Enable WebRTC Audio/Video

- Turn Servers
  - Install Coturn Turn Server

Turn Servers

Install Coturn Turn Server

https://www.youtube.com/embed/eZ9Jrxy6NVM

What you'll need

- A VPS is HIGHLY recommended, that gives you a public IP address and control over the firewall
- Also, to make this very easy, I recommend a debian based distro.
- A Domain name for your server which you control the A-Records for
- About 15 minutes of your time

Install Coturn

We will install Coturn from the standard repositories for Debian

```
sudo apt install coturn -y
```

Next, we need to setup our Coturn configuration file. This file will tell Coturn what attributes it should work with. This is an extremely long file. It is very well documented as to what most settings mean.

```
sudo nano /etc/turnserver.conf
```

The contents I have for this file are as follows:

```
# Coturn TURN SERVER configuration file
#
# Boolean values note: where a boolean value is supposed to be used,
# you can use '0', 'off', 'no', 'false', or 'f' as 'false',
# and you can use '1', 'on', 'yes', 'true', or 't' as 'true'
# If the value is missing, then it means 'true' by default.
#

# Listener interface device (optional, Linux only).
# NOT RECOMMENDED.
#
```

```
listening-device=eth0

# TURN listener port for UDP and TCP (Default: 3478).
# Note: actually, TLS & DTLS sessions can connect to the
# "plain" TCP & UDP port(s), too - if allowed by configuration.
#
listening-port=3478

# TURN listener port for TLS (Default: 5349).
# Note: actually, "plain" TCP & UDP sessions can connect to the TLS & DTLS
# port(s), too - if allowed by configuration. The TURN server
# "automatically" recognizes the type of traffic. Actually, two listening
# endpoints (the "plain" one and the "tls" one) are equivalent in terms of
# functionality; but Coturn keeps both endpoints to satisfy the RFC 5766 specs.
# For secure TCP connections, Coturn currently supports
# TLS version 1.0, 1.1 and 1.2.
# For secure UDP connections, Coturn supports DTLS version 1.
#
tls-listening-port=5349

# Alternative listening port for UDP and TCP listeners;
# default (or zero) value means "listening port plus one".
# This is needed for RFC 5780 support
# (STUN extension specs, NAT behavior discovery). The TURN Server
# supports RFC 5780 only if it is started with more than one
# listening IP address of the same family (IPv4 or IPv6).
# RFC 5780 is supported only by UDP protocol, other protocols
# are listening to that endpoint only for "symmetry".
#
alt-listening-port=0

# Alternative listening port for TLS and DTLS protocols.
# Default (or zero) value means "TLS listening port plus one".
#
alt-tls-listening-port=0

# Listener IP address of relay server. Multiple listeners can be specified.
# If no IP(s) specified in the config file or in the command line options,
# then all IPv4 and IPv6 system IPs will be used for listening.
#
```

```
#listening-ip=172.17.19.101
#listening-ip=10.207.21.238
#listening-ip=2607:f0d0:1002:51::4
listening-ip=<your VPS public IP address here eg: 211.151.62.140>


# Relay address (the local IP address that will be used to relay the
# packets to the peer).
# Multiple relay addresses may be used.
# The same IP(s) can be used as both listening IP(s) and relay IP(s).
#
# If no relay IP(s) specified, then the turnserver will apply the default
# policy: it will decide itself which relay addresses to be used, and it
# will always be using the client socket IP address as the relay IP address
# of the TURN session (if the requested relay address family is the same
# as the family of the client socket).
#
relay-ip=<your VPS public IP address here eg: 211.151.62.140>
#relay-ip=<ipv6 here>


# Lower and upper bounds of the UDP relay endpoints:
# (default values are 49152 and 65535)
#
min-port=49152
max-port=65535

# Uncomment to run TURN server in 'normal' 'moderate' verbose mode.
# By default the verbose mode is off.
#verbose

# Uncomment to run TURN server in 'extra' verbose mode.
# This mode is very annoying and produces lots of output.
# Not recommended under normal circumstances.
#
#Verbose

# Uncomment to use fingerprints in the TURN messages.
# By default the fingerprints are off.
#
```

```
#fingerprint

# Uncomment to use long-term credential mechanism.
# By default no credentials mechanism is used (any user allowed).
#
#lt-cred-mech
#user=someawesomeusername:aBunchOfLettersAndNumbersMixedTogetherHere41

# This option is the opposite of lt-cred-mech.
# (TURN Server with no-auth option allows anonymous access).
# If neither option is defined, and no users are defined,
# then no-auth is default. If at least one user is defined,
# in this file, in command line or in usersdb file, then
# lt-cred-mech is default.
#
#no-auth

# Be aware that use-auth-secret overrides some parts of lt-cred-mech.
# The use-auth-secret feature depends internally on lt-cred-mech, so if you set
# this option then it automatically enables lt-cred-mech internally
# as if you had enabled both.
#
# Note that you can use only one auth mechanism at the same time! This is because,
# both mechanisms conduct username and password validation in different ways.
#
# Use either lt-cred-mech or use-auth-secret in the conf
# to avoid any confusion.
#
use-auth-secret

# 'Static' authentication secret value (a string) for TURN REST API only.
# If not set, then the turn server
# will try to use the 'dynamic' value in the turn_secret table
# in the user database (if present). The database-stored  value can be changed on-the-fly
# by a separate program, so this is why that mode is considered 'dynamic'.
#
static-auth-secret=Some-Really-Long-Str0ng-password-Secret

# Server name used for
# the oAuth authentication purposes.
```

```
# The default value is the realm name.
#
server-name=turn.my-awesome-domain.com

# Option to redirect all log output into system log (syslog).
#
syslog

# Disable RFC5780 (NAT behavior discovery).
#
# Originally, if there are more than one listener address from the same
# address family, then by default the NAT behavior discovery feature enabled.
# This option disables the original behavior, because the NAT behavior
# discovery adds extra attributes to response, and this increase the
# possibility of an amplification attack.
#
# Strongly encouraged to use this option to decrease gain factor in STUN
# binding responses.
#
no-rfc5780

# Disable handling old STUN Binding requests and disable MAPPED-ADDRESS
# attribute in binding response (use only the XOR-MAPPED-ADDRESS).
#
# Strongly encouraged to use this option to decrease gain factor in STUN
# binding responses.
#
no-stun-backward-compatibility

# Only send RESPONSE-ORIGIN attribute in binding response if RFC5780 is enabled.
#
# Strongly encouraged to use this option to decrease gain factor in STUN
# binding responses.
#
response-origin-only-with-rfc5780
```

Once you've set your turnserver.conf file, you'll want to save it with CTRL + O, then pres Enter to confirm, then use CTRL + X to exit the nano editor.

Now we will restart our turn server with the command

```
sudo systemctl restart coturn
```

and then check the status of the service with

```
sudo systemctl status coturn
```

You should see a message saying that the service is ‘active’, and showing no errors.

Lastly, but certainly not least, we'll point our domain name to our server. For this it's difficult to give specific instruction, as every domain registrar is different. You need, however, to access the DNS tools for your domain registrar, and create an A record.

In the A record, for the sub-domain put “turn”, for the IP address, put your VPS public IPv4 address. Set the DNS record to update in 10 minutes time or 600 seconds (if your registrar provides this option). Then save.

Within 10 minutes, you should be able to ping your sub-domain, and see the public IP address of your VPS server coming back in the terminal.

Using the Turn Server

Now, we'll want to put our turn server to use. You can use turn for dozens of applications that are made to include audio / video chat. XMPP servers will almost always have a turn option in the configuration somewhere. NextCloud Talk uses Turn with a shared secret as well. Matrix (in the dendrite.yml configuration file) has a section for setting up your turn information and using a shared secret of username / password combination. In the video I show how to setup the Matrix / Dendrite server to use Turn, but once you've done it, you'll see that it's just a matter of entering your turn server URL and authentication details on pretty much any software where you want to have WebRTC based audio / video communication.

Support My Channel and Content

Support my Channel and ongoing efforts through Patreon:
https://www.patreon.com/awesomeopensource