User manual

（Onvif Rtsp Server）

Happytimesoft Technology Co., LTD

Declaration

Since the product version upgrade or other reasons, this manual will subsequently be updated. Unless otherwise agreed, this manual only as a guide, this manual all statements, information, recommendations do not constitute any express or implied warranties。

Table of Contents

Chapter 1 ONVIF SERVER

Onvif server uses default configuration file onvif.cfg.

1.1 Configuration

1.1.1 Configuration Templates

```
<?xml version="1.0" encoding="utf-8"?>
<config>
    <server_ip></server_ip>
    <http_enable>1</http_enable>
    <http_port>8000</http_port>
    <https_enable>1</https_enable>
    <https_port>8443</https_port>
    <cert_file>ssl.ca</cert_file>
    <key_file>ssl.key</key_file>
    <http_max_users>16</http_max_users>
    <need_auth>0</need_auth>
    <log_enable>1</log_enable>
    <log_level>1</log_level>
    <information>
        <Manufacturer>Happytimesoft</Manufacturer>
        <Model>IPCamera</Model>
        <FirmwareVersion>2.4</FirmwareVersion>
        <SerialNumber>123456</SerialNumber>
        <HardwareId>1.0</HardwareId>
    </information>
    <user>
        <username>admin</username>
        <password>admin</password>
        <userlevel>Administrator</userlevel>
    </user>
    <profile>
        <video_source>
            <width>1280</width>
```

```
            <height>720</height>
        </video_source>
        <video_encoder>
            <width>1280</width>
            <height>720</height>
            <quality>4</quality>
            <session_timeout>10</session_timeout>
            <framerate>25</framerate>
            <encoding_interval>1</encoding_interval>
            <bitrate_limit>2048</bitrate_limit>
            <encoding>H264</encoding>
            <h264>
                <gov_length>25</gov_length>
                <h264_profile>Main</h264_profile>
            </h264>
        </video_encoder>
        <audio_source></audio_source>
        <audio_encoder>
            <session_timeout>10</session_timeout>
            <sample_rate>8</sample_rate>
            <bitrate>64</bitrate>
            <encoding>G711</encoding>
        </audio_encoder>
        <stream_uri append_params="0"></stream_uri>
    </profile>
    <scope>onvif://www.onvif.org/location/country/china</scope>
    <scope>onvif://www.onvif.org/name/IP-Camera</scope>
    <scope>onvif://www.onvif.org/hardware/HI3518C</scope>
    <event>
        <renew_interval>60</renew_interval>
        <simulate_enable>1</simulate_enable>
    </event>
</config>
```

1.1.2 Configuring Node Description

<server_ip>

Specify the IP address of the onvif server, if not specified, the onvif server will listen to all network interfaces.

<http_enable>

Indicates whether enable http server, 0-disable, 1-enable.

<http_port>

Specify the http server port, providing onvif web service on this port, the default is 8000.

Note: On Linux systems, ports below 1024 are reserved by the system and require root privileges to be used.

<https_enable>

Indicates whether enable https server, 0-disable, 1-enable.

<https_port>

Specify the https server port, providing onvif web service on this port, the default is 8443.

Note: On Linux systems, ports below 1024 are reserved by the system and require root privileges to be used.

<cert_file>

If HTTPS is enabled, specify the SSL certificate file.

<key_file>

If HTTPS is enabled, specify the SSL key file.

Note: The certificate file ssl.ca and key file ssl.key provided by default are self signed local hosts certificates, only for testing purposes (browsers may pop up untrusted certificate warnings), and cannot be used in formal deployment environments.

<http_max_users>

Maximum supported HTTP clients numbers, if both HTTP and HTTPS are enabled, they can support 2 * http_max_users connections in total.

<need_auth>

Indicates whether authentication is required, 0 don't require, 1 require.

<log_enable>

Indicates whether logging is enabled, 0-disable, 1-enable.

<log_level>

The log level:

TRACE 0

DEBUG 1

INFO 2

WARN 3

ERROR 4

FATAL 5

<information> : Config the ONVIF device basic information

<Manufacturer>

The manufactor of the device.

<Model>

The device model.

<FirmwareVersion>

The firmware version of the device.

<SerialNumber>

The serial number of the device.

<HardwareId>

The hardware ID of the device.

<user> : Contains a list of the onvif users, it can configure multiple nodes

<username>

Username string

<password>

Password string

<userlevel>

User level string, the following values can be configured:

Administrator

Operator

User

Anonymous

<profile> : A media profile maps a video and audio source to a video and audio encoder configurations. It can configure multiple nodes.

Currently, a maximum of 8-10 profiles are supported, because too many profiles will result in too large GetProfiles response messages.

<video_source> : If the media profile contains video, the video source configuration

<width>

The video source width.

<height>

The video source height.

Note : If the video sources of two profiles are of the same size, it is considered that they are using the same video source. Otherwise, it is considered that they are using two different video sources.

<video_encoder>: If the media profile contains video, the video encoder configuration

<width>

Encoded video width.

<height>

Encoded video height.

<quality>

Relative value for the video quantizers and the quality of the video. A high value within supported quality range means higher quality.

<session_timeout>

The rtsp session timeout for the related video stream.

<framerate>

Maximum output framerate in fps.

<encoding_interval>

Interval at which images are encoded and transmitted. (A value of 1 means that every frame is encoded, a value of 2 means that every 2nd frame is encoded ...).

<bitrate_limit>

The maximum output bitrate in kbps.

<encoding>

Used video codec, either JPEG, MPEG4, H264 or H265.

<h264>: Configure H.264 related parameters

<gov_length>

Group of Video frames length. Determines typically the interval in which the I-Frames will be coded. An entry of 1 indicates I-Frames are continuously generated. An entry of 2 indicates that every 2nd image is an I-Frame, and 3 only every 3rd frame, etc. The frames in between are coded as P or B Frames.

<h264_profile>

The H.264 profile, either Baseline, Main, Extended or High.

<h265>: Configure H.265 related parameters

<gov_length>

Group of Video frames length. Determines typically the interval in which the I-Frames will be coded. An entry of 1 indicates I-Frames are continuously generated. An entry of 2 indicates that every 2nd image is an I-Frame, and 3 only every 3rd frame, etc. The frames in between are coded as P or B Frames.

<h265_profile>

The H.265 profile, either Main or Main10.

<mpeg4>: Configure MPEG4 related parameters

<gov_length>

Determines the interval in which the I-Frames will be coded. An entry of 1 indicates I-Frames are continuously generated. An entry of 2 indicates that every 2nd image is an I-Frame, and 3 only every 3rd frame, etc. The frames in between are coded as P or B Frames.

<mpeg4_profile>

The Mpeg4 profile, either simple profile (SP) or advanced simple profile (ASP).

<audio_source> : If the media profile contains audio, the audio source configuration

<audio_encoder>:If the media profile contains audio, the audio encoder configuration

<session_timeout>

The rtsp session timeout for the related audio stream.

<sample_rate>

The output sample rate in kHz.

<bitrate>

The output bitrate in kbps.

<encoding>

Audio codec used for encoding the audio input (either G711, G726 or AAC).

<stream_uri append_params="0">

The RTSP stream address of the profile, if not specify, the default is: *rtsp://ip:port/test.mp4*

The **append_params** attribute specifies whether to append audio and video encoding parameters to the end of the rtsp stream.If the stream_uri attribute does not specify an rtsp stream address, the default rtsp stream address will append audio and video encoding parameters regardless of whether append_params is 0 or 1.The format of the appended parameters is as follows:

&params=value

The supported params are as follows:

t, transmission mode, taking the value of unicast to represent unicast or multicast to represent multicast.

p, transmission protocol, value udp, tcp, rtsp, http.

ve, video encoding, value JPEG, MP4V-ES, H264, H265.

w, video width.

h, video height.

ae, audio encoding, value PCMU, G726, MP4A-LATM (AAC).

sr, audio sample rate.

For example:

rtsp://127.0.0.1/test.mp4&t=unicast&p=udp&ve=H264&w=1280&h=720&ae=PCMU&sr=8000

Indicates UDP unicast mode, video encoding is H264, video resolution is 1280*720, audio encoding is PCMU, sampling rate is 8K.

<scope>

Contains a list of URI definining the device scopes.

All ONVIF defined scope URIs have the following format:

onvif://www.onvif.org/<path>

A device may have other scope URIs. These URIs are not restricted to ONVIF defined

scopes.

A device shall include at least one fixed entry (defined by the device vendor) of the profile, hardware and name categories respectively in the scopes list. A device may include any other additional scope attributes in the scopes list.

A device might include an arbitrary number of scopes in its scope list. This implies that one unit might for example define several different location scopes. A probe is matched against all scopes in the list.

<event> : Event Configuration parameters

<renew_interval>

Event renew interval.

The onvif client subscribes or creates an event polling point. If the renew or pullmessage request is not called within the renew_interval interval, the onvif server will delete the subscription or event polling point.

<simulate_enable>

Specifies whether to generate simulation event, 0-disable, 1-enable.

1.2 Configuration file

When running onvif rtsp server for the first time, use the default configuration file onvif.cfg, which sets 2 profiles.

When stop onvif rtsp server, it writes the runtime configuration into the onvifrun.cfg file, and the configuration in the onvifrun.cfg file will be load at the next time it runs.

If you modify the default configuration file onvif.cfg, you should stop the onvif rtsp server first, then delete the runtime configuration file onvifrun.cfg, and run onvif rtsp server again to make the default configuration effective.

1.3 Compatibility test

ONVIF SERVER PROFILE T passed the compatibility test version.

Windows version download from:

https://www.happytimesoft.com/downloads/happytime-onvif-rtsp-server-profilet.zip

Linux version download from:

https://www.happytimesoft.com/downloads/happytime-onvif-rtsp-server-profilet.tar.gz

Follow the steps below to perform compatibility testing.

1. Modify the ONVIF SERVER configuration file onvif.cfg and specify the <need_auth> value as 1.

2. Modify the RTSP SERVER configuration file rtsp.cfg and specify the <need_auth> value as 1.

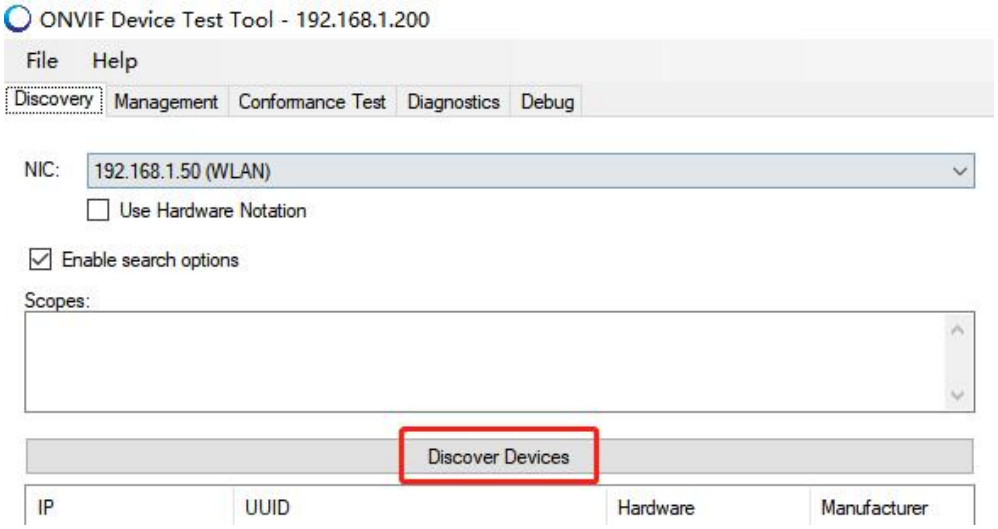
3. If there is an onvif runtime configuration file onvifrun.cfg, delete the runtime configuration file onvifrun.cfg.

4. Run the onvifrtspserver server.

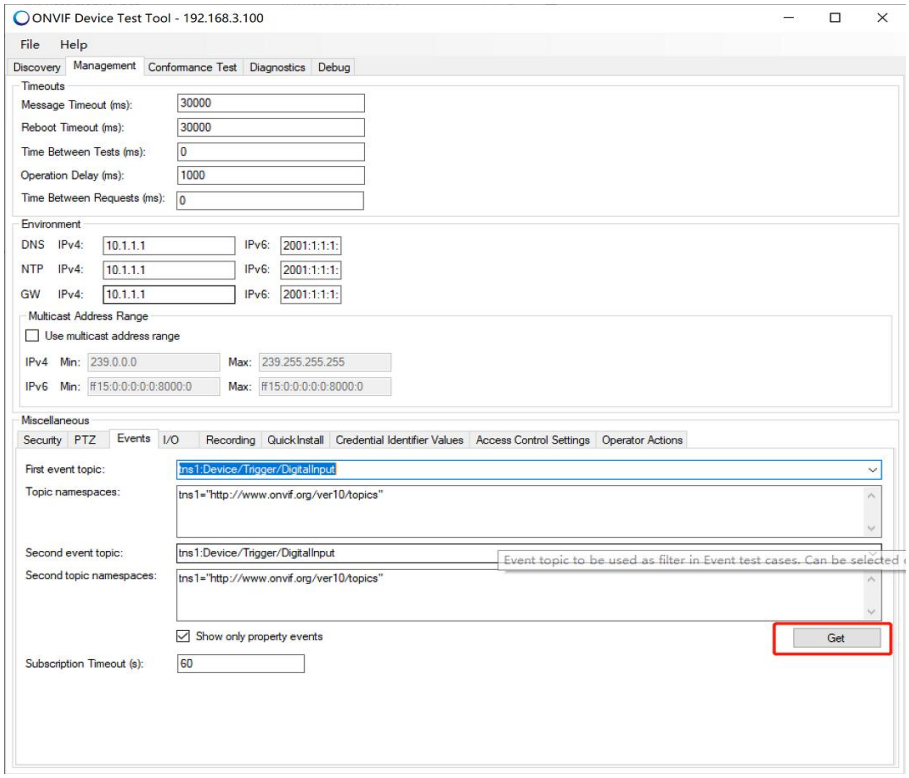
5. Run the ONVIF Device Test Tool.

Note: ONVIF RTSP SERVER and test tools should run on different computers.

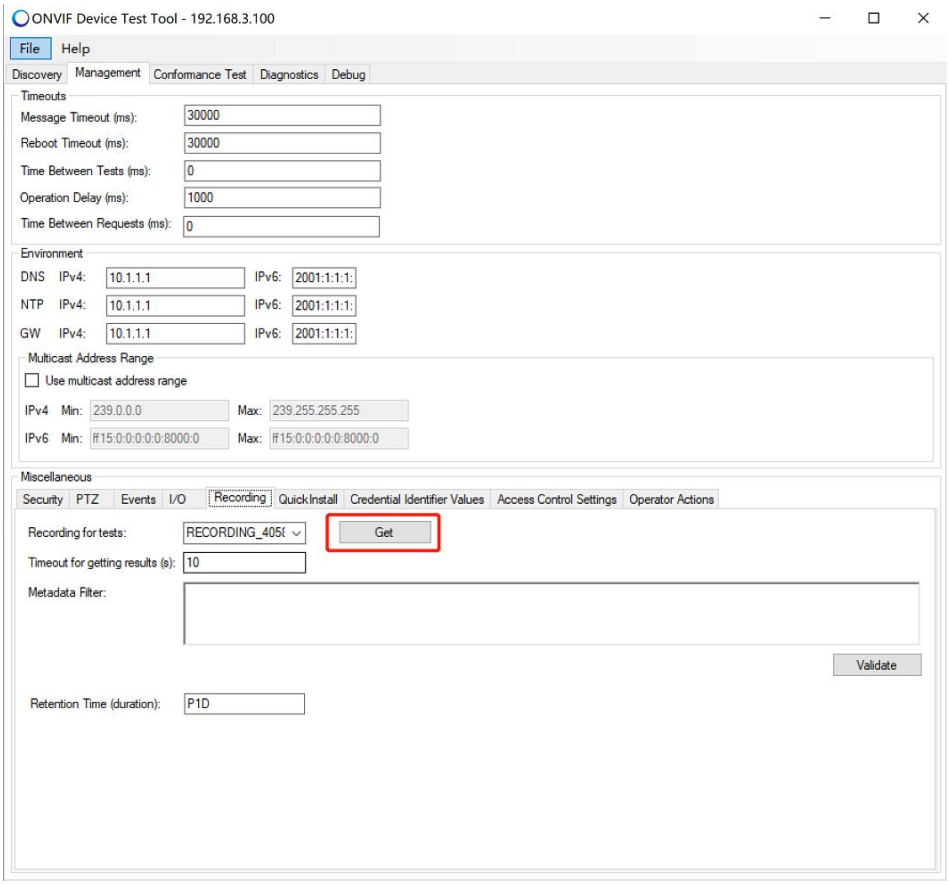
6. Click "Discover Devices" button, as the following:



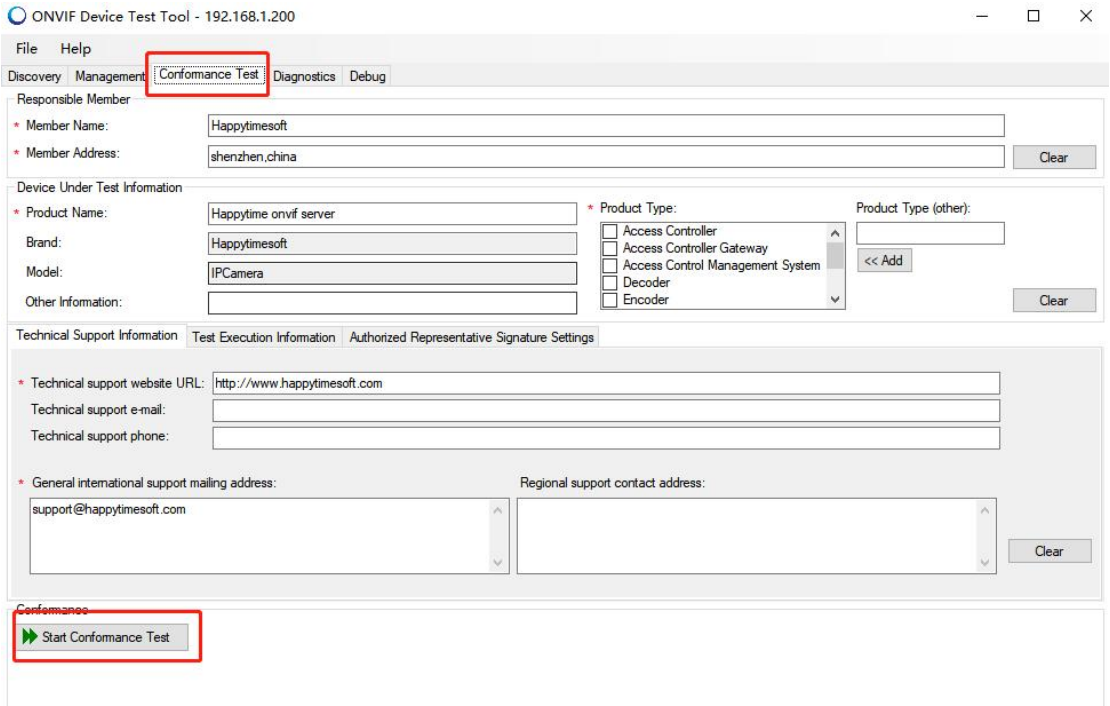
7. Switch to "Management" tab, select "Events" tab, then click "Get" button, as the following:



8. select “Recording” tab, then click “Get” button, as the following:



9. Switch to “Conformance Test” tab, click “Start Conformance Test” button:



1.4 ONVIF features

The onvif server supports the onvif features listed in the following table:

Feature		
Security	WS-Username Token	
	Digest	
Discovery	BYE Message	
	Types	tds:Device
		dn:Network Video Transmitter
Device Service	Capabilities	GetCapabilities
		GetService
	Network	Zero Configuration
		NTP
		Dynamic DNS
		IP Filter
		HTTPS
	System	System Logging
		HTTP System Logging
		HTTP Firmware Upgrade
		HTTP Support Information

		HTTP System Backup
	Security	Default Access Policy
		Maximum Users
		Remote User Handling
		Maximum Username Length
		Maximum Password Length
	I/O	Relay outputs
Event Service	WS Basic Notification	
	Message Content Filter	ONVIF Message Content Filter Dialect
	Get Service Capabilities	MaxPullPoints capability
	Pull-Point Notification	
Media Service	Video	JPEG
		H.264
		MPEG4
	Audio	G.711
		G.726
		AAC
	Audio Output	G.711
		AAC
	Real-time Streaming	RTP/UDP
		RTP/RTSP/HTTP
		RTP/RTSP/TCP
		RTP-Multicast/UDP
	Snapshot URI	
Media2 Service	Video	H.265
		H.264
	Audio	G.711
		AAC
	Audio outputs	G.711
		AAC
	Real-time Streaming	RTP/UDP
		RTP/RTSP/HTTP
		RTP/RTSP/TCP
		RTP-Multicast/UDP
	RTSP WebSocket	
	Snapshot URI	

	Video Source Mode	
	OSD	
	Analytics	
	Metadata	
	Media2 Events	Media/ProfileChanged
		Media/ConfigurationChanged
PTZ Service	Absolute move	Pan/Tilt movement
		Zoom movement
	Relative move	Pan/Tilt movement
		Zoom movement
	Continuous move	Pan/Tilt movement
		Zoom movement
	Presets	
	Home position	Configuration
	Auxiliary operations	
	Speed	Speed for Pan/Tilt
		Speed for Zoom
	Move Status	
	Status Position	
	Get Compatible Configurations	
Device IO Service	Relay outputs	Bistable Mode
		MonoStable Mode
	Digital Inputs	Digital Input Options
Imaging Service	IrCutfilter Configuration	
	Tampering Events	Image Too Blurry
		Image Too Dark
		Image Too Bright
		Global Scene Change
	Motion Alarm	
	Focus Control	
Analytics Service	Rule Engine	Rule Options
		Motion Region Detector Rule
	Analytics Modules	Analytics Module Options
Recording Control Service	Dynamic Recordings	
	Dynamic Tracks	
	Audio Recording	
	Recording Options	

	tns1:RecordingCofig/DeleteTrckData	
	Metadata Recording	
	Encoding	JPEG
		H264
		MPEG4
Recording Search Service	Metadata Search	
	PTZ Position Search	
Door Control Service	Door Entity	Access Door
		Lock Door
		Double Lock Door
		Block Door
		Lock Down Door
		Lock Open Door
		Door Monitor
		Double Lock Monitor
		Alarm
		Tamper
		Fault
	Door Control Events	
	Door Management	
	Client Supplied Token	
Access Control Service	Area Entity	
	Access Point Entity	Enable/Disable Access Point
		Duress
		Access Taken
		Anonymous Access
	Access Point Management	
	Area Management	
	Access Control Events	
Replay Service	RTP/RTSP/TCP	
Receiver Service		
Credential Service	Credential Validity	
	Credential Access Profile Validity	
	pt:Card	
	pt:PIN	

	pt:Fingerprint	
	Reset Antipassback Violation	
	Client Supplied Token	
	Whitelist	
	Blacklist	
	Validity Supports Time Value	
Access Rules Service	Multiple Schedules Access Point	
	Client Supplied Token	
Schedule Service		
Thermal Service		

1.5 ONVIF Version

The onvif server implements the following ONVIF service:

ONVIF Service	Prefix	Url	version
device	tds	http://www.onvif.org/ver10/device/wsdl	23.06
event	tev	http://www.onvif.org/ver10/events/wsdl	22.06
media	trt	http://www.onvif.org/ver10/media/wsdl	21.12
media 2	tr2	http://www.onvif.org/ver20/media/wsdl	23.06
ptz	tptz	http://www.onvif.org/ver20/ptz/wsdl	22.12
image	timg	http://www.onvif.org/ver20/imaging/wsdl	22.06
analytics	tan	http://www.onvif.org/ver20/analytics/wsdl	22.06
recording control	trc	http://www.onvif.org/ver10/recording/wsdl	23.06
search	tse	http://www.onvif.org/ver10/search/wsdl	22.06
replay	trp	http://www.onvif.org/ver10/replay/wsdl	21.12
access control	tac	http://www.onvif.org/ver10/accesscontrol/wsdl	21.06
door control	tdc	http://www.onvif.org/ver10/doorcontrol/wsdl	21.06
device IO	tmd	http://www.onvif.org/ver10/deviceIO/wsdl	22.06
thermal	tth	http://www.onvif.org/ver10/thermal/wsdl	22.06
credential	tcr	http://www.onvif.org/ver10/credential/wsdl	21.06
access rules	tar	http://www.onvif.org/ver10/accessrules/wsdl	19.06
schedule	tsc	http://www.onvif.org/ver10/schedule/wsdl	18.12
receiver	trv	http://www.onvif.org/ver10/receiver/wsdl	21.12
provisioning	tpv	http://www.onvif.org/ver10/provisioning/wsdl	18.12

1.6 Supports multiple channels

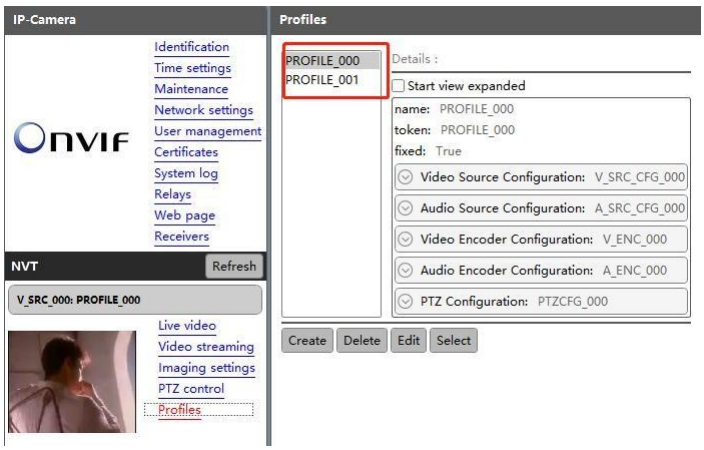
The onvif server supports multi channel. Each <profile> tag represents a channel in the configuration file.

The default configuration file supports 2 channels, you can add <profile> tag to support more channels.

Note : If <video_source>.width and <video_source>.height of multiple <profile> tags are the same, it is considered that they are using the same video source, example:

```
<profile>
    <video_source>
        <width>1280</width>
        <height>720</height>
    </video_source>
    ....
</profile>
<profile>
    <video_source>
        <width>1280</width>
        <height>720</height>
    </video_source>
    ....
</profile>
```

The onvif device manager will show the profiles as the following:



If <video_source>.width and <video_source>.height of multiple <profile> tags are not the same, it is considered that they are using different video sources,

example:

```
<profile>
    <video_source>
        <width>1280</width>
        <height>720</height>
    </video_source>
    ....
</profile>

<profile>
    <video_source>
        <width>640</width>
        <height>480</height>
    </video_source>
    ....
</profile>
```

The onvif device manager will show the profiles as the following:

1.7 Modify RTSP stream address

If the value of <stream_uri> in the <profile> tag in the onvif server configuration file is not modified, the RTSP stream address provided by the onvif server by default is rtsp://ip:port/test.mp4, and the user can modify the

<stream_uri> in <profile> tag to specify the rtsp stream address provided by the onvif server. such as:

```
<profile>
    ...
    <stream_uri>rtsp://192.168.3.27/live</stream_uri>
</profile>
```

Chapter 2 RTSP SERVER

2.1 Introduction

Happytime RTSP Server is a complete RTSP server application. It can stream audio and video files in various formats.

It can also stream video from camera, living screen and application windows, stream audio from audio device.

It can stream H265, H264, MP4, MJPEG video stream and G711, G722, G726, AAC, OPUS audio stream.

These streams can be received/played by standards-compliant RTSP/RTP media clients.

It supports rtsp proxy function.

It supports audio back channel function.

It supports rtsp over http function.

It supports rtsp over https function.

It supports rtp multicast function.

Support for data pusher function.

Enjoying multimedia content from your computer can be a pleasant way for you to spend your free time. However, sometimes you might need to access it from various locations, such as a different computer or a handheld device, Happytime RTSP Server, that can help you achieve quick and efficient results.

2.2 Key features

The server can transmit multiple streams concurrently

It can stream audio and video files in various formats

It can stream audio from audio device

It can stream video from camera and living screen

It can stream video from application windows

It can stream H265, H264, MP4, MJPEG video stream

It can stream G711, G722, G726, AAC, OPUS audio stream

It supports rtsp over http function

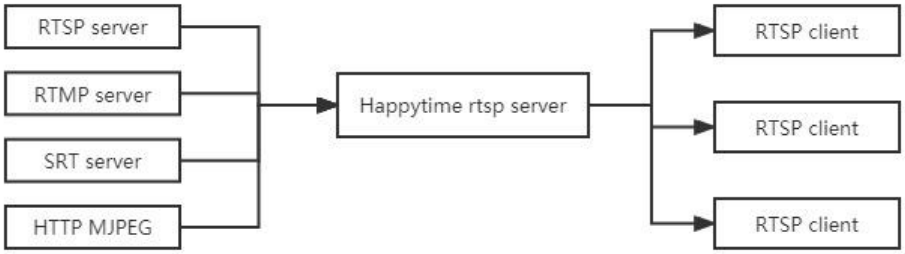
It supports rtsp over https function

It supports rtsp over websocket function

It supports rtp multicast function

It supports data pusher function

It supports RTSP proxy function, as the following:



It supports audio backchannel.

Happytime rtsp server complies with onvif audio backchannel specification, please refer to the link below for specification details:

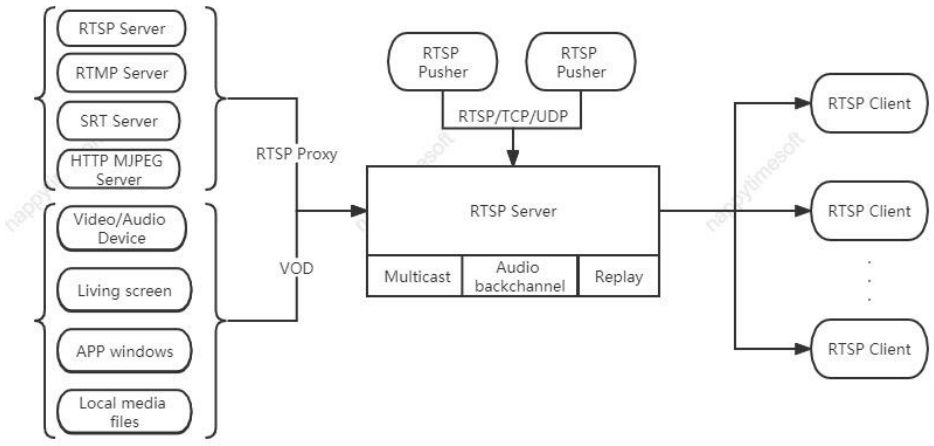
https://happytimesoft.com/knowledge/audio-back-channel.html

It supports audio and video playback.

Happytime rtsp server complies with onvif audio and video playback specification, please refer to the link below for specification details:

https://happytimesoft.com/knowledge/audio-video-playback.html

2.3 Function chart



2.4 Configuration

2.4.1 Configuration Templates

```
<?xml version="1.0" encoding="utf-8"?>
<config>
    <serverip></serverip>
    <serverport>554</serverport>
    <loop_nums>1</loop_nums>
    <multicast>0</multicast>
    <udp_base_port>22000</udp_base_port>
    <metadata>1</metadata>
    <rtsp_over_http>1</rtsp_over_http>
    <http_port>8080</http_port>
    <rtsp_over_https>1</rtsp_over_https>
    <https_port>443</https_port>
    <cert_file>ssl.ca</cert_file>
    <key_file>ssl.key</key_file>
    <need_auth>0</need_auth>
    <log_enable>1</log_enable>
    <log_level>1</log_level>

    <user>
        <username>admin</username>
        <password>admin</password>
    </user>

    <output>
        <url>screenlive</url>
        <video>
            <codec>H264</codec>
            <width></width>
            <height></height>
            <framerate></framerate>
            <bitrate></bitrate>
```

```
    </video>
    <audio>
        <codec>G711U</codec>
        <samplerate>8000</samplerate>
        <channels>1</channels>
        <bitrate></bitrate>
    </audio>
</output>
<proxy>
    <suffix>proxy</suffix>
    <url></url>
    <user></user>
    <pass></pass>
    <transfer>TCP</transfer>
    <ondemand>0</ondemand>
    <output>
        <video>
            <codec>H264</codec>
            <width></width>
            <height></height>
            <framerate></framerate>
            <bitrate></bitrate>
        </video>
        <audio>
            <codec>AAC</codec>
            <samplerate></samplerate>
            <channels></channels>
            <bitrate></bitrate>
        </audio>
    </output>
</proxy>

<pusher>
    <suffix>pusher</suffix>
```

```
    <video>
        <codec>H264</codec>
    </video>
    <audio>
        <codec>G711U</codec>
        <samplerate>8000</samplerate>
        <channels>1</channels>
    </audio>
    <transfer>
        <mode>RTSP</mode>
        <ip></ip>
        <vport>50001</vport>
        <aport>50002</aport>
    </transfer>
    <output>
        <video>
            <codec></codec>
            <width></width>
            <height></height>
            <framerate></framerate>
            <bitrate></bitrate>
        </video>
        <audio>
            <codec></codec>
            <samplerate></samplerate>
            <channels></channels>
            <bitrate></bitrate>
        </audio>
    </output>
</pusher>

<backchannel>
    <codec>G711U</codec>
    <samplerate>8000</samplerate>
```

```
            <channels>1</channels>
        </backchannel>
    </config>
```

2.5 Configuring Node Description

2.5.1 System parameters

<serverip>

Specify the IP address of the RTSP server, if not specified, the rtsp server will listen to all network interfaces.

<serverport>

Specify the RTSP server port, the default is 554.

Note: On Linux systems, ports below 1024 are reserved by the system and require root privileges to be used.

<loop_nums>

When streaming media files, specify the number of loop playback,-1 means infinite loop.

<multicast>

Whether to enable rtp multicast function, 0-disable, 1-enable.

<udp_base_port>

UDP media transmission base port, RTSP over UDP mode assign UDP port on this base port.

Each rtsp session needs to assign 8 UDP ports, video RTP/RTCP port, audio RTP/RTCP port, METADATA stream RTP/RTCP port, audio back-channel RTP/RTCP port.

<metadata>

Whether to enable the meta data stream, 0-disable, 1-enable.

<rtsp_over_http>

Whether to enable rtsp over http function, 0-disable,1-enable.

<http_port>

Specify the HTTP server port for rtsp over http function.

Note: On Linux systems, ports below 1024 are reserved by the system and require root privileges to be used.

<rtsp_over_https>

Whether to enable rtsp over https function, 0-disable,1-enable.

<https_port>

Specify the HTTPS server port for rtsp over https function.

Note: On Linux systems, ports below 1024 are reserved by the system and require root privileges to be used.

<cert_file>

Specify the HTTPS server certificate file.

<key_file>

Specify the HTTPS server key file.

Note: The certificate file ssl.ca and key file ssl.key provided by default are self signed local hosts certificates, only for testing purposes (browsers may pop up untrusted certificate warnings), and cannot be used in formal deployment environments.

<need_auth>

Whether enable the user authentication function,0-disable,1-enable.

<log_enable>

Whether enable the log function,0-disable,1-enable.

<log_level>

The log level:

TRACE 0

DEBUG	1
INFO	2
WARN	3
ERROR	4
FATAL	5

2.5.2 *User* node

<user> : Specify the login username password, it can configure multiple nodes

<username>

The login username.

<password>

The login password.

2.5.3 *Output* node

<output> : Specify the audio and video output parameters, it can configure multiple nodes.

<url>

Match URL address, it can be filename, or file extension name, or special suffix. Such as:

screenlive : match living screen stream

videodevice : match camera video stream

*.mp4 : match all mp4 media file

sample.flv : match sample.flv file

If not config this node, it will match all url as the audio/video default output parameters.

The match order from top to bottom, therefore the default output configuration should be placed in the last.

<video> : Specify the video output parameters

<codec>

Specify the video stream codec, it can specify the following value:

H264 : output H264 video stream

H265 : output H265 video stream

MP4: output MP4 video stream

JPEG: output MJPEG video stream

<width>

Specify the output video width, If 0 use the original video width (live screen stream use the screen width, camera stream use the default width)

<height>

Specify the output video height, If 0 use the original video height (live screen stream use the screen height, camera stream use the default height)

<framerate>

Specify the output video framerate, If 0 use the original video framerate (live screen use the default value 15, camera stream use the default value 25)

<bitrate>

Specify the output video bit rate, if 0, automatically calculate the output bit rate, the unit is kb/s.

Note: This parameter is valid only if encoding is required (eg screenlive, videodevice) or transcoding is required.

<audio> : Specify the audio output parameters

<codec>

Specify the audio stream codec, it can specify the following value:

G711A: output G711 a-law audio stream

G711U: output G711 mu-law audio stream

G722: output G722 audio stream

G726: output G726 audio stream

AAC: output AAC audio stream

OPUS: output OPUS audio stream

<samplerate>

Specify the audio sample rate, it can specify the following values:

8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000

If 0 use the original audio sample rate (audio device stream use the default value 8000)

<channels>

Specify the audio channel number, 1 is mono, 2 is stereo

If 0 use the original audio channel number (audio device stream use the default value 2)

Note : G726 only support mono.

<bitrate>

Specify the output audio bit rate, if 0, automatically calculate the output bit rate, the unit is kb/s.

Note: This parameter is valid only if encoding is required (eg screenlive, videodevice) or transcoding is required.

2.5.4 *Proxy* node

<proxy> : Specify the rtsp proxy parameters, it can configure multiple nodes

<suffix>

Specify the rtsp stream suffix, you can play the proxy stream from:

rtsp://[serverip]:[serverport]/[suffix]

<url>

The original rtsp/rtmp/srt/http mjpeg stream address.

<user> <pass>

Specify the original rtsp/rtmp/srt/http mjpeg stream login user and password.

<transfer>

Specify the rtsp client transfer protocol:

TCP: rtsp client uses RTP over TCP

UDP: rtsp client uses RTP over UDP

MULTICAST: rtsp client uses multicast

<ondemand>

Connect on demand, 1-Connect when needed, 0-Always keep connected.

<output>

Specify the stream output parameter. If the parameter does not appear, use the parameters of the original RTSP/RTMP/SRT/HTTP MJPEG stream. If it

appears and the configured parameters are inconsistent with the parameters of the original RTSP/RTMP/SRT/HTTP MJPEG stream, then the transcode output is performed.

The child nodes under this node are consistent with the meaning of the <output> node.

2.5.5 *Pusher* node

<pusher> : Specify the data pusher parameters, it can configure multiple nodes

<suffix>

Specify the rtsp stream suffix, you can play the pusher stream from: rtsp://[serverip]:[serverport]/[suffix]

<video> : Specify the the input video data parameters

<codec>

Specify the video codec, it can specify the following value:

H264 : H264 video stream

H265 : H265 video stream

JPEG: MJPEG video stream

MP4: MPEG4 video stream

<audio> : Specify the input audio data parameters

<codec>

Specify the audio codec, it can specify the following value:

G711A: G711 a-law audio stream

G711U: G711 mu-law audio stream

G722: G722 audio stream

G726: G726 audio stream

OPUS: OPUS audio stream

AAC: AAC audio stream

<samplerate>

Specify the audio sample rate, it can specify the following values: 8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000

<channels>

Specify the audio channel number, 1 is mono, 2 is stereo

Note : G726 only support mono.

<transfer>: Specify the data transfer parameters

<mode>: Specify the data transer protocol, it can specify the following value:

TCP: Use TCP connection to transfer the data

UDP: Use UDP connection to transfer the data

RTSP: Use RTSP connection to transfer the data. It supports standard rtsp push, such as FFMPEG rtsp push.

<ip>: Specified data receiving IP address, if there is no configuration, the default IP address is used.

<vport>: Specify the video data receiving port

<aport>: Specify the audio data receiving port

Note : <ip>, <vport>, <aport> these 3 parameters are valid when <mode> is TCP or UDP.

<output>

Specify the stream output parameter. If the parameter does not appear, use the parameters of the original pusher stream. If it appears and the configured parameters are inconsistent with the parameters of the original pusher stream, then the transcode output is performed.

The child nodes under this node are consistent with the meaning of the <output> node.

2.5.6 *Backchannel* node

<backchannel> : specify the audio back channel parameters

<codec>

Specify the audio back channel stream codec, it can specify the following value:

G711A: G711 a-law audio stream

G711U: G711 mu-law audio stream

G722: G726 audio stream

G726: G726 audio stream

OPUS: OPUS audio stream

<samplerate>

Specify the audio back channel sample rate, it can specify the following values:

8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000

If 0 use the default value 8000

<channels>

Specify the audio channel number, 1 is mono, 2 is stereo

If 0 use the default value 1

Note : G726 only support mono.

2.6 Data pusher

Data push means that RTSP server receives external data sources and then sends them out as RTSP streams.

The data pusher supports TCP, UDP and RTSP mode.

Audio and video data are packaged and sent in RTP format.

If it is TCP mode, you need to add 4 bytes in front of the RTP header, as the following:

```
typedef struct
{
    uint32  magic      : 8;
    uint32  channel : 8;
    uint32  rtp_len : 16;
} RILF;
```

magic: 0x24

channel: 0

rtp_len: the RTP load length, including RTP header

Note: If you use TCP or UDP mode data push, you need to add <pusher> tag in the rtsp server configuration file, specify the push audio and video parameters and push port, etc.

If you use RTSP mode to push data, no configuration is required. The url suffix

of the pushed RTSP address can be any legal string.

If it is RTSP mode, it supports standard RTSP push stream, such as FFMPEG rtsp push.

FFMPEG rtsp over UDP:

ffmpeg -re -i test.mp4 -vcodec libx264 -acodec copy -preset ultrafast -f rtsp rtsp://[serverip]:[serverport]/pusher

FFMPEG rtsp over TCP:

ffmpeg -re -i test.mp4 -vcodec libx264 -acodec copy -preset ultrafast -f rtsp -rtsp_transport tcp rtsp://[serverip]:[serverport]/pusher

After the above ffmpeg rtsp push command is run, you can use the following address to play the rtsp stream:

rtsp://[serverip]:[serverport]/pusher

2.7 RTSP over HTTP

The key of RTSP over HTTP is to allow RTSP packets to communicate via HTTP port.

We know that the standard port of RTSP is 554, but due to various security policy configurations such as firewalls, there may be restrictions when the client accesses port 554, which prevents the normal transmission of RTSP packets.

But the HTTP port (port 80) is generally open, so there is the idea of letting RTSP packets pass through port 80, namely RTSP over HTTP

The details of RTSP over HTTP are as follows:

First, the client opens two socket connect to the rtsp server HTTP ports. We call these two sockets "data socket" and "command socket".

Step 1. The client sends an HTTP GET command through the "data socket" to request an RTSP connection.

Step 2. The server responds to the HTTP GET command through the "data socket" and responds with success/failure.

Step 3. The client creates a "command socket" and sends an HTTP POST command through the "command socket" to establish an RTSP session.

At this point, the auxiliary function of HTTP is completed, and the server does

not return the client's HTTP POST command. Next is the standard process of RTSP on the HTTP port, but it needs to be completed through two sockets. The "command socket" is only responsible for sending, and the "data socket" is only responsible for receiving.

Step 4. The client sends RTSP commands (BASE64 encoding) through the "command socket".

Step 5. The server responds to the RTSP command (in plain text) through the "data socket".

Step 6. Repeat Step4-Step5 until the client sends the RTSP PLAY command and the server responds to the RTSP PLAY command.

Step 7. The server transmits audio and video data to the client through the "data socket"

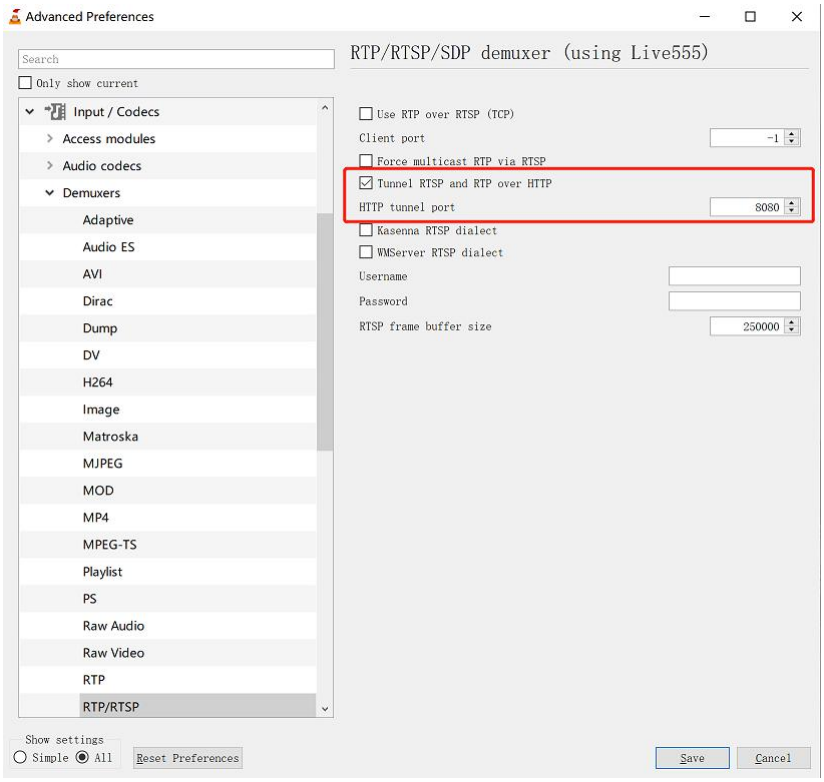
After the data exchange is complete...

Step 8. The client sends the RTSP TEARDOWN command (BASE64 encoding and) through the "command socket"

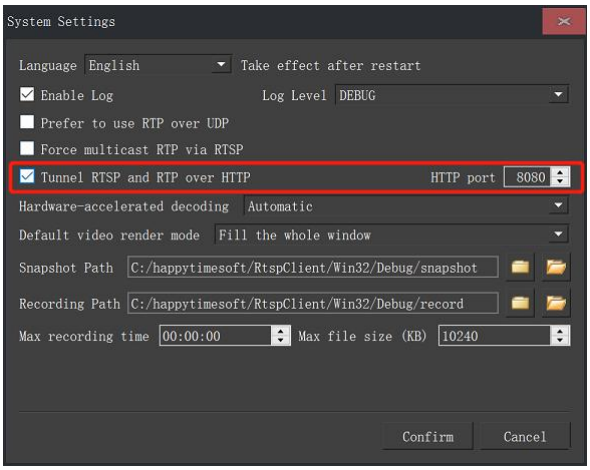
Step 9. The server responds to the RTSP TEARDOWN command (in plain text) through the "data socket".

Step 10. Close the two sockets.

VLC supports RTSP over HTTP, the settings as the follows:



Happytime rtsp client (https://happytimesoft.com/products/rtsp-client/index.html) supports RTSP over HTTP, The setting as the following:



Happytime rtsp client also supports rtsp streams starting with http:// or https://. If it starts with http://, it is considered to be a rtsp over http stream. If it starts with https://, it is considered to be a rtsp over https stream.

2.8 RTSP over Websocket

First establish an HTTP connection, and then upgrade to the websocket protocol,

RTSP over websocket protocol upgrade process:

C-->S:

GET /websocket HTTP/1.1 Host: 192.168.3.27

Upgrade: websocket Connection: Upgrade

Sec-WebSocket-Key: KSO+hOFs1q5SkEnx8bvp6w== Origin: http://192.168.3.27

Sec-WebSocket-Protocol: rtsp.onvif.org Sec-WebSocket-Version: 13

S-->C:

HTTP/1.1 101 Switching Protocols Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: G/cEt4HtsYEnP0MnSVkKRk459gM= Sec-WebSocket-Protocol: rtsp.onvif.org

Sec-WebSocket-Version: 13

After the protocol upgrade is successful, perform normal rtsp protocol exchange, and send and receive data through websocket connection.

2.9 RTP Multicast

To enable the rtp multicast function, it need to specify the <multicast> to 1 in the configuration file.

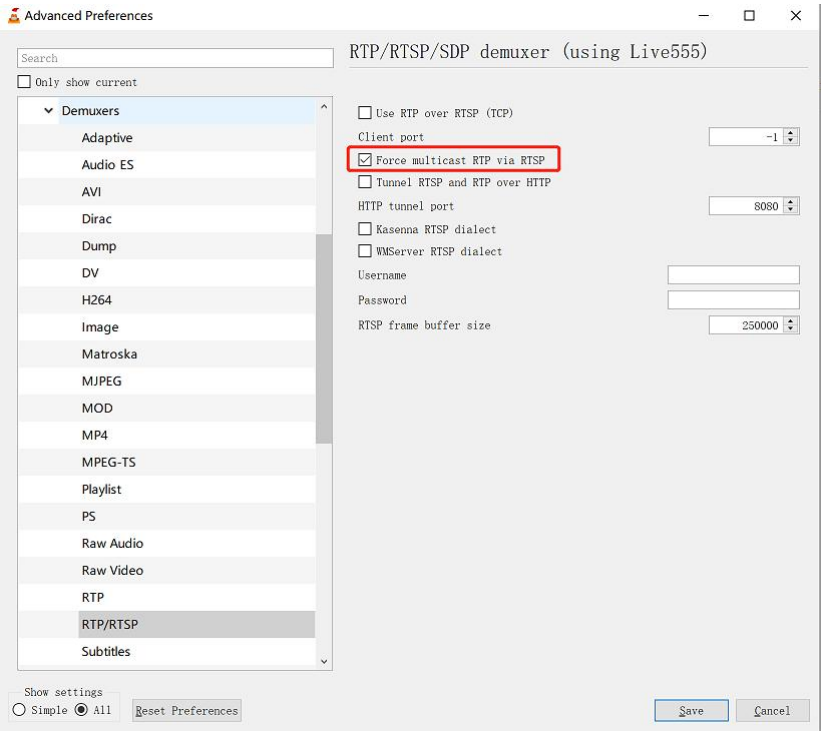
The rtsp server does not support the configuration of multicast addresses.

Different rtsp stream addresses use multicast, randomly assigned multicast addresses starting with 232.

Different rtsp sessions use rtp multicast to play the same rtsp stream, using the same multicast address. Only the first rtsp session sends audio and video data, and subsequent sessions refer to the first rtsp session.

The rtp multicast stream address is the same as the other rtsp stream address.

Use VLC to test rtp multicast, use the following settings:



2.10 Audio back channel

The backchannel connection handling is done using RTSP [RFC 2326]. Therefore a mechanism is introduced which indicates that a client wants to built up a backchannel connection. RTSP provides feature-tags to deal with such functionality additions. A device that supports bi-directional connections (e.g audio or metadata connections) shall support the introduced RTSP extensions.

2.10.1 RTSP Require- Tag

The RTSP standard [RFC 2326] can be extended by using additional headers objects. For that purpose a Require tag is introduced to handle special functionality additions (see [RFC 2326], 1.5 Extending Rtsp and 12.32 Require).

The Require-tag is used to determine the support of this feature. This header shall be included in any request where the server is required to understand that feature to correctly perform the request.

A device that supports backchannel and signals Audio output support via the AudioOutputs capability shall understand the backchannel tag:

www.onvif.org/ver20/backchannel

An RTSP client that wants to built up an RTSP connection with a data backchannel shall include the Require header in its requests.

2.10.2 Connection setup for a bi- directional connection

A client shall include the feature tag in it's DESCRIBE request to indicate that a bidirectional data connection shall be established.

A server that understands this Require tag shall include an additional media stream in its SDP file as configured in its Media Profile.

An RTSP server that does not understand the backchannel feature tag or does not support bidirectional data connections shall respond with an error code 551 Option not supported according to the RTSP standard. The client can then try to establish an RTSP connection without backchannel.

A SDP file is used to describe the session. To indicated the direction of the media data the server shall include the a=sendonly in each media section representing media being sent from the client to the server and a=recvonly attributes in each media section representing media being sent from the server to the client.

The server shall list all supported decoding codecs as own media section and the client chooses which one is used. The payload type and the encoded bitstream shall be matched with one of the a=rtpmap fields provided by the server so that the server can properly determine the audio decoder.

Example 1: Server without backchannel support:

```
Client - Server:        DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                        Cseq: 1
                        User-Agent: ONVIF Rtsp client
                        Accept: application/sdp
                        Require: www.onvif.org/ver20/backchannel

Server - Client:        RTSP/1.0 551 Option not supported
                        Cseq: 1
                        Unsupported: www.onvif.org/ver20/backchannel
```

Example 2: Server with Onvif backchannel support:

```
Client - Server:          DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                          Cseq: 1
                          User-Agent: ONVIF Rtsp client
                          Accept: application/sdp
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          Cseq: 1
                          Content-Type: application/sdp
                          Content-Length: xxx

                          v=0
                          o= 2890842807 IN IP4 192.168.0.1
                          s=RTSP Session with audiobackchannel
                          m=video 0 RTP/AVP 26
                          a=control:rtsp://192.168.0.1/video
                           a=recvonly
                           m=audio 0 RTP/AVP 0
                          a=control:rtsp://192.168.0.1/audio
                          a=recvonly
                          m=audio 0 RTP/AVP 0
                          a=control:rtsp://192.168.0.1/audioback
                          a=rtpmap:0 PCMU/8000
                          a=sendonly
```

This SDP file completely describes the RTSP session. The Server gives the client its control URLs to setup the streams.

In the next step the client can setup the sessions:

```
Client - Server:          SETUP rtsp://192.168.0.1/video RTSP/1.0
                          Cseq: 2
                          Transport: RTP/AVP;unicast;client_port=4588-4589

Server - Client:    RTSP/1.0 200 OK
                          Cseq: 2
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=4588-4589;
                          server_port=6256-6257

Client - Server:          SETUP rtsp://192.168.0.1/audio RTSP/1.0
                          Cseq: 3
                          Session: 123124
                          Transport: RTP/AVP;unicast;client_port=4578-4579

Server - Client:          RTSP/1.0 200 OK
                          Cseq: 3
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=4578-4579;
                          server_port=6276-6277

Client - Server:          SETUP rtsp://192.168.0.1/audioback RTSP/1.0
                          Cseq: 4
                          Session: 123124
                          Transport: RTP/AVP;unicast;client_port=6296-6297
                          Require: www.onvif.org/ver20/backchannel

Server - Client:          RTSP/1.0 200 OK
                          Cseq: 4
                          Session: 123124;timeout=60
                          Transport:RTP/AVP;unicast;client_port=6296-6297;
                          server_port=2346-2347
```

The third setup request establishes the audio backchannel connection.

In the next step the client starts the session by sending a PLAY request.

```
Client - Server:        PLAY rtsp://192.168.0.1 RTSP/1.0
                        Cseq: 5
                        Session: 123124
                        Require: www.onvif.org/ver20/backchannel

Server - Client:        RTSP/1.0 200 OK
                        Cseq: 5
                        Session: 123124;timeout=60
```

After receiving the OK response to the PLAY request the client MAY start sending audio data to the server. It shall not start sending data to the server before it has received the response.

The Require-header indicates that a special interpretation of the PLAY command is necessary. The command covers both starting of the video and audio stream from NVT to the client and starting the audio connection from client to server.

To terminate the session the client sends a TEARDOWN request.

```
Client - NVT:       TEARDOWN rtsp://192.168.0.1 RTSP/1.0
                    Cseq: 6
                    Session: 123124
                    Require: www.onvif.org/ver20/backchannel

NVT - Client:       RTSP/1.0 200 OK
                    Cseq: 6
                    Session: 123124
```

2.10.3 Example

Server with Onvif backchannel support (with multiple decoding capability)

If a device supports multiple audio decoders as backchannel, it can signal such capability by listing multiple a=rtpmap fields illustrated as follows.

```
Client - Server:        DESCRIBE rtsp://192.168.0.1 RTSP/1.0
                        Cseq: 1
                        User-Agent: ONVIF Rtsp client
                        Accept: application/sdp
                        Require: www.onvif.org/ver20/backchannel

Server - Client:        RTSP/1.0 200 OK
                        Cseq: 1
                        Content-Type: application/sdp
                        Content-Length: xxx

                        v=0
                        o= 2890842807 IN IP4 192.168.0.1
                        s=RTSP Session with audiobackchannel
                        m=video 0 RTP/AVP 26
                        a=control:rtsp://192.168.0.1/video
                        a=recvonly
                        m=audio 0 RTP/AVP 0
                        a=control:rtsp://192.168.0.1/audio
                        a=recvonly
                        m=audio 0 RTP/AVP 0 97 98 99 100
                        a=control:rtsp://192.168.0.1/audioback
                        a=rtpmap:0 PCMU/8000
                        a=rtpmap:97 G726-16/8000
                        a=rtpmap:98 G726-24/8000
                        a=rtpmap:99 G726-32/8000
                        a=rtpmap:100 G726-40/8000
                        a=sendonly
```

2.11 Multiple capture devices support

If your system have multiple audio capture device, you can use

rtsp://[serverip]:[serverport]/audiodeviceN

The N to specify the audio capture device index, start from 0, such as:

rtsp://192.168.0.100/audiodevice ; stream audio from the first audio device

rtsp://192.168.0.100/audiodevice1 ; stream audio from the second audio device

If your system have multiple video capture device, you can use

rtsp://[serverip]:[serverport]/videodeviceN

The N to specify the video capture device index, start from 0, such as:

rtsp://192.168.0.100/videodevice ; stream video from the first video device

rtsp://192.168.0.100/videodevice1 ; stream video from the second video device

If your system have multiple monitors, you can use

rtsp://[serverip]:[serverport]/screenliveN

The N to specify the monitor index, start from 0, such as:

rtsp://192.168.0.100/screenlive ; stream living screen from the first monitor

rtsp://192.168.0.100/screenlive1 ; stream living screen from the second monitor

The audio index or video index represents which device can run the follow command to view.

onvifrtspserver -l device

videodevice or audiodevice can also specify the device name, such as

rtsp://[serverip]:[serverport]/videodevice=testvideo

Run the *onvifrtspserver -l device* command to get the device name.

Note that there can be no spaces in the device name, if the device name contains spaces, you need to use %20 instead of spaces.

If the device name is “FaceTime HD Camera”, the rtsp stream address is:

rtsp://[serverip]:[serverport]/videodevice=FaceTime%20HD%20Camera

2.12 Capture application window

The onvif rtsp server supports capturing application windows, you can use the following command to list valid application windows:

onvifrtspserver -l window

Below is a sample output of the command

Available window name :

C:\Windows\system32\cmd.exe - OnvifRtspServer.exe -l window
user manual.doc - WPS Office
Rtsp-server Project - Source Insight - [Main.cpp]
RtspServer

You can use the following url to capture the specified application window:

rtsp://[serverip]:[serverport]/window=[window title]

Note : window title case insensitive

Such as :

rtsp://[serverip]:[serverport]/window=rtspserver

Note that there can be no spaces in the window title, if the window title contains spaces, you need to use %20 instead of spaces. Such as:

rtsp://[serverip]:[serverport]/window=user%20manual.doc%20-%20WPS%20Office

Chapter 3 Run ONVIF RTSP Server

The server is a console application.

Windows: to run the server, simply type "onvifrtspserver".

Linux: to run the server, type "./start.sh", on linux platform, the server run as deamon by default.

Onvif rtsp server supports the following command line options:

-l [device|videodevice|audiodevice|window]

-l device list available video and audio capture device

-l videodevice list available video capture device

-l audiodevice list available audio capture device

-l window list available application window

Below is sample output of -l device:

onvifrtspserver -l device

Available video capture device :

index : 0, name : FaceTime HD Camera (Built-in)

Available audio capture device :

index : 0, name : Headset Microphone (Apple Audio Device)

index : 1, name : Internal Digital Microphone (Apple Audio Device)

Note : The demo version supports up to 4 concurrent streams.

The release version supports up to 100 concurrent streams