User manual

（SRT 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 Introduction

Happytime srt server is a live streaming server for low latency based on Secure Reliable Tranport(SRT). It 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 over SRT 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 srt server supports linux, windows, macos, ios, android, embeded linux platforms, supports cross-compiler, can be easily ported to other platforms.

Chapter 2 Key features

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 SRT pusher

Support video codec H264,H265

Support audio codec AAC

Support automatic transcoding function

Support RTSP/RTMP stream to SRT 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 srtserver.cfg will be used.

3.1 Configuration Templates

```
<?xml version="1.0" encoding="utf-8"?>
<config>
    <serverip></serverip>
    <serverport>8080</serverport>
    <loop_nums>-1</loop_nums>
    <log_enable>1</log_enable>
    <log_level>1</log_level>

    <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>1</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 SRT server, if not specified, the SRT server will listen to all network interfaces.

<serverport>

Specify the SRT server service port, the default is 8080.

<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 application node

<application> : it can configure multiple nodes

<name>: Application Name

The srt stream address is

srt://[serverip]:[serverport]?streamid=[application-name]/FILENAME

The [application-name] is name tag value.

3.2.2.1 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

<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 25, 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:

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)

<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.

3.2.2.2 Proxy node

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

<suffix>

Specify the srt stream suffix, you can play the proxy stream from: srt://[serverip]:[serverport]?streamid=[application-name]/[suffix]

<url>

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

<user> <pass>

Specify the original rtsp/rtmp 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 SRT Server

The SRT server is a console application.

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

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

srt 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 srtserver.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:

srtserver -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

srt://[serverip]:[serverport]?streamid=[application-name]/audiodeviceN

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

*srt://192.168.0.100?streamid=*myapp*/audiodevice ; stream audio from the first audio device*

*srt://192.168.0.100?streamid=*myapp*/audiodevice1 ;stream audio from the second audio device*

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

srt://[serverip]:[serverport]?streamid=[application-name]/videodeviceN

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

*srt://192.168.0.100?streamid=*myapp*/videodevice ; stream video from the first video device*

*srt://192.168.0.100?streamid=*myapp*/videodevice1 ; stream video from the second video device*

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

srt://[serverip]:[serverport]?streamid=[application-name]/screenliveN

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

*srt://192.168.0.100?streamid=*myapp*/screenlive ; stream living screen from the first monitor*

*srt://192.168.0.100?streamid=*myapp*/screenlive1; stream living screen from the second monitor*

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

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

srt://[serverip]:[serverport]?streamid=[application-name]/videodevice=testvideo

Run the ***srtserver -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 srt stream address is:

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

Chapter 6 Capture application window

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

srtserver -l window

Below is a sample output of the command

Available window name :

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

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

srt://[serverip]:[serverport]?streamid=[application-name]/window=[window title]

Note : window title case insensitive

Such as :

srt://[serverip]:[serverport]?streamid=[application-name]/window=srtserver

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:

srt://[serverip]:[serverport]?streamid=[application-name]/window=user%20manual.doc%20-%20WPS%20Office

Chapter 7 Push data to SRT server

7.1 push with ffmpeg

You can push camera live stream by FFMPEG.Please download ffmpeg sourcecode from https://github.com/FFmpeg/FFmpeg, then compile FFMPEG with --enable-libsrt.

use ffmpeg to push camera stream with SRT (on mac platform):

$ ./ffmpeg -f avfoundation -framerate 30 -i "0:0" -vcodec libx264 -preset ultrafast -tune zerolatency -flags2 local_header -acodec libmp3lame -g 30 -pkt_size 1316 -flush_packets 0 -f mpegts "srt://[serverip]:[serverport]?streamid=[application-name]/live,m=publish"

play the SRT stream with ffplay:

./ffplay -fflags nobuffer -i "srt://[serverip]:[serverport]?streamid=[application-name]/live"

7.2 push with OBS

The OBS supports srt protocol to publish stream when version is later than v25.0.

you can use the following url whith custom service:

srt://[serverip]:[serverport]?streamid=[application-name]/live,m=publish