Configuration

Let’s configure jitsi-meet. The first step is to set up jitsi-meet and allow starting conference on your server like the website meet.ji.si

If you use my package, all configurations files are secure to the user that runs the service. So to edit them easily I suggest using the user "root".

Organization of the files

The package’s files are organized like this:

- jicofo-bin
  - /etc/jicofo (configurations)
  - /usr/lib/jicofo (binaries)
- jitsi-videobridge-bin
  - /etc/jitsi-videobridge (configurations)
  - /usr/lib/jitsi-videobridge (binaries)
- jitsi-meet-bin
  - /etc/webapps/jitsi-meet (configurations)
  - /usr/share/webapps/jitsi-meet (html files)
  - /usr/share/doc/jitsi-meet (examples of configuration for nginx or apache)
- prosody
  - /etc/prosody (general configurations)
  - /etc/prosody/conf.d (configurations)
  - /usr/lib/jitsi-meet-prosody (lua plugins)
  - /usr/share/doc/jitsi-meet-prosody (example of configuration)
- coturn
  - /etc/turnserver (configurations)
  - /usr/share/doc/jitsi-meet-turnserver (example of configuration)
- nginx
  - /etc/nginx

Loopback

Let’s set up a local loopback for your jitsi-meet domain to allow each project to communicate locally together.

/etc/hosts:

```
127.0.0.1 YOUR_DOMAIN auth.YOUR_DOMAIN
::1 YOUR_DOMAIN auth.YOUR_DOMAIN
```

Jicofo

/etc/jicofo/jicofo.conf

```
jicofo {
  xmpp: {
    client: {
      client-proxy: "focus.YOUR_DOMAIN"
      xmpp-domain: "YOUR_DOMAIN"
      domain: "auth.YOUR_DOMAIN"
      username: "focus"
      password: "FOCUS_PASSWORD"
      conference-muc-jid = conference.YOUR_DOMAIN
    }
    trusted-domains: [ "recorder.YOUR_DOMAIN" ]
  }
  bridge: {
    brewery-jid: "JvbBrewery@internal.auth.YOUR_DOMAIN"
  }
}
```

Jitsi videobridge

/etc/jitsi-videobridge/sip-communicator.properties

```
org.jitsi.videobridge.xmpp.user.shard.DOMAIN=auth.YOUR_DOMAIN
org.jitsi.videobridge.xmpp.user.shard.PASSWORD=__PASSWORD_FOR_USER_jvb__
org.jitsi.videobridge.xmpp.user.shard.MUC_JIDS=JvbBrewery@internal.auth.YOUR_DOMAIN
#use uuidgen
org.jitsi.videobridge.xmpp.user.shard.MUC_NICKNAME=1a988801-1476-4417-9bcc-03f648be86c7
```

Jitsi meet

/etc/webapps/jitsi-meet/config.js

```
var config = {
  hosts: {
    domain: 'YOUR_DOMAIN',
    // ...
    muc: 'conference.YOUR_DOMAIN'
  },
  bosh: '//YOUR_DOMAIN/http-bind',
  // ...
}
```

Prosody

"jitsi-meet-prosody" provide an example of configuration at "/usr/share/doc/jitsi-meet-prosody/prosody.cfg.lua-jvb.example".

Let’s copy the example to the prosody directory:

```
cd /etc/prosody
mkdir conf.d
cp /usr/share/doc/jitsi-meet-prosody/prosody.cfg.lua-jvb.example conf.d/jitsi.cfg.lua
```

Include this file to the main prosody configuration:

/etc/prosody/prosody.cfg.lua

```
-- at the end of the file
Include "conf.d/*.cfg.lua"
```

/etc/prosody/conf.d/jitsi.cfg.lua

Replace all:

- "jitmeet.example.com" with "YOUR_DOMAIN"
- "focusUser@auth.YOUR_DOMAIN" with "focus@auth.YOUR_DOMAIN"

Then adjust the configuration at your needs.

```
plugin_paths = { "/usr/lib/jitsi-meet-prosody/" }
```

SSL

We need to generate SSL for prosody and nginx for YOUR_DOMAIN and auth.YOUR_DOMAIN.

Self-signed

Generate your certificate

```
sudo -u prosody prosodyctl cert generate YOUR_DOMAIN
sudo -u prosody prosodyctl cert generate auth.YOUR_DOMAIN
mv /var/lib/prosody/*.{crt,cnf,key} /etc/prosody/certs/
trust anchor /etc/prosody/certs/YOUR_DOMAIN.crt
trust anchor /etc/prosody/certs/auth.YOUR_DOMAIN.crt
update-ca-trust # for java
```

Letsencrypt

If you use letsencrypt, you can import your certificate automatically to prosody.
Prosody warns you about "No certificate for ...". Don’t worry, these virtual hosts are internal. The only ones you needs are YOUR_DOMAIN and auth.YOUR_DOMAIN.

/etc/letsencrypt/renewal-hooks/deploy/prosody.sh

```
#!/bin/sh
/usr/bin/prosodyctl --root cert import /etc/letsencrypt/live
```

The set execution flag to the file and run it once

```
chmod +x /etc/letsencrypt/renewal-hooks/deploy/prosody.sh
/etc/letsencrypt/renewal-hooks/deploy/prosody.sh
```

Each time letsencrypt renew his certificates, it will automatically import the certificate to prosody.

Prosody

The prosody example should point to the directory "/etc/prosody/certs". Readjust the path if needed. Also, add the certificate for the "auth" virtual host.

/etc/prosody/conf.d/jitsi.cfg.lua

```
VirtualHost "YOUR_DOMAIN"
ssl = {
  key = "/etc/prosody/certs/YOUR_DOMAIN.key";
  certificate = "/etc/prosody/certs/YOUR_DOMAIN.crt";
```

```
}


VirtualHost "auth.YOUR_DOMAIN"
ssl = {
  key = "/etc/prosody/certs/auth.YOUR_DOMAIN.key";
  certificate = "/etc/prosody/certs/auth.YOUR_DOMAIN.crt";
}
authentication = "internal_hashed"
```

Nginx

Nginx is a proxy that allows us to deliver jitsi-meet webpages to the users' web browser.

Let's copy the provided example.

```
cd /etc/nginx
mkdir sites
cp /usr/share/doc/jitsi-meet/jitsi-meet.example sites/jitsi.conf
```

/etc/nginx/nginx.conf

```
http {
  // ...
  // this should be placed near to the close bracket of the http block
  include sites/*.conf;
}
```

/etc/nginx/sites/jitsi.conf

```
server {
  # ...
  server_name YOUR_DOMAIN;



  # ...
  # use prosody path directly
  ssl_certificate /etc/prosody/certs/YOUR_DOMAIN.crt;
  ssl_certificate_key /etc/prosody/certs/YOUR_DOMAIN.key;
  # or use letencrypt path
```

```
  ssl_certificate /etc/letsencrypt/live/YOUR_DOMAIN/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/YOUR_DOMAIN/privkey.pem;



  # set the config path
  # replace alias /etc/jitsi/meet/jitmeet.example.com-config.js by
  location = /config.js {
    alias /etc/webapps/jitsi-meet/config.js;
  }
  # ...
  location ~ ^/([^/?&:'"]+)/config.js$
  {
    set $subdomain "$1.";
    set $subdir "$1/";
    alias /etc/webapps/jitsi-meet/config.js
  }
}
```

Check your configuration.

```
nginx -t
```

Revision #3
Created 9 May 2023 14:01:53 by Celogeek
Updated 22 May 2023 07:40:09 by Celogeek