Video Streaming Service Wrapper Service

High Level Definition

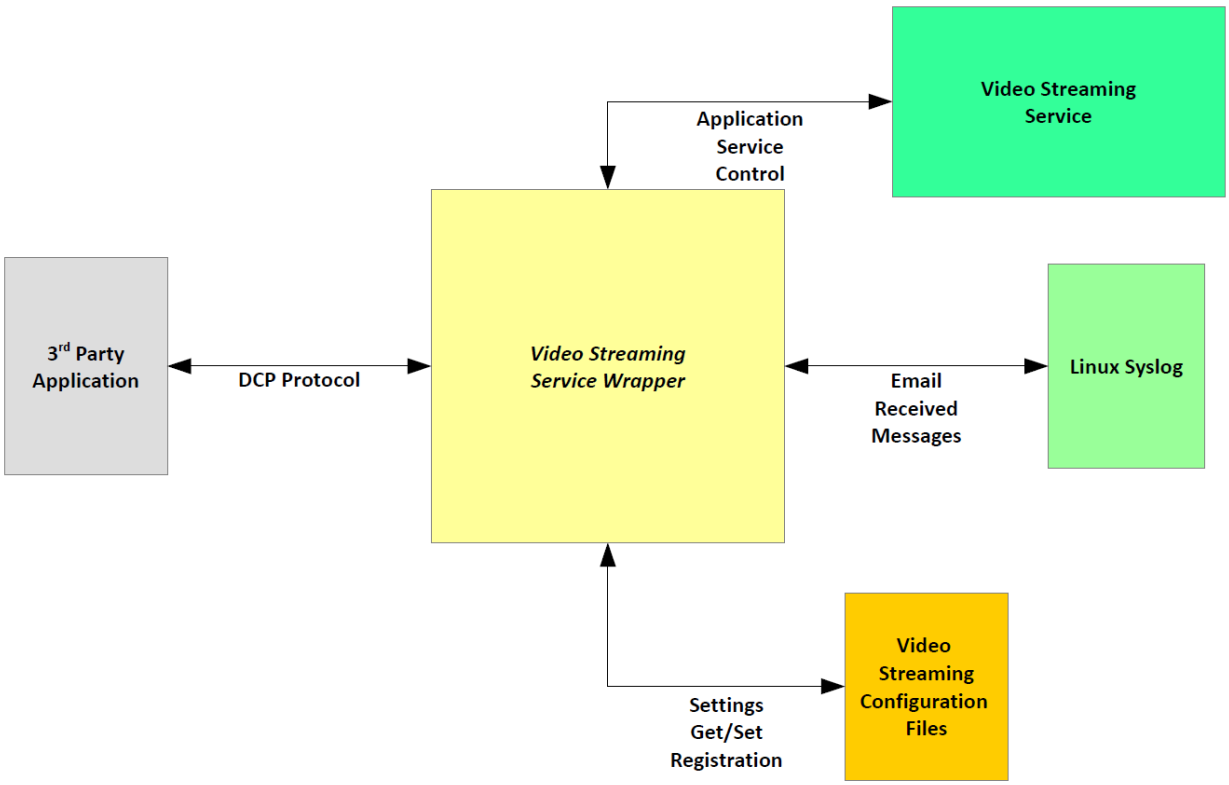
The required software is basically a wrapper to configure and manage an open-source application solution that can capture a video stream and re-stream it without too much buffering. It is basically needed to forward IP camera data.

This open-source application must be wrapped with a Python service and run on Linux (Debian 10). A 3rd party application will configure and run the Video Streaming service without bothering with the internal details of configuring/working with the Video Streaming service.

Apart from configuring the Video Streaming server, the required software is responsible of getting “stream X connected” events from the Video Streaming server through its syslog messages. For this the Linux syslog messages can be relayed to the wrapper or the postfix server can be configured to send syslog messages to the Video Streaming wrapper directly.

The software needs to communicate with a 3rd party application through UDP via a JSON based message protocol. Over this protocol messages such as “set the email domain name to abc@testing.com”, “restart video streaming server”, etc.. will be sent and the wrapper is expected to run these messages properly.

The software will be used as an always-on service that will run on a Debian 10 Linux distribution. The programming language must be Python 3.7.



General Requirements

1. The software must be written in Python 3.7.
2. The software must run on Debian 10 with minimal dependencies.
3. The software will run as a service.
4. The software will manage a video streaming service.
5. The software will receive commands from a 3rd party application via UDP using a proprietary JSON based protocol named DCP.
6. All of the DCP protocol commands detailed on the next section must be implemented by the Video Streaming Service wrapper.
7. The wrapper must configure Postfix/Dovecot server settings to the appropriate Video Streaming Service configuration files.
8. The wrapper must be able to start/restart/stop the Video Streaming Service server accordingly for the configuration to work.
9. The software must get events from the Video Streaming Service server and relay these events to the 3rd party software through the DCP protocol. The commands are detailed on the next section.
10. The following video stream protocols must be supported to accept:
    a. RTSP
    b. RTP
    c. RTMP
    d. MPEG-TS
    e. SRT
    f. HLS
11. Received video streams must be re-streamed in the following protocol:
    a. RTSP
    b. RTP
    c. RTMP
    d. MPEG-TS
    e. SRT
    f. HLS
12. The following codes for video must be supported:
    a. MJPEG
    b. MPEG4
    c. H264
    d. H265
13. The following codes for audio must be supported:
    a. G726
    b. AAC
    c. G711

DCP Protocol Requirements

1. The DCP protocol messages must be sent over UDP through port 1234.
2. Every message must be in a valid JSON format.

3. The JSON data for a message from the 3rd party application to the python wrapper must include the “command” and “arguments” fields. The “arguments” field is a JSON object with all the arguments provided as key:value pairs.
4. The JSON command format must comply with the “Example DCP Command Message” section below.
5. Events can be sent from the python wrapper to the 3rd party application and these messages must include the “event” and “arguments” fields. The “arguments” field is a JSON object with all the arguments provided as key:value pairs.
6. The JSON event format must comply with the “Example DCP Event Message” section below.

Example DCP Command Message

```
{
        “command”:”set-domain”,
        ”arguments”:
        {
                “domain-name”:”testing”,
                “domain-fqdn”:”testing.rovenma.com”,
                “domain-ip”:”192.168.1.1”
        }
}
```

Example DCP Event Message

```
{
        “event”:”email-received”,
        ”arguments”:
        {
                “email-from”:”testing”,
                “email-to”:”testing.rovenma.com”,
                “file-path”:”/var/mail/etc..etc../”
        }
}
```

DCP Commands and Events

1. When a stream is received and event must provide the ip/port information along with the format of the stream (protocol/codec/etc..).
2. The protocol/video codec/audio codec for the re-streaming operation must be configurable via a command.
3. There should be a command to start, stop and restart the Video Streaming application.
4. There should be a command to terminate a stream.
5. There should be a command to set the re-stream ip/port and similar information.
6. All commands must return a success/fail message is JSON.