# Setting up OpenSIPS and RTPProxy on Amazon AWS

In the example below OpenSIPS and RTPProxy serves as a full SIP/RTP proxy for a SBC located on the same private network on the Amazon AWS.

The server OS is Ubuntu 18.04 LTS

## Installation

### OpenSIPS installation

In our example we will Install OpenSIPS v.2.4.3 from sources although the setup should be similar for any series v2.x.x version.

Make sure we have all prerequisites:

```
apt install build-essential bison flex libncurses5-dev
```

Download from official sources and build:

```
wget http://opensips.org/pub/opensips/2.4.x/opensips-2.4.3.tar.gz
tar -xvzf opensips-2.4.3.tar.gz
cd opensips-2.4.3
make all
make menuconfig
make install
```

When build finished, verify if opensips binary starts. It may complain about incorrect configuration and exit:

```
opensips
```

### RTPProxy installation

Download RTPProxy from Github and compile from sources:

```
git clone https://github.com/sippy/rtpproxy
git -C rtpproxy submodule update --init --recursive
cd rtpproxy/
./configure
make
make install
```

# Configuration

## OpenSIPS configuration

OpenSIPS configuration file is located at /usr/local/etc/opensips/opensips.cfg

It can be generated automatically from installation option make menuconfig

The following sections below are relavant for our setup of full SIP/RTP proxy. Add/modify them to your opensips.cfg file:

Addresses. Public and private IP addresses of server instance:

```
# public IP address of opensips instance
advertised_address="18.X.X.X"

alias="18.X.X.X"


# private address of opensips instance
listen=udp:172.X.X.X:5060
```

Connection to RTPProxy. OpenSIPS needs to know the socket where RTPProxy listens to communicate with it to do address rewrite:

```
loadmodule "rtpproxy.so"
modparam("rtpproxy", "rtpproxy_sock", "udp:localhost:12221") # address and
port of rtpproxy socket
```

Now, inside your "route" section you have your main routing script. Your configuration may vary depending on your setup, so we will just add a call to a subroutine whenever valid route is found and selected by script:

```
route(relay);
```

And add the subrouting block which will handle it:

```
route[relay] {
        # for INVITEs enable some additional helper routes
        if (is_method("INVITE")) {
                if(has_body("application/sdp")){
                        xlog("we have sdp on this $rm");
                        rtpproxy_offer("cro");
                }
                t_on_reply("handle_nat");
                t_on_failure("missed_call");
        }

        if (isflagset(NAT)) {
                add_rr_param(";nat=yes");
        }
        if (!t_relay()) {
                send_reply("500","Internal Error");
        }
        exit;
}
```

As you can see from the piece above, for every inbound call OpenSIPS will request RTPProxy to rewrite SDP body.

Below code block is responsible for modifying all replies in the dialog. It makes RTPProxy to offer external public IP to the remote client.

```
onreply_route[handle_nat] {

        if (nat_uac_test("1")) {

                xlog("onreply route handle NAT - fix contact");

                fix_nated_contact();

        }
        # we receive a reply, we need to check application/sdp

        # on our body, if we have, we answer that

        if(is_method("ACK") && has_body("application/sdp")){

                rtpproxy_answer("roc");

        }else if(has_body("application/sdp")){

                # offering rtpproxy on a non ack message

                rtpproxy_offer("roc","18.X.X.X");  # real external IP
address of opensips here

        }

        xlog("incoming reply\n");

}
```

Finally, add the failure block if not present:

```
failure_route[missed_call] {

        if (t_was_cancelled()) {

                exit;

        }
}
```

## RTPProxy configuration

RTPProxy does not have configuration file. Instead it takes options as CLI parameters.

For our current setup, RTPProxy should be started like this:

```
rtpproxy -s udp:localhost:12221 -F -A 172.X.X.X/18.X.X.X -l 172.X.X.X/127.
0.0.1 -d INFO -m 16384 -M 32768
```

Important options to configure:

-s  - UDP socket to connect to OpenSIPS. Should match with the setting in OpenSIPS configuration file

-A - Internal / external IP addresses

-l - IP addresses of RTPProxy itself

# Running

The configuration is complete. Start opensips in foreground mode and check its output. It should be able to establish the connection with RTPProxy :

```
opensips -D
...
...
Jan 25 16:02:50 [31781] DBG:rtpproxy:connect_rtpproxies: [RTPProxy] set
list 0x7f01bc0edae0
Jan 25 16:02:50 [31781] DBG:rtpproxy:connect_rtpproxies: [Re]connecting
sockets (1 > 0)
Jan 25 16:02:50 [31781] DBG:rtpproxy:connect_rtpproxies: connected
localhost:12221
...
```

Once done, start OpenSIPS as a daemon:

```
opensips
```