

MaryTTS with HTTPS

This is a manual how to run [RT-Voice](#) with MaryTTS over HTTPS.

MaryTTS supports more than 5 different languages with over 30 voices.

For more, see <http://mary.dfki.de/>

crosstales LLC creates assets to help you realize great solutions in **Unity**:

[Visit us at the Unity AssetStore](#)

Note:

In this Tutorial we used

- Marytts 5.2
- Ubuntu 16.04.1

We use **Digitalocean** as our host service. If you like to get **\$10 free credit**, please use our referral link:

<https://m.do.co/c/2a7de537032e>

Important!

Recommended minimum RAM for the server is **2GB**.

MaryTTS

Start with this command to get the latest list of Linux-packages:

```
USER@UBUNTU: sudo apt-get update
```

Then follow this steps to install MaryTTS on your Linux server.

<https://github.com/marytts/marytts/wiki/Local-MaryTTS-Server-Installation>

HINT: If you get any error while installing **openjdk-7-jdk maven** use this Command instead.

```
USER@UBUNTU: sudo apt-get install -y openjdk-8-jdk maven
```

HINT: Start the MaryTTS server without **.sh**.

```
USER@UBUNTU: sudo -u mary /local/mary/marytts/target/marytts-5.2/bin/marytts-server
```

NGINX and Let's Encrypt

After you installed MaryTTS on your server, let's start with NGINX.

Installing NGINX

Please Install NGINX first, if you already installed NGINX skip this part.

```
USER@UBUNTU: sudo apt-get install nginx
```

Server block Configuration

First we create the configuration.

```
USER@UBUNTU: sudo vim /etc/nginx/sites-available/yourdomain
```

Write this in the configuration:

```
server {
    listen 80;
    server_name yourdomain.com;

    location ~ /\.well-known {
        allow all;
    }

    location / {
        # Simple requests
        if ($request_method ~* "(GET|POST)") {
            add_header "Access-Control-Allow-Origin" *;
        }

        # Preflighted requests
        if ($request_method = OPTIONS ) {
            add_header "Access-Control-Allow-Origin" *;
            add_header "Access-Control-Allow-Methods" "GET, POST, OPTIONS, HEAD";
            add_header "Access-Control-Allow-Headers" "Authorization, Origin, X-Requested-With, Content-Type, Accept";
            return 200;
        }

        proxy_pass      http://127.0.0.1:59125;
        proxy_redirect  off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Host $server_name;
    }
}
```

Enable Configuration

```
USER@UBUNTU: sudo ln -s /etc/nginx/sites-available/yourdomain /etc/nginx/sites-enabled/yourdomain
```

Test Configuration

```
USER@UBUNTU: nginx -t //or nginx -t -c /etc/nginx/nginx.conf
```

If you don't have any syntax errors, restart the Server.

```
USER@UBUNTU: sudo service nginx restart
```

Let's Encrypt

Installation

1. `USER@UBUNTU: cd /usr/local/sbin`
2. `USER@UBUNTU: sudo wget https://dl.eff.org/certbot-auto`

Enable the execution of the script.

```
USER@UBUNTU: sudo chmod a+x /usr/local/sbin/certbot-auto
```

Creating the Certification

```
USER@UBUNTU: certbot-auto certonly -a webroot --webroot-path=/usr/share/nginx/html -d  
yourdomain.com
```

If everything worked, you should see this:

IMPORTANT NOTES:

- If you lose your account credentials, you can recover through e-mails sent to sammy@digitalocean.com
- Congratulations! Your certificate and chain have been saved at `/etc/letsencrypt/live/example.com/fullchain.pem`. Your cert will expire on 2016-03-15. To obtain a new version of the certificate in the future, simply run Let's Encrypt again.
- Your account credentials have been saved in your Let's Encrypt configuration directory at `/etc/letsencrypt`. You should make a secure backup of this folder now. This configuration directory will also contain certificates and private keys obtained by Let's Encrypt so making regular backups of this folder is ideal.
- If like Let's Encrypt, please consider supporting our work by:

Donating to ISRG / Let's Encrypt: <https://letsencrypt.org/donate>

Donating to EFF: <https://eff.org/donate-le>

With this command you see if your certificate was generated correctly:

```
USER@UBUNTU: sudo ls -l /etc/letsencrypt/live/yourdomain.com
```

Strong Diffie-Hellman Group

```
USER@UBUNTU: sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048
```

[More about Diffie-Hellman Group.](#)

Configuration from TLS/SSL (Nginx)

Server block Configuration

Add this to the Server Block.

IMPORTANT! Change the **Listen** from **80** to **443**!

```
server {
    listen 443 http2 ssl;
    listen [::]:443 http2 ssl;
    server_name yourdomain.com;

    ssl_certificate /etc/letsencrypt/live/yourdomain.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/yourdomain.com/privkey.pem;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;
    ssl_dhparam /etc/ssl/certs/dhparam.pem;
    ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-
SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-
ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-
AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-
AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-
SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:AES128-
GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-
SHA:AES:CAMELLIA:DES-CBC3-
SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-SHA:!EDH-
RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA';
    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:50m;
    ssl_stapling on;
    ssl_stapling_verify on;
    add_header Strict-Transport-Security max-age=15768000;

    location ~ /\.well-known {
        allow all;
    }

    location / {
        # Simple requests
        if ($request_method ~* "(GET|POST)") {
            add_header "Access-Control-Allow-Origin" *;
        }

        # Preflighted requests
        if ($request_method = OPTIONS ) {
            add_header "Access-Control-Allow-Origin" *;
            add_header "Access-Control-Allow-Methods" "GET, POST, OPTIONS, HEAD";
            add_header "Access-Control-Allow-Headers" "Authorization, Origin, X-Requested-With,
Content-Type, Accept";
            return 200;
        }

        proxy_pass      http://127.0.0.1:59125;
        proxy_redirect  off;
        proxy_set_header    Host $host;
        proxy_set_header    X-Real-IP $remote_addr;
```

```
    proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header    X-Forwarded-Host $server_name;
}
}
```

Test Configuration

```
USER@UBUNTU: nginx -t //or nginx -t -c /etc/nginx/nginx.conf
```

If you don't have any syntax errors, restart the server.

```
USER@UBUNTU: sudo service nginx restart
```

Automatic renew (Certification)

```
USER@UBUNTU: certbot-auto renew
```

Output:

```
Checking for new version...
Requesting root privileges to run letsencrypt...
/home/sammy/.local/share/letsencrypt/bin/letsencrypt renew
Processing /etc/letsencrypt/renewal/example.com.conf

The following certs are not due for renewal yet:
/etc/letsencrypt/live/example.com/fullchain.pem (skipped)
No renewals were attempted.
```

Adding Cron jobs

To add a [Cron job](#), you have to create a configuration file.

```
USER@UBUNTU: sudo crontab -e
```

Add this lines:

```
30 2 * * 1 /usr/local/sbin/certbot-auto renew >> /var/log/le-renew.log
35 2 * * 1 /etc/init.d/nginx reload
```

More about certbot:

<https://certbot.eff.org/all-instructions/>

Useful Links

HTTP-Auth

<https://www.nginx.com/resources/admin-guide/restricting-access-auth-basic/>

Installing NGINX

<https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-16-04>

X11-Forwarding Over SSH

http://www.geo.mtu.edu/geoschem/docs/putty_install.html

Firewall with UFW

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-16-04>

Linux Hardening

<https://www.tecmint.com/linux-server-hardening-security-tips/>