MPEG-TS RTP stream publishing

- Overview
- Codecs supported
- Operation flowchart
- Testing
- Configuration
    - Stop stream publishing if there are no media data
    - Close subscribers sessions if publisher stops sending media data
- REST API
    - REST methods and response states
    - Parameters
- Audio only or video only publishing
- Publishing audio with various samplerates
- Renewing the stream publishing after interruption
- Publishers restriction by IP address
- H265 publishing
- Known issues

Overview

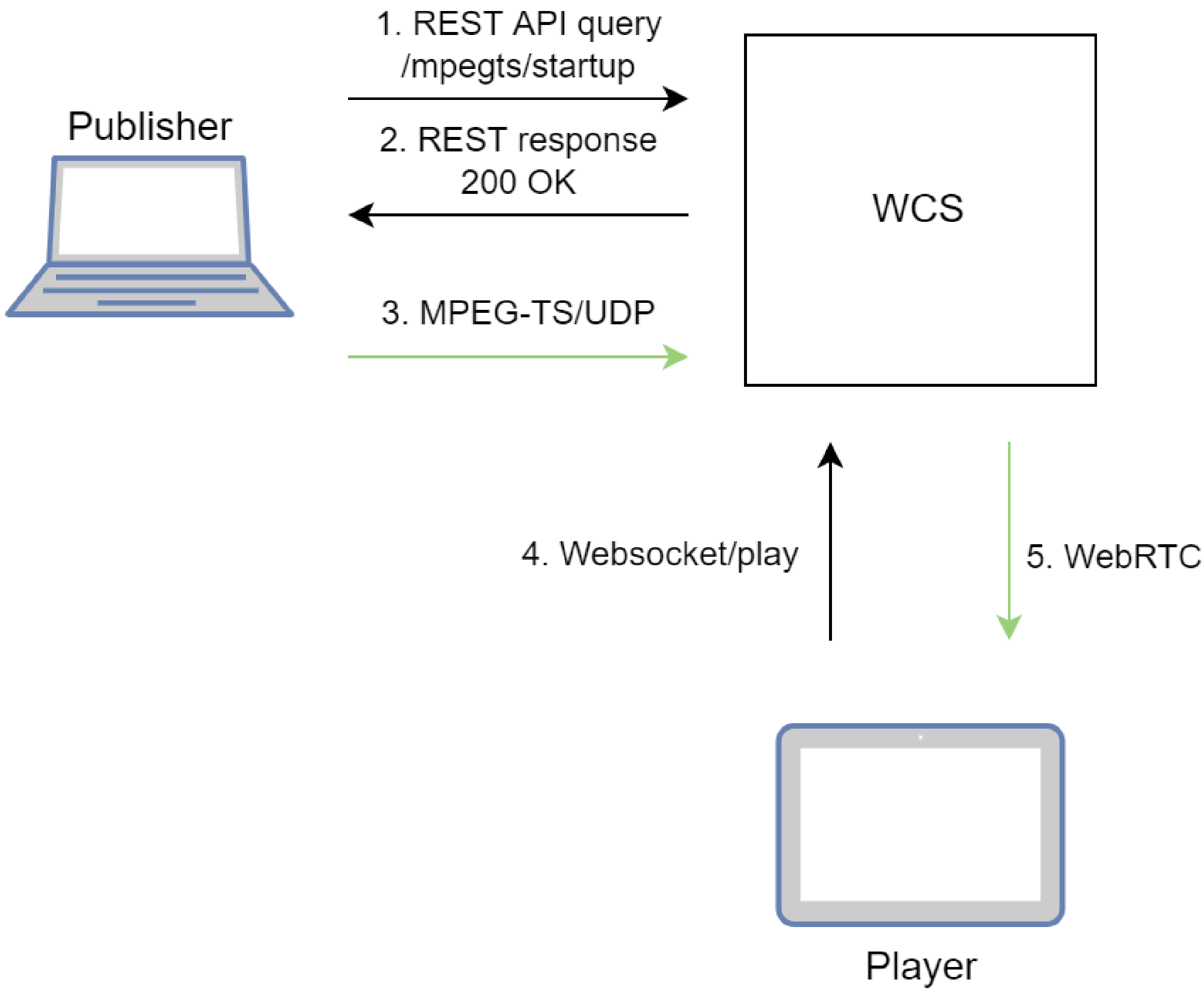
Since WCS build 5.2.1193 it is possible to publish MPEG-TS RTP stream via UDP to WCS, and since build 5.2.1253 MPEG-TS stream may be published via SRT. The feature can be used to publish H264+AAC stream from software or hardware encoder supporting MPEG-TS. Since build 5.2.1577 H265+AAC stream publishing is also allowed.

SRT protocol is more reliable than UDP, so it is recommended to use SRT for MPEG-TS publishing if possible.

Codecs supported

- H264
- H265 (since build 5.2.1577)
- AAC

Operation flowchart



1. Publisher sends REST API query `/mpegts/startup`

2. Publisher receives 200 OK with URI to publish

3. Stream is publishing to WCS using URI

4. Browser establishes Websocket connestion and sends `play` command.

5. Browser receives WebRTC stream and plays it on web page.

Testing

1. For test we use:

- WCS server
- ffmpeg to publish MPEG-TS stream
- Player web application in Chrome browser to play the stream

2. Send `/mpegts/startup` query with stream name `test`

SRT:

```
curl -H "Content-Type: application/json" -X POST http://test1.flashphoner.com:8081/rest-api/mpegts/startup -d
'{"localStreamName":"test","transport":"srt"}'
```

UDP:

```
curl -H "Content-Type: application/json" -X POST http://test1.flashphoner.com:8081/rest-api/mpegts/startup -d
'{"localStreamName":"test","transport":"udp"}'
```

Where `test1.flashphoner.com` - WCS server address

3. Receive `200 OK` response

SRT:


```
{
  "localMediaSessionId": "32ec1a8e-7df4-4484-9a95-e7eddc45c508",
  "localStreamName": "test",
  "uri": "srt://test1.flashphoner.com:31014",
  "status": "CONNECTED",
  "hasAudio": false,
  "hasVideo": false,
  "record": false,
  "transport": "SRT",
  "cdn": false,
  "timeout": 90000,
  "maxTimestampDiff": 1,
  "allowedList": []
}
```


UDP:


```
{
  "localMediaSessionId": "32ec1a8e-7df4-4484-9a95-e7eddc45c508",
  "localStreamName": "test",
  "uri": "udp://test1.flashphoner.com:31014",
  "status": "CONNECTED",
  "hasAudio": false,
  "hasVideo": false,
  "record": false,
  "transport": "UDP",
  "cdn": false,
  "timeout": 90000,
  "maxTimestampDiff": 1,
  "allowedList": []
}
```


4. Publish MPEG-TS stream using URI from the response

SRT:

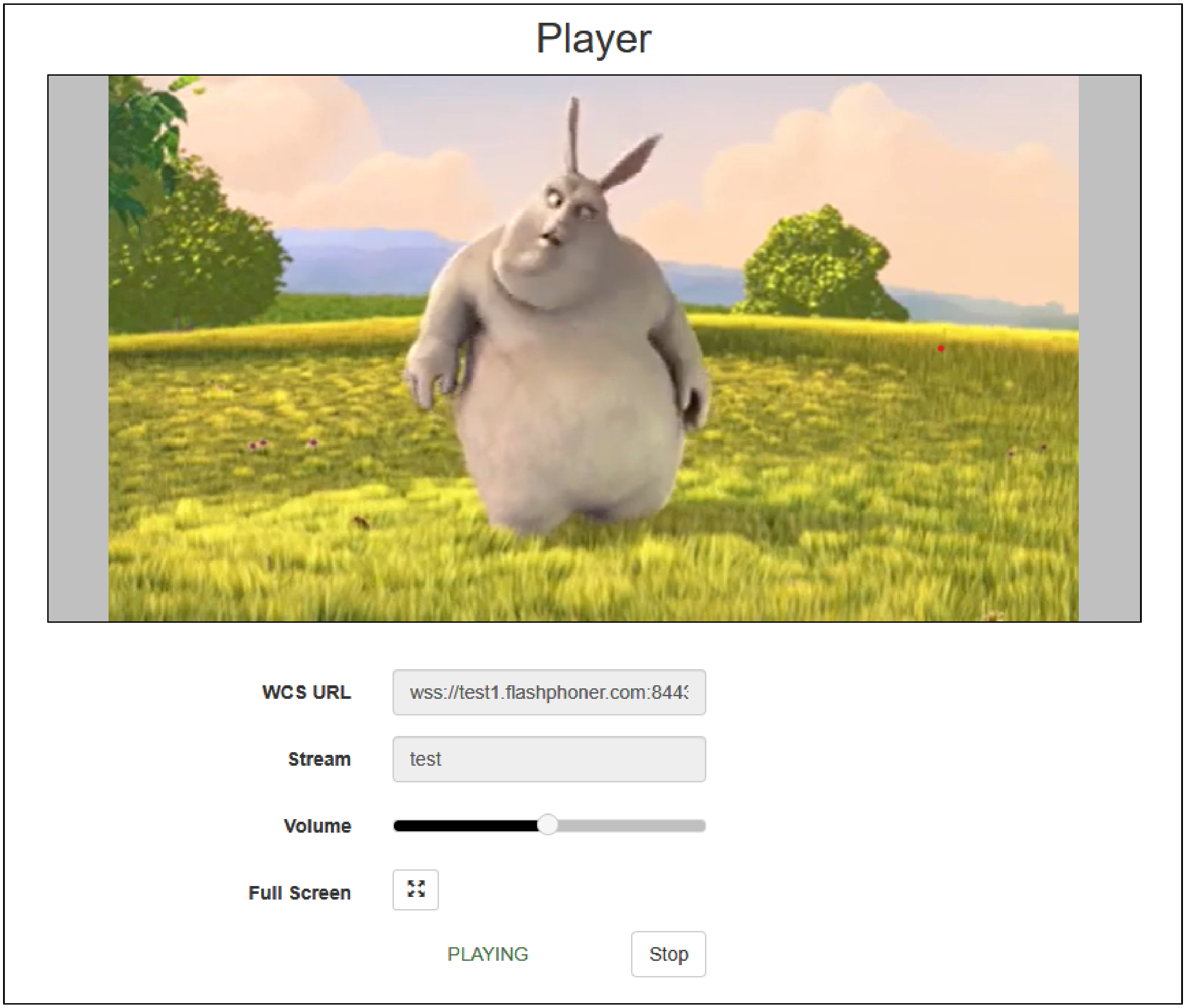
```
 ffmpeg -re -i bunny360p.mp4 -c:v libx264 -c:a aac -b:a 160k -bsf:v h264_mp4toannexb -keyint_min 60 -profile:v
baseline -preset veryfast -f mpegts "srt://test1.flashphoner.com:31014"
```

UDP:

```
 ffmpeg -re -i bunny360p.mp4 -c:v libx264 -c:a aac -b:a 160k -bsf:v h264_mp4toannexb -keyint_min 60 -profile:v
baseline -preset veryfast -f mpegts "udp://test1.flashphoner.com:31014?pkt_size=1316"
```

```
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'bunny360p.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.12.100
  Duration: 00:09:56.46, start: 0.000000, bitrate: 631 kb/s
  Stream #0:0[0x1](eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p(progressive), 640x360, 499 kb/s, 24 fps, 24 tbr, 12288 tbn (default)
    Metadata:
      handler_name    : VideoHandler
      vendor_id       : [0][0][0][0]
  Stream #0:1[0x2](eng): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 128 kb/s (default)
    Metadata:
      handler_name    : SoundHandler
      vendor_id       : [0][0][0][0]
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> h264 (libx264))
  Stream #0:1 -> #0:1 (aac (native) -> aac (native))
Press [q] to stop, [?] for help
[libx264 @ 00000249853ac540] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2
[libx264 @ 00000249853ac540] profile Constrained Baseline, level 3.0, 4:2:0, 8-bit
Output #0, mpegts, to 'udp://95.191.130.39:31006?pkt_size=1316':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf59.16.100
  Stream #0:0(eng): Video: h264, yuv420p(progressive), 640x360, q=2-31, 24 fps, 90k tbn (default)
    Metadata:
      handler_name    : VideoHandler
      vendor_id       : [0][0][0][0]
      encoder         : Lavc59.18.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
  Stream #0:1(eng): Audio: aac (LC), 48000 Hz, stereo, fltp, 160 kb/s (default)
    Metadata:
      handler_name    : SoundHandler
      vendor_id       : [0][0][0][0]
      encoder         : Lavc59.18.100 aac
frame= 5553 fps= 24 q=28.0 size=   18251kB time=00:03:51.65 bitrate= 645.4kbits/s speed=   1x
```

5. Open Player web application. Set the stream name `test` to "Stream name" field and click "Start" button. Stream playback will start



Configuration

Stop stream publishing if there are no media data

By default, MPEG-TS stream publishing will stop at server side if server doe not receive any media data from publisher in 90 seconds. The timeout is set in milliseconds by the following papameter

```
mpegts_stream_timeout=90000
```

Close subscribers sessions if publisher stops sending media data

If publisher stopped sending media data by some reason, then started again (for example, ffmpeg was restarted), the stream frame timestamps sequence is corrupting. Te stream cannot be played via WebRTC correctky in this case. As workaround, all the subscribers sessions will be closed if stream timestamps sequence corruption occurs, then all the si=ubscribers should connect to the stream again. A maximum timestamp difference is set in seconds by the following parameter

```
mpegts_max_pts_diff=1
```

REST API

A REST-query should be HTTP/HTTPS POST request as follows:

- HTTP:http://test.flashphoner.com:8081/rest-api/mpegts/startup
- HTTPS:https://test.flashphoner.com:8444/rest-api/mpegts/startup

Where:

- test.flashphoner.com - is the address of the WCS server
- 8081 - is the standard REST / HTTP port of the WCS server
- 8444 - is the standard HTTPS port
- rest-api - is the required part of the URL
- /mpegts/startup - REST mathod to use

REST methods and response states

REST method	REST query body example	REST response body example	Response states	Description

/mpegts
/startup
	{
	  "localStreamName":"
	test",
	  "transport":"srt",
	  "hasAudio": true,
	  "hasVideo": true
	}
		{
		  "localMediaSessionId": "32ec1a8e-7df4-
		4484-9a95-e7eddc45c508",
		  "localStreamName": "test",
		  "uri": "srt://192.168.1.39:31014",
		  "status": "CONNECTED",
		  "hasAudio": false,
		  "hasVideo": false,
		  "record": false,
		  "transport": "SRT",
		  "cdn": false,
		  "timeout": 90000,
		  "maxTimestampDiff": 1,
		  "allowedList": []
		}
			200 - OK
			409 - Conflict
			500 - Internal error
				Start MPEG-TS publishing

/mpegts
/find
	{
	  "localStreamName":"
	test",
	  "uri": "srt://192.
	168.1.39:31014"
	}
		[
		 {
		  "localMediaSessionId": "32ec1a8e-7df4-
		4484-9a95-e7eddc45c508",
		  "localStreamName": "test",
		  "uri": "srt://192.168.1.39:31014",
		  "status": "PROCESSED_LOCAL",
		  "hasAudio": false,
		  "hasVideo": false,
		  "record": false,
		  "transport": "SRT",
		  "cdn": false,
		  "timeout": 90000,
		  "maxTimestampDiff": 1,
		  "allowedList": []
		 }
		]
			200 – streams found
			404 – streams not found
			500 - Internal error
				Find the MPEG-TS stream by criteria

/mpegts
/find_all
		<pre>[
		 {
		  "localMediaSessionId": "32ec1a8e-7df4-4484-9a95-e7eddc45c508",
		  "localStreamName": "test",
		  "uri": "srt://192.168.1.39:31014",
		  "status": "CONNECTED",
		  "hasAudio": true,
		  "hasVideo": true,
		  "record": false,
		  "timeout": 90000,
		  "maxTimestampDiff": 90000
		 }
		]</pre>
			200 – streams found
			
			404 – streams not found
			
			500 - Internal error
				Find all MPEG-TS streams

/mpegts
/terminate
	<pre>{
	  "localStreamName":"test"
	}</pre>
			200 - stream stopped
			
			404 - stream not found
			
			500 - Internal error
				Stop MPEG-TS stream

Parameters

Name	Description	Example
localStreamName	Name to set to the stream on server	test
transport	Transport to use	srt
uri	Endpoint URI to publish the stream	udp://192.168.1.39:31014
localMediaSessionId	Stream media session Id	32ec1a8e-7df4-4484-9a95-e7eddc45c508
status	Stream status	CONNECTED
hasAudio	Stream has audio track	true
hasVideo	Stream has video track	true
record	Stream is recording	false
timeout	Maximum media data receiving timeout, ms	90000
maxTimestampDiff	Maximum stream timestamps difference, s	1
allowedList	Client addresses list which are allowed to publish the stream	`["192.168.1.0/24"]`

Audio only or video only publishing

Since build 5.2.1253, audio only or video only stream can be published using REST API query `/mpegts/startup` parameters

- video only stream publishing

```
{
  "localStreamName":"mpegts-video-only",
  "transport":"srt",
  "hasAudio": false
}
```

- audio only stream publishing

```
{
  "localStreamName":"mpegts-audio-only",
  "transport":"srt",
  "hasVideo": false
}
```

Publishing audio with various samplerates

By default, the following video and audio parameters are used to publish MPEG-TS stream

```
v=0
o=- 1988962254 1988962254 IN IP4 0.0.0.0
c=IN IP4 0.0.0.0
t=0 0
a=sdplang:en
m=audio 1 RTP/AVP 102
a=rtpmap:102 mpeg4-generic/44100/2
a=sendonly
m=video 1 RTP/AVP 119
a=rtpmap:119 H264/90000
a=sendonly
```

Video track must be published in H264 codec with clock rate 90000 Hz, audio track must be published in AAC with samplerate 44100 Hz, two channels.

An additional samplerates or one channel may be enabled for audio publishing if necessary. Do the following to enable:

1. Create the file `mpegts_agent.sdp` in`/usr/local/FlashphonerWebCallServer/conf` folder

```
sudo touch /usr/local/FlashphonerWebCallServer/conf/mpegts_agent.sdp
```

2. Add necessary SDP parameters to the file

```
sudo nano /usr/local/FlashphonerWebCallServer/conf/mpegts_agent.sdp
```

for example

```
v=0
o=- 1988962254 1988962254 IN IP4 0.0.0.0
c=IN IP4 0.0.0.0
t=0 0
a=sdplang:en
m=audio 1 RTP/AVP 102 103 104
a=rtpmap:102 mpeg4-generic/44100/2
a=rtpmap:103 mpeg4-generic/48000/2
a=rtpmap:104 mpeg4-generic/32000/1
a=sendonly
m=video 1 RTP/AVP 119
a=rtpmap:119 H264/90000
a=sendonly
```

3. Set the necessary permissions and restart WCS to apply changes

```
sudo nano /usr/local/FlashphonerWebCallServer/bin/webcallserver set-permissions
sudo systemctl restart webcallserver
```

Renewing the stream publishing after interruption

A separate UDP port is opened for every MPEG-TS publishing session to accept client connection (for SRT only) and receive media traffic. Due to security reasons, since build 5.2.1299, the stream will be stopped on server if client stops publishing (like WebRTC one), and publisher can't connect and send traffic to the same port. All the stream viewers will receive `STREAM_STATUS.FAILED` in this case. A new REST API query should be used to renew the stream publishing, with the same name if necessary.

Publishers restriction by IP address

Since build 5.2.1314 it is possible to restrict client IP addresses which are allowed to publish MPEG-TS stream via UDP using REST API `/mpegts/startup` query parameter

```
{
  "localStreamName":"mpegts-stream",
  "transport":"udp",
  "allowedList": [
    "192.168.0.100",
    "172.16.0.1/24"
  ]
}
```

Since build 5.2.1485 MPEG-TS via SRT publishers may also be restricted

```
{
  "localStreamName":"mpegts-stream",
  "transport":"srt",
  "allowedList": [
    "192.168.0.100",
    "172.16.0.1/24"
  ]
}
```

The list may contain both exact IP addresses and address masks. If REST API query contains a such list, only the clients with IP addresses matching the list can publish the stream.

H265 publishing

Since build 5.2.1577 it is possible to publish MPEG-TS H265+AAC stream. H265 codec should be set in `mpegts_agent.sdp` file:

```
v=0
o=- 1988962254 1988962254 IN IP4 0.0.0.0
c=IN IP4 0.0.0.0
t=0 0
a=sdplang:en
m=audio 1 RTP/AVP 102
a=rtpmap:102 mpeg4-generic/48000/2
a=sendonly
m=video 1 RTP/AVP 119
a=rtpmap:119 H265/90000
a=sendonly
```

Since build 5.2.1598, WCS supports MPEG-TS stream publishing both in H264 and H265 codecs by default without SDP settings change.

H265 must also be added to supported codecs list

```
codecs=opus,alaw,ulaw,g729,speex16,g722,mpeg4-generic,telephone-event,h264,vp8,flv,mpv,h265
```

and to exclusion lists

```
codecs_exclude_sip=mpeg4-generic,flv,mpv,h265
codecs_exclude_sip_rtmp=opus,g729,g722,mpeg4-generic,vp8,mpv,h265
codecs_exclude_sfu=alaw,ulaw,g729,speex16,g722,mpeg4-generic,telephone-event,flv,mpv,h265
```

H265 publishing example using ffmpeg

```
ffmpeg -re -i source.mp4 -c:v libx265 -c:a aac -ar 48000 -ac 2 -b:a 160k -bsf:v hevc_mp4toannexb -keyint_min 120 -profile:v main -preset veryfast -x265-params crf=23:bframes=0 -f mpegts "srt://test.flashphoner.com:31014"
```

H265 will be transcoded to H264 or VP8 to play it from server!

Known issues

1. When MPEG-TS stream publishing via UDP is stopped at server side via REST API query `/mpegts/terminate` , publishing encoder still sends media data

Symptoms: ffmpeg still sends data via UDP when MPEG-TS stream publishing is stopped on server

Solution: this is normal behaviour for UDP because the protocol itself provides no any methods to let publisher know the UDP port is already closed. Use SRT which handles the case correctly if possible.