

# Setup guide: Emby setup

## Emby module **WISECP**

[Order now](#) | [Download](#) | [FAQ](#)

### 1. Install Emby on your server

To simplify deployment and help automate this for as many users as possible, we provide a BASH script to handle repo installation as well as installing Emby. All you need to do is run this command on your system

```
wget https://github.com/MediaBrowser/Emby.Releases/releases/download/4.7.14.0/emby-server-deb_4.7.14.0_amd64.deb
```

```
dpkg -i emby-server-deb_4.7.14.0_amd64.deb
```

### 2. SSL certificate generation:

```
sudo apt install nginx certbot python3-certbot-nginx
```

```
sudo certbot --nginx --agree-tos --redirect --hsts --staple-ocsp --email YOUR_EMAIL -d  
DOMAIN_NAME
```

```
echo "0 0 * * * root certbot renew --quiet --no-self-upgrade --post-hook 'systemctl reload  
nginx'" | sudo tee -a /etc/cron.d/renew_certbot
```

### 3. Configure nginx from a subdomain

Create a file named nginx.conf.

```
cd /etc/nginx/conf.d/  
nano nginx.conf
```

Then, insert the following text, replacing **DOMAIN\_NAME** with your domain.

```
# Uncomment the commented sections after you have acquired a SSL Certificate  
server {  
    listen 80;  
    #listen [::]:80;  
    server_name DOMAIN_NAME;  
  
    # Uncomment to redirect HTTP to HTTPS  
    return 302 https://$host$request_uri;  
}  
  
server {  
    listen 443 ssl http2;  
    #listen [::]:443 ssl http2;  
    server_name DOMAIN_NAME;  
  
    ## The default `client_max_body_size` is 1M, this might not be enough for some  
posters, etc.  
    client_max_body_size 20M;  
  
    # use a variable to store the upstream proxy  
    # in this example we are using a hostname which is resolved via DNS  
    # (if you aren't using DNS remove the resolver line and change the variable to point  
to an IP address e.g `set $emby 127.0.0.1`)  
    set $emby 127.0.0.1;  
    resolver 127.0.0.1 valid=30;  
  
    ssl_certificate /etc/letsencrypt/live/DOMAIN_NAME/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/DOMAIN_NAME/privkey.pem;  
    include /etc/letsencrypt/options-ssl-nginx.conf;  
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;  
    add_header Strict-Transport-Security "max-age=31536000" always;  
    ssl_trusted_certificate /etc/letsencrypt/live/DOMAIN_NAME/chain.pem;  
    ssl_stapling on;  
    ssl_stapling_verify on;
```

```

# Security / XSS Mitigation Headers
# NOTE: X-Frame-Options may cause issues with the webOS app
add_header X-Frame-Options "SAMEORIGIN";
add_header X-XSS-Protection "0"; # Do NOT enable. This is obsolete/dangerous
add_header X-Content-Type-Options "nosniff";

# COOP/COEP. Disable if you use external plugins/images/assets
add_header Cross-Origin-Opener-Policy "same-origin" always;
add_header Cross-Origin-Embedder-Policy "require-corp" always;
add_header Cross-Origin-Resource-Policy "same-origin" always;

# Permissions policy. May cause issues on some clients
add_header Permissions-Policy "accelerometer=(), ambient-light-sensor=(), battery=(),
bluetooth=(), camera=(), clipboard-read=(), display-capture=(), document-domain=(), encrypted-
media=(), gamepad=(), geolocation=(), gyroscope=(), hid=(), idle-detection=(), interest-
cohort=(), keyboard-map=(), local-fonts=(), magnetometer=(), microphone=(), payment=(),
publickey-credentials-get=(), serial=(), sync-xhr=(), usb=(), xr-spatial-tracking=()"
always;

# Tell browsers to use per-origin process isolation
add_header Origin-Agent-Cluster "?1" always;

# Content Security Policy
# See: https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
# Enforces https content and restricts JS/CSS to origin
# External Javascript (such as cast_sender.js for Chromecast) must be
whitelisted.

# NOTE: The default CSP headers may cause issues with the webOS app
#add_header Content-Security-Policy "default-src https: data: blob:
http://image.tmdb.org; style-src 'self' 'unsafe-inline'; script-src 'self' 'unsafe-inline'
https://www.gstatic.com https://www.youtube.com blob;; worker-src 'self' blob;; connect-src
'self'; object-src 'none'; frame-ancestors 'self'";

location = / {
    return 302 http://$host/web/;
    #return 302 https://$host/web/;
}

location / {

```

```

        # Proxy main Emby traffic
        proxy_pass http://$emby:8096;
        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-Proto $scheme;
        proxy_set_header X-Forwarded-Protocol $scheme;
        proxy_set_header X-Forwarded-Host $http_host;

        # Disable buffering when the nginx proxy gets very resource heavy upon
streaming
        proxy_buffering off;
    }

    # location block for /web - This is purely for aesthetics so /web/#!/ works instead
of having to go to /web/index.html/#!/
    location = /web/ {
        # Proxy main Emby traffic
        proxy_pass http://$emby:8096/web/index.html;
        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-Proto $scheme;
        proxy_set_header X-Forwarded-Protocol $scheme;
        proxy_set_header X-Forwarded-Host $http_host;
    }

    location /socket {
        # Proxy Emby Websockets traffic
        proxy_pass http://$emby:8096;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        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-Proto $scheme;
        proxy_set_header X-Forwarded-Protocol $scheme;
        proxy_set_header X-Forwarded-Host $http_host;
    }

```

```
}
```

## Restarting nginx

```
sudo service nginx restart
```

---

Revision #3

Created 27 November 2023 08:35:09 by Yuliia Noha

Updated 27 November 2023 10:32:47 by Yuliia Noha