# Install Nginx RTMP for live streaming (ubuntu 20.4)

<https://www.servermania.com/kb/articles/nginx-rtmp/>

<https://www.nginx.com/blog/video-streaming-for-remote-learning-with-nginx/#configure>

1. Install the build tools

Sudo apt update

Sudo apt install build-essential git

1. Install dependencies

Sudo apt install **libpcre3-dev libssl-dev zlib1g-dev**

1. Compile nginx with the RTMP module

$**cd /path/to/build/dir**

$**git clone** <https://github.com/arut/nginx-rtmp-module.git>

$**git clone** <https://github.com/nginx/nginx.git>

$**cd nginx**

$**./auto/configure --add-module=../nginx-rtmp-module --with-cc-opt=”-Wimplicit-fallthrough=0”**

$**make**

$**sudo make install**

**Added —with-cc-opt=”-Wimplicit-fallthrough=0” when compiling errors.**

./configure --prefix=/usr/local/nginx --with-http\_ssl\_module --add-module=../nginx-rtmp-module  
make -j 1  
make install

1. Configure Nginx

The way **nginx** and its modules work is determined in the **configuration file**. By default, the **configuration file** is named **nginx**. **conf** and placed in the **directory** /usr/local/**nginx**/**conf** , /etc/**nginx** , or /usr/local/etc/**nginx**.

\*\* Nginx with RTMP is located at /usr/local/nginx/sbin/nginx by default. It loads the config files in /usr/local/nginx/conf.

Test the configuration file

|  |  |
| --- | --- |
| 1 | $ sudo /usr/local/nginx/sbin/nginx -t |

Start nginx in the background

|  |  |
| --- | --- |
| 1 | /usr/local/nginx/sbin/nginx |

Start nginx in the foreground

|  |  |
| --- | --- |
| 1 | /usr/local/nginx/sbin/nginx -g 'daemon off;' |

Reload the config on the go

|  |  |
| --- | --- |
| 1 | /usr/local/nginx/sbin/nginx -t && nginx -s reload |

Kill nginx

|  |  |
| --- | --- |
| 1 | /usr/local/nginx/sbin/nginx -s stop |

## **4. Pushing live stream to nginx using rtmp[¶](https://docs.peer5.com/guides/setting-up-hls-live-streaming-server-using-nginx/" \l "4-pushing-live-stream-to-nginx-using-rtmp" \o "Permanent link)**

nginx accepts rtmp stream as input. For a proper HLS stream the video codec should be [x264](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) and audio codec aac/mp3/ac3 most commonly being aac.

### **Options 1: From existing rtmp stream already in h264**[**¶**](https://docs.peer5.com/guides/setting-up-hls-live-streaming-server-using-nginx/#options-1-from-existing-rtmp-stream-already-in-h264)

if you have an existing rtmp stream in the correct codec, you can skip ffmpeg and tell nginx to pull the stream directly. In order to do so add a pull directive under application section in nginx.conf like so:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | application show {  live on;  pull rtmp://example.com:4567/sports/channel3 live=1;  # to change the local stream name use this syntax: ... live=1 name=ch3;  # other directives...  # hls\_...  } |

FFMEG <https://johnathan.org/attempting-to-stream-a-webcam-to-an-rtmp-server/>

Install ffmpeg: sudo apt install ffmpeg

My /usr/local/nginx/conf/nginx.conf:

#user www-data;

worker\_processes auto;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

}

# RTMP configuration

rtmp {

server {

listen 1935;

chunk\_size 4096;

application live {

live on;

record off;

interleave on;

hls on;

hls\_path /var/www/winn-live/hls/;

hls\_fragment 3; # was 15s

dash on;

dash\_path /var/www/winn-live/dash/;

dash\_fragment 15s;

# disable consuming the sream from nginx as rtmp

deny play all;

}

application live2 {

live on;

record off;

interleave on;

hls on;

hls\_path /var/www/winn-live/hls2/;

hls\_fragment 3; # was 15s

dash on;

dash\_path /var/www/winn-live/dash2/;

dash\_fragment 15s;

# disable consuming the sream from nginx as rtmp

deny play all;

}

application live3 {

live on;

record off;

interleave on;

hls on;

hls\_path /var/www/winn-live/hls3/;

hls\_fragment 3; # was 15s

# disable consuming the sream from nginx as rtmp

deny play all;

}

application live4 {

live on;

record off;

interleave on;

hls on;

hls\_path /var/www/winn-live/hls4/;

hls\_fragment 3;

# disable consuming the sream from nginx as rtmp

deny play all;

}

}

}

http {

sendfile off;

tcp\_nopush on;

directio 512;

default\_type application/octet-stream;

#log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

keepalive\_timeout 65;

#gzip on;

server {

listen 81;

listen [::]:81;

server\_name win-live.com;

#charset koi8-r;

#access\_log logs/host.access.log main;

location /hls {

# Disable cache

add\_header Cache-Control non-cache;

# CORS setup

add\_header 'Access-Control-Allow-Origin' '\*' always;

add\_header 'Access-Control-Expose-Headers' 'Content-Length';

if ($request\_method = 'OPTIONS') {

add\_header 'Access-Control-Allow-Origin' '\*';

add\_header 'Access-Control-Max-Age' 1728000;

add\_header 'Content-Type' 'text/plain charset=UTF-8';

add\_header 'Content-Length' 0;

return 204;

}

types {

application/vnd.apple.mpegurl m3u8;

video/mp2t ts;

application/dash+xml mpd;

text/html html;

}

root /var/www/winn-live;

}

location /hls2 {

# Disable cache

add\_header Cache-Control non-cache;

# CORS setup

add\_header 'Access-Control-Allow-Origin' '\*' always;

add\_header 'Access-Control-Expose-Headers' 'Content-Length';

if ($request\_method = 'OPTIONS') {

add\_header 'Access-Control-Allow-Origin' '\*';

add\_header 'Access-Control-Max-Age' 1728000;

add\_header 'Content-Type' 'text/plain charset=UTF-8';

add\_header 'Content-Length' 0;

return 204;

}

types {

application/vnd.apple.mpegurl m3u8;

video/mp2t ts;

application/dash+xml mpd;

text/html html;

}

root /var/www/winn-live;

}

root /var/www/winn-live;

index index.html;

location / {

try\_files $uri $uri/ =404;

}

#error\_page 404 /404.html;

#

# proxy the PHP scripts to Apache listening on 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

#

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's document root

# concurs with nginx's one

#

#location ~ /\.ht {

# deny all;

#}

}

types {

application/vnd.apple.mpegurl m3u8;

video/mp2t ts;

text/html html;

application/dash+xml mpd;

}

# another virtual host using mix of IP-, name-, and port-based configuration

#

#server {

# listen 8000;

# listen somename:8080;

# server\_name somename alias another.alias;

# location / {

# root html;

# index index.html index.htm;

# }

#}

# HTTPS server

#

#server {

# listen 443 ssl;

# server\_name localhost;

# ssl\_certificate cert.pem;

# ssl\_certificate\_key cert.key;

# ssl\_session\_cache shared:SSL:1m;

# ssl\_session\_timeout 5m;

# ssl\_ciphers HIGH:!aNULL:!MD5;

# ssl\_prefer\_server\_ciphers on;

# location / {

# root html;

# index index.html index.htm;

# }

#}

}