Vitec Devices Remote Control API

Overview

HTTP API provides an easy and efficient way of remotely control one or several MGW Ace Encoder appliances using HTTP commands.

This document describes how to control and obtain status information when using the VITEC HTTP API v1.

The required commands as well as parameter values are fully detailed.

Commands can be triggered and executed by simultaneously using both HTTP –based interface and the Web GUI application.

During the implementation of a customer high level software, the Web GUI application can be used to verify that commands are properly applied.

The Web GUI application can be used also to check what are the available parameter values that can be utilized.

Warning: Warning: An attempt to use commands to modify HEVC or H.264 channels encoding parameters while the channels are streaming, will cause the corresponded channels to stop and will have to be manually restarted.

In this document [RO] means Read only parameter.

Vitec WebApi credentials are:

- Login: remote
- Password: as configured in the WebUI (default is 1qaz!QAZ)

Allowed Transport Layer security versions: TLSv1.1 and TLSv1.2

Example of use with curl :

start channel 1

```
curl --anyauth -u username:password -k -X PUT -d '{"state":"play"}' https://192.168.1.1/api/v1/channels/1
```

stop channel 1

```
curl --anyauth -u username:password -k -X PUT -d '{"state":"stop"}' https://192.168.1.1/api/v1/channels/1
```

change bitrate of channel 1

```
curl --anyauth -u username:password -k -X PUT -d '{"streamBitrate":2000000}' https://192.168.1.1/api/v1/channels/1/processing
```

get processing configuration of channel 1

```
curl --anyauth -u username:password -k -X GET "https://192.168.1.1/api/v1
/channels/1/processing"
```

set rateControl on processing

```
curl --anyauth -u username:password -k -X PUT -d '{"rateControl":"vbr"}'
https://192.168.1.1/api/v1/channels/1/processing
```

get list of channels

```
curl --anyauth -u username:password -k -X GET "https://192.168.1.1/api/v1
/channels"
```

get targets configuration of channel 1

```
curl --anyauth -u username:password -k -X GET "https://192.168.1.1/api/v1
/channels/1/targets"
```

get first target configuration of channel 1

```
curl --anyauth -u username:password -k -X GET "https://192.168.1.1/api/v1
/channels/1/targets/1"
```

set streaming address of first target of channel 1

```
curl --anyauth -u username:password -k -X PUT -d
'{"address":"192.168.0.112"}' https://192.168.1.1/api/v1/channels
/1/targets/1
```

set network configuration for Ethernet 1

```
curl --anyauth -u username:password -k -X PUT -d
'{"netmask":"255.255.255.0", "ipaddress": "192.168.0.5", "gateway":
"192.168.0.10"} "https://10.0.0.64/api/v1/networks/1"
```

Example of use with python :

```
#!/usr/bin/env python

import requests # $ python -m pip install requests

import json

from requests.auth import HTTPDigestAuth

url = 'https://192.168.1.1/api/v1/channels/1/targets'

r = requests.get(url, auth=HTTPDigestAuth("username", "password"),
verify=False)

print(r.text)

r = requests.put(url, auth=HTTPDigestAuth("username", "password"),
```

> *verify=False, data = json.dumps({'state':'stop'}))*
>
> *print(r.text)*

Example of use with Powershell :

> *$url = "https://192.168.1.1/api/v1/channels/1/targets"*
>
> *$web = New-Object Net.WebClient*
>
> *$web.Credentials = New-Object System.Net.NetworkCredential("username","password")*
>
> *[System.Net.ServicePointManager]::ServerCertificateValidationCallback = { $true }*
>
> *[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12*
>
> *$web.DownloadString($url)*

Version information

Version : 1

URI scheme

BasePath : /v1 *Schemes* : HTTPS

Produces

- `application/json`

Paths

Get the list of channels

```
GET /channels
```

Responses

HTTP Code	Description	Schema
200	An array of channels	< Channel > array
default	Unexpected error	Error

Get the channel data

```
GET /channels/{channelId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	

Responses

HTTP Code	Description	Schema
200	A channel	Channel
default	Invalid request	Error

Set the channel data

```
PUT /channels/{channelId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	
Body	**channelParameters** *required*		Channel	

Responses

HTTP Code	Description	Schema
default	Channel parameters result	Error

Get the list of inputs for this channel

```
GET /channels/{channelId}/inputs
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	

Responses

HTTP Code	Description	Schema
200	An array of inputs	< Input > array

Get the input data

```
GET /channels/{channelId}/inputs/{inputId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	
Path	**inputId** *required*	Id of input.	number	

Responses

HTTP Code	Description	Schema
200	Get the input data	Input

This ressources contains the processing settings

```
GET /channels/{channelId}/processing
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	

Responses

HTTP Code	Description	Schema
200	Get the processing data	< Processing > array

This ressource change the processing settings

```
PUT /channels/{channelId}/processing
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	
Body	**processingParameters** *required*		< Processing > array	

Responses

HTTP Code	Description	Schema
default	Processing data result	< Error > array

Get the list of targets for this channel

```
GET /channels/{channelId}/targets
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	

Responses

HTTP Code	Description	Schema
200	An array of targets	< Target > array

Get the target data

```
GET /channels/{channelId}/targets/{targetId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	
Path	**targetId** *required*	Id of input.	number	

Responses

HTTP Code	Description	Schema
200	Get the target data	Target

Set the target data

```
PUT /channels/{channelId}/targets/{targetId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**channelId** *required*	Id of channel.	number	
Path	**targetId** *required*	Id of input.	number	
Body	**targetParameters** *required*		< Target > array	

Responses

HTTP Code	Description	Schema
default	target data result	< Error > array

Get the list of networks

```
GET /networks
```

Responses

HTTP Code	Description	Schema
200	An array of networks	< Network > array
default	Unexpected error	Error

Get the Network Interface data

```
GET /networks/{interfaceId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**interfaceId** *required*	Id of network interface.	number	

Responses

HTTP Code	Description	Schema
200	A Network Interface	Network
default	Invalid request	Error

Set the Network Interface data

```
PUT /networks/{interfaceId}
```

Parameters

Type	Name	Description	Schema	Default
Path	**interfaceId** *required*	Id of network interface.	number	
Body	**networkParameters** *required*		Network	

Responses

HTTP Code	Description	Schema
default	Network parameters result	Error

Definitions

Channel

Name	Description	Schema
channelId *optional*	Unique identifier representing a channel [RO]	number
name *optional*	Channel name [RO]	enum (1-HEVC, 2-H.264)
state *optional*	Channel state	enum (Stopped, Playing, Error)

Error

Name	Description	Schema
code *optional*		integer(int32)
message *optional*		string

Input

Name	Description	Schema
format *optional*	Input format. Some formats are detected (SDI resolution, ...) while other can be set (Analog audio sampling, ...)	string
inputId *optional*	Unique identifier representing an input.	number
source *optional*	The source for this input. Refer to the integration manual to have the list of possible sources for the different inputs	string
type *optional*	Input type.	enum (video, audio, klv)

Network

Name	Description	Schema
gateway *optional*	Network gateway	enum (192.168.1.10)
ipaddress *optional*	Network address	enum (192.168.1.1)
netmask *optional*	Network netmask	enum (255.255.255.0)
networkId *optional*	Unique identifier representing a network [RO]	number

Processing

Name	Description	Schema
aesKey *optional*	Encryption key	enum (00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00, FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF)
audioBitrate *optional*	Average audio bitrate in	number

Name	Description	Schema
	Kpbs (1st Audio)	
audioBitrate2 *optional*	Average audio bitrate in Kpbs (2nd Audio)	number
audioCodec *optional*	Audio codec to use (1st Audio)	enum (AAC-LC, MPEG-1 Layer 2)
audioCodec2 *optional*	Audio codec to use (2nd Audio)	enum (AAC-LC)
audioEnabled *optional*	Enable first audio encoding	boolean
audioEnabled2 *optional*	Enable second audio encoding	boolean
audioMode *optional*	Audio mode	enum (Stereo, Mono Left, Mono Right)
audioMode2 *optional*	Audio mode (2nd Audio)	enum (Stereo, Mono Left, Mono Right)
audioSource *optional*	Unique identifier representing a audio input (1st Audio)	enum (SDI Embedded, HDMI Embedded, Unbalanced analog, Balanced analog, Unbalanced digital 1, Unbalanced digital 2)
audioSource2 *optional*	Unique identifier representing a audio input (2nd Audio)	enum (SDI Embedded, HDMI Embedded, Unbalanced analog, Balanced analog, Unbalanced digital 1, Unbalanced digital 2)
encryption *optional*	Encryption mode to use	enum (None, AES 128 bit(PC+STB))
format *optional*	Encoding format. Can not be set when matchToInput is enabled.	enum (352x288p, 352x576p, 544x576p, 704x576p, 720x480i, 720x576i, 1280x720p, 1920x1080i, 1920x1080p)
frameRate *optional*	Frame Rate of the ending format [1-60]	integer

Name	Description	Schema
klvSource *optional*	Unique identifier representing a klv source (1st KLV)	enum (None, Serial, IP, SDI-Sync)
klvSource2 *optional*	Unique identifier representing a klv source (2nd KLV)	enum (None, Serial, IP, SDI-Sync)
level *optional*	Encoding level [RO]	enum (1.0, 1.1, 1.2, 1.3, 2.0, 2.1, 2.2, 3.0, 3.1, 3.2, 4.0, 4.1, 4.2, 5.0, 5.1, 5.2, 6.0, 6.1, 6.2)
matchToInput *optional*	Match encoding resolution to input resolution	boolean
maxStreamBitrate *optional*	Maximum stream bitrate in Mbps for VBR	number
profile *optional*	Encoding profile	enum (H.264 Baseline Profile, H.264 Main Profile, H.264 High Profile, HEVC 8-Bit 4:2:0, HEVC 8-Bit 4:2:2, HEVC 10-Bit 4:2:0, HEVC 10-Bit 4:2:2)
rateControl *optional*	Rate control setting	enum (Capped VBR, CBR)
regionOfInterest *optional*	Region Of Interest. Allows to crop the input video [H.264 only]	boolean
streamBitrate *optional*	Stream average bitrate in Mbps	number
videoEnabled *optional*	Enable video encoding	boolean
videoSource *optional*	Unique identifier representing a video input.	enum (Composite, HD-SDI, HDMI, DVI-D)

Target

Name	Description	Schema
address *optional*	Target address	enum (225.1.1.1, 192.168.1.1)
enabled *optional*	Target status	boolean
fec1dPortNumber *optional*	Pro-Mpeg fec1dPortNumber value [1-65535]	number
fec2dPortNumber *optional*	Pro-Mpeg fec2dPortNumber value [1-65535]	number
fecColumns *optional*	Pro-Mpeg fecColumns value [4-20]	number
fecMode *optional*	Pro-Mpeg fecMode value	enum (1-D, 2-D)
fecRows *optional*	Pro-Mpeg fecRows value [4-20]	number
multicastInterface *optional*	Multicast Interface	enum (Ethernet 1, Ethernet 2)
name *optional*	Target name	enum (Channel_HEVC, Channel_H264)
port *optional*	Target Port	number
protocol *optional*	Target protocol	enum (UDP TS, Pro-MPEG, Zixi, RTP ES, RTP TS, RTMP)
rtpServerPort *optional*	RTP Server Port	number
ttl *optional*	interface ttl value	number
zixiLatency *optional*	Zixi latency value	number
zixiPassword *optional*	Zixi password value	enum (none)