Deploy Jitsi with Docker + High Quality

Every `eolab.de` has to be changed to your URL.

Prerequisits

- Ubuntu Server 20.04
- Docker and Docker-Compose are installed
- Running NGINX Reverse Proxy
- A SSL certificate for the desired url

Downloading Jitsi

1. `git clone https://github.com/jitsi/docker-jitsi-meet.git`
2. `mv docker-jitsi-meet/ jitsi-meet/`
3. `cd jitsi-meet/`
4. `cp env.example .env`
5. `./gen-passwords.sh`
6. `mkdir -p ~/.jitsi-meet-cfg/{web/letsencrypt,transcripts,prosody,jicofo,jvb,jigasi,jibri}`

Configuring Jitsi

1. `nano .env`
2. make sure it uses an unused port

   `HTTP_PORT=8000`
3. set timezone

   `TZ=Europe/Berlin`
4. set the later used public URL

   `PUBLIC_URL=https://meet.eolab.de`

Configuring NGINX

1. `sudo nano /etc/nginx/sites-available/eolab.de`
2. make sure there is a redirect from HTTP to https (port 80 to 443)
3. add this to the top

```
upstream jitsi {
  server localhost:8000;
}
```

   change the port if needed

4. add this to the bottom

```
server {
  listen 443 ssl http2;
  listen [::]:443 ssl http2;
  server_name meet.eolab.de;

  ssl_certificate /etc/letsencrypt/live/meet.eolab.de/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/meet.eolab.de/privkey.pem;
  include snippets/ssl-params.conf;

  add_header Strict-Transport-Security "max-age=31536000;
includeSubdomains; preload";
  add_header X-Xss-Protection "1; mode=block";
  add_header X-Content-Type-Options nosniff;
  add_header Referrer-Policy same-origin;
  proxy_cookie_path / "/; HTTPOnly; Secure";
  add_header Expect-CT "enforce, max-age=21600";
  add_header Feature-Policy "payment none";

  keepalive_timeout    70;
  sendfile             on;
  client_max_body_size 0;

  gzip on;
  gzip_disable "msie6";
  gzip_vary on;
  gzip_proxied any;
  gzip_comp_level 6;
  gzip_buffers 16 8k;
  gzip_http_version 1.1;
  gzip_types text/plain text/css application/json
application/javascript text/xml application/xml application/xml+rss
text/javascri$

   location / {
        log_not_found off;
```

```
        proxy_cache_valid 200 120m;
        proxy_set_header        Host    $http_host;
        proxy_set_header        X-Real-IP $remote_addr;
        proxy_set_header        X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header        X-Scheme $scheme;
        proxy_pass http://jitsi/;
        }

   location ~ ^/colibri-ws/([a-zA-Z0-9-\.]+)/(.*) {
        tcp_nodelay on;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_pass http://jitsi/colibri-ws/$1/$2$is_args$args;
        }

   location /xmpp-websocket {
        tcp_nodelay on;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
        proxy_pass http://jitsi/xmpp-websocket;
        }
}
```

make sure to change the `server_name`, `ssl_certificate` and `ssl_certificate_key`

5. Save and quit the file
6. `sudo systemctl restart nginx`

Start Jitsi

1. `docker-compose up -d`

Configure Jitsi with internal authentication for creating rooms

1. make sure Jitsi was run at least once
2. stop it if its running

```
docker-compose stop
```

3. `rm -r ~/.jitsi-meet-cfg`

4. `mkdir -p ~/.jitsi-meet-cfg/{web/letsencrypt,transcripts,prosody,jicofo,jvb,jigasi,jibri}`
5. `nano .env`
6. `ENABLE_AUTH=1`
7. `ENABLE_GUESTS=1`
8. `AUTH_TYPE=internal`
9. Save and quit the file
10. `docker-compose up -d`
11. find the container id of the jitsi_prosody container and copy that

    `docker ps -a`
12. `docker exec -it CONTAINER-ID-HERE /bin/bash`
13. `su`
14. change the `username` and `password` in this command:

    `prosodyctl --config /config/prosody.cfg.lua register username meet.jitsi password`

    These credentials are now needed to create a new room
15. `exit`
16. `exit`

Configure Jitsi for high quality webcams and screenshares

1. make sure Jitsi was run at least once
2. stop it if its running

   `docker-compose stop`
3. `sudo nano ~/.jitsi-meet-cfg/web/config.js`
4. only the moderator should be starting with audio

```
startAudioMuted: 1,
```

5.
```
resolution: 1080,
```

6.
```
constraints: {
       video: {
           height: {
               ideal: 1080,
               max: 1440,
               min: 480
           }
       }
     },
```

7. only the moderator should be starting with video

```
startVideoMuted: 1,
```

8.
```
desktopSharingFrameRate: {
       min: 5,
       max: 24
     },
```

9.
```
videoQuality: {
        // Provides a way to prevent a video codec from being
negotiated on the JVB connection. The codec specified
        // here will be removed from the list of codecs present in the
SDP answer generated by the client. If the
        // same codec is specified for both the disabled and preferred
option, the disable settings will prevail.
        // Note that 'VP8' cannot be disabled since it's a mandatory
codec, the setting will be ignored in this case.
    //    disabledCodec: 'H264',

        // Provides a way to set a preferred video codec for the JVB
connection. If 'H264' is specified here,
        // simulcast will be automatically disabled since JVB doesn't
support H264 simulcast yet. This will only
        // rearrange the the preference order of the codecs in the SDP
answer generated by the browser only if the
        // preferred codec specified here is present. Please ensure
that the JVB offers the specified codec for this
        // to take effect.
    //    preferredCodec: 'VP8',

        // Provides a way to configure the maximum bitrates that will
be enforced on the simulcast streams for
        // video tracks. The keys in the object represent the type of
```

```
the stream (LD, SD or HD) and the values
        // are the max.bitrates to be set on that particular type of
stream. The actual send may vary based on
        // the available bandwidth calculated by the browser, but it
will be capped by the values specified here.
        // This is currently not implemented on app based clients on
mobile.
        maxBitratesVideo: {
            low: 1500000,
            standard: 5000000,
            high: 10000000
        },

        // The options can be used to override default thresholds of
video thumbnail heights corresponding to
        // the video quality levels used in the application. At the
time of this writing the allowed levels are:
        //     'low' - for the low quality level (180p at the time of
this writing)
        //     'standard' - for the medium quality level (360p)
        //     'high' - for the high quality level (720p)
        // The keys should be positive numbers which represent the
minimal thumbnail height for the quality level.
        //
        // With the default config value below the application will use
'low' quality until the thumbnails are
        // at least 360 pixels tall. If the thumbnail height reaches
720 pixels then the application will switch to
        // the high quality.
    //    minHeightForQualityLvl: {
    //        360: 'standard',
    //        720: 'high'
    //    },

        // Provides a way to resize the desktop track to 720p (if it is
greater than 720p) before creating a canvas
        // for the presenter mode (camera picture-in-picture mode with
screenshare).
    //    resizeDesktopForPresenter: false
     },
```

this will set the low bitrate to 1,5Mbit, the standard to 5Mbit and the high to 10Mbit

10. Save and quit the file
11. ```
    docker-compose up -d
    ```
    ```

From:
https://wiki.eolab.de/ - **HSRW EOLab Wiki**

Permanent link:
https://wiki.eolab.de/doku.php?id=user:jan001:jitsi_docker&rev=1617290251

Last update: **2021/08/24 17:34**