User manual

(RTMP Server)

HAPPYTIME SOFTWARE CO., LIMITED

Declaration

www.happytimesoft.com

Table of Contents

Chapter 1 Introduction

Happytime rtmp server is a simple, lightweight, high-performance, and stable stream server.

It can be used to stream local media files, living screen, application windows, camera, microphone, live video/audio content to adobe flash player clients over RTMP protocol.

It developed based on C/C++, the code is stable and reliable, cross-platform porting is simple and convenient, and the code is clear and concise. The server is written to be lightweight and easy to understand, while having good performance, very low latency, video opened immediately.

Happytime rtmp server supports linux, windows, macos, ios, android, embeded linux platforms, supports cross-compiler, can be easily ported to other platforms.

Chapter 2 Key features

RTMP live streaming

Support variety of audio and video files

Support push video from camera and living screen

Support push video from application window

Support push audio from audio device

Support recording system audio on Windows

Support RTMP pusher

Support video codec H264,H265

Support audio codec AAC,G711A,G711U

Support automatic transcoding function

Support RTSP/SRT stream to RTMP stream

Support RTSP/RTMP/SRT relay

Support for configuring audio and video output parameters

Small size, suitable for embedded development

Code structure clear, easy to use

Chapter 3 Configuration

If no configuration file is specified at startup, the default configuration file rtmpserver.cfg will be used.

3.1 Configuration Templates

```
<?xml version="1.0" encoding="utf-8"?>
<config>
    <serverip></serverip>
    <serverport>1935</serverport>
    <ipv6_enable>1</ipv6_enable>
    <loop_nums>-1</loop_nums>
    <log_enable>1</log_enable>
    <log_level>1</log_level>

    <http_notify>
        <on_connect></on_connect>
        <on_play></on_play>
        <on_publish></on_publish>
        <on_done></on_done>
        <notify_method></notify_method>
    </http_notify>

    <application>
        <name>myapp</name>
        <output>
            <url></url>
            <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>
            <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>
    </application>
</config>
```

3.2 Configuring Node Description

3.2.1 System parameters

<serverip>

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

<serverport>

Specify the RTMP server service port, the default is 1935.

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

<ipv6_enable>

Indicates whether IPv6 is enabled, 0-disable, 1-enable.

Note: If the server does not specify a server ip in <serverip> and the <ipv6_enable> is 1, and the server has an IPv6 address, the client can connect to the server through the IPv6 address.

<loop_nums>

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

<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

3.2.2 http_notify

<http_notify> : Specify the HTTP notification callback address.

<on_connect> :

Sets HTTP connection callback. When clients issues connect command an HTTP request is issued and command processing is suspended until it returns result code. If HTTP 200 code is returned then RTMP session continues.

HTTP request receives a number of arguments. POST method is used with application/x-www-form-urlencoded MIME type. The following arguments are passed to caller:

protocol=rtmp

call=connect

addr - client IP address

app - application name

tcurl - tcUrl

name - stream name

clientid - rtmp session id

<on_play> :

Sets HTTP play callback. Each time a clients issues play command an HTTP request is issued and command processing is suspended until it returns result code. If HTTP 200 code is returned then RTMP session continues.

HTTP request receives a number of arguments. POST method is used with application/x-www-form-urlencoded MIME type. The following arguments are passed to caller:

protocol=rtmp

call=play

addr - client IP address

app - application name

tcurl - tcUrl

name - stream name

clientid - rtmp session id

<on_publish> :

The same as on_play above with the only difference that this node sets callback on publish command. Instead of remote pull push is performed in this case.

<on_done> :

Sets play/publish terminate callback. All the above applies here. However HTTP status code is not checked for this callback.

<notify_method>

Sets HTTP method for notifications. Default is POST with application/x-www-form-urlencoded content type.

Support GET and POST method.

3.2.3 application node

<application> : it can configure multiple nodes

<name>: Application Name

The rtmp stream address is:

rtmp://[serverip]:[serverport]/[application-name]/FILENAME

The [application-name] is name tag value.

3.2.3.1 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 live 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 (Non-RTMP standard, custom extension function)

<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 if 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

AAC: output AAC 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)

Note : G711A and G711U only support 8000 samplerate

<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)

<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 if transcoding is required.

3.2.3.2 Proxy node

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

<suffix>

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

rtmp://[serverip]:[serverport]/[application-name]/[suffix]

<url>

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

<user> <pass>

Specify the original rtsp/rtmp/srt stream or http mjpeg stream address login user and password information

<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 RTP 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 stream. If it appears and the configured parameters are inconsistent with the parameters of the original RTSP/RTMP/SRT stream, then the transcode output is performed.

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

Chapter 4 Run RTMP Server

The rtmp server is a console application.

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

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

rtmp server supports the following command line options:

-c config   specify the configuration file

-c option specifies the configuration file,if not specified, the default configuration rtmpserver.cfg is used.

-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:

rtmpserver -l device

```
Avaiable video capture device :
index : 0, name : FaceTime HD Camera (Built-in)

Avaiable 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.

Chapter 5 Multiple capture devices support

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

rtmp://[serverip]:[serverport]/[application-name]/audiodeviceN

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

*rtmp://192.168.0.100/*myapp*/audiodevice* *; stream audio from the first audio device*

*rtmp://192.168.0.100/*myapp*/audiodevice1* *; stream audio from the second audio device*

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

rtmp://[serverip]:[serverport]/[application-name]/videodeviceN

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

*rtmp://192.168.0.100/*myapp*/videodevice* *; stream video from the first video device*

*rtmp://192.168.0.100/*myapp*/videodevice1* *; stream video from the second video device*

3. If your system have multiple monitors, you can use

rtmp://[serverip]:[serverport]/[application-name]/screenliveN

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

*rtmp://192.168.0.100/*myapp*/screenlive* *; stream living screen from the first monitor*

*rtmp://192.168.0.100/*myapp*/screenlive1* *; stream living screen the second monitor*

The audio index or video index represents which device can run ***rtmpserver -l device*** to view.

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

rtmp://[serverip]:[serverport]/[application-name]/videodevice=testvideo

Run the ***rtmpserver -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 (Built-in)”, the rtmp stream address is:

rtmp://[serverip]:[serverport]/[application-name]/videodevice=FaceTime%20HD%20Camera%20(Built-in)

Chapter 6 Capture application window

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

```
rtmpserver -l window
```

Below is a sample output of the command

```
Available window name :

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

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

```
rtmp://[serverip]:[serverport]/[application-name]/window=[window title]
```

Note : window title case insensitive

Such as :

```
rtmp://[serverip]:[serverport]/[application-name]/window=rtmpserver
```

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:

```
rtmp://[serverip]:[serverport]/[application-name]/window=user%20manual.doc%20-%20WPS%20Office
```

Chapter 7 Support URL parameters

RTMP server supports URL parameters, with the following format:

Taking playing the test.mp4 file as an example:

rtmp://[serverip]:[serverport]/[application-name]/test.mp4?param1=value1&param2=value2

Pararm1 and param2 represent URL parameters, while value1 and value2 represent the values of param1 and param2.

Note：The parameter value specified through the URL has a higher priority than the parameters configured in the configuration file.

The RTMP server supports the following parameters:

ve : Specify video encoding

Possible values:

H264

H265

fps ：Specify video frame rate

w : Specify video width

h : Specify video height

vb ：Specify video bitrate

ae ：Specify audio encoding

Possible values:

PCMU : g711 ulaw

PCMA: g711 alaw

AAC

sr : Specify audio samplerate

ch ：Specify the number of audio channels

ab ：Specify audio bitrate

Example:

Output video encoding as H264, resolution as 1920 * 1080, frame rate as 30, bit rate as 4000K, audio as AAC, sampling rate as 44100, stereo RTMP stream:

rtmp://[serverip]:[serverport]/[application-name]/test.mp4?ve=h264&w=1920&h=1080&fps=30&vb=4000&ae=aac&sr=44100&ch=2