cmd[12] = 0x00;
cmd[13] = 0x00;
```

© Copyright 2001-2021 Bosch Security Systems

```
cmd[14] = 0x00;
cmd[15] = 0x01;

cmd[16] = 0x01;
```

Transfer the command into a VARIANT and send it as described above. A corresponding response via ReceivedRCP() should be waited for. An alternative is to disconnect the connection by calling

```
m_mpeg.DisconnectVideo();
```

During the replay certain messages are generated by the VideoJet.

Signals the current position inside the recording. This may be displayed by a slider positions in the GUI.

Signals, that the current recording has changed its duration (normally every second). The payload of this message has the same format as the file list, except, that there is only one line (the current recording)

Signals that there is an updated file list available (e.g. a new alarm recording has been started). The application may reload the file list again

© Copyright 2001-2021 Bosch Security Systems

3.6 Bicom Command Access Levels

The default protection level for BICOM / AUX commands is 'bl_user' for get-operations (get, getMax, getMin) and 'bl_service' for all other operations. All BICOM / AUX commands which have a protection level different from the default are listed in the following tables. 'bl_priv' means that the command is not accessible at all.

© Copyright 2001-2021 Bosch Security Systems

3.7 Specific error codes

Each rcp command can generate an generic RCP fault as described in the chapter "Rcp Protocol Procedure". Additional command specific error codes are defined. If an command specific error occurs the Response is 2 bytes long. The first byte is set to RCP_ERROR_COMMAND_SPECIFIC (0xf0), the second byte is set to the specific fault.

Storage specific error codes

Span errors

SPAN_ERROR_INV_SPN_IDX	0x02
SPAN_ERROR_INV_SPN_IDX	0x02
SPAN_ERROR_INV_HDR_TYPE	0x03
SPAN_ERROR_INV_ADDR_LIST	0x04
SPAN_ERROR_NOT_MNTD	0x05
SPAN_ERROR_INV_FS	0x06
SPAN_ERROR_INV_LUN_NFO	0x07
SPAN_ERROR_BAD_HDR_CKSM	0x08
SPAN_ERROR_INV_IDX	0x09
SPAN_ERROR_RD_ONLY	0x0a
SPAN_ERROR_NO_REC_DAT	0x0b
SPAN_ERROR_INV_PART_NFO	0x0c
SPAN_ERROR_CONV_REC	0x0d
SPAN_ERROR_SPAN_REQUEST_FAILED	0x0e
SPAN_ERROR_SPAN_REQUEST_RETENTION_TIME	0x0f
SPAN_ERROR_REMOUNT_REFUSED	0x10
SPAN_ERROR_OLD_LUN_NFO	0x20

HDD PMM errors

HD_PMM_ERROR_TIMEOUT	0x12
HD_PMM_ERROR_CREATE_FAILED	0x22
HD_PMM_ERROR_ACCESS_DENIED	0x32
HD_PMM_ERROR_DEVICE_PRESENT_TIMEOUT	0x42
HD_PMM_ERROR_LUN_LOCK	0x52
HD_PMM_ERROR_INVALID_ACCESS	0x62
HD_PMM_ERROR_LUN_MGMT_FILE_NOT_FOUND	0x72
HD_PMM_ERROR_LUN_WRITE_PROTECTED	0x82
HD_PMM_ERROR_COMMON	0xf2

ISCSI errors

ISCSI_ERR_CONNECT	0x31
ISCSI_ERR_INV_LUN	0x33
ISCSI_ERR_LOGIN	0x34
ISCSI_ERR_INV_TARG_IDX	0x35
ISCSI_ERR_PWD	0x36
ISCSI_ERR_PROTO	0x37
ISCSI_ERR_TARG_NOT_REACH	0x38

© Copyright 2001-2021 Bosch Security Systems

ISCSI_ERR_NO_MEMORY	0x3a
ISCSI_ERR_SESS_CREATE	0x3b
ISCSI_ERR_INV_PARAMS	0x3c
ISCSI_ERR_SESS_NOT_FOUND	0x3d
ISCSI_ERR_DISCONN	0x3e
ISCSI_ERR_TIMEOUT	0x3f
ISCSI_ERR_TARGET_NOT_SUPP	0x40
ISCSI_ERR_TARGET_SESSION_LIMIT	0x41
ISCSI_ERR_CMD_NOT_SUPP	0x42
ISCSI_ERR_TARGET_NOT_FOUND	0x43
ISCSI_ERR_SOCK	0x5f
ISCSI_ERR_TARG_PM	0x6f
ISCSI_SOCK_CLOSED	0x7f
ISCSI_ERR_TCP_CONN_RESET	0x8f
ISCSI_ERR_INTR_NOT_SUPP	0x9f
ISCSI_ERR_IP_ZERO	0xa0
ISCSI_ERR_OUT_OF_RESOURCES	0xa1

Forensic search errors

ForensicSearchErrorScriptTooLong	0x01
ForensicSearchErrorInternal	0x02
ForensicSearchErrorNoRuleSelected	0x03
ForensicSearchErrorREConfiguration	0x04
ForensicSearchErrorSyntax	0x05
ForensicSearchErrorMemory	0x06

© Copyright 2001-2021 Bosch Security Systems