<2020-11-30 Mo> Dr. Arne Babenhauserheide / draketo.de

How to stream into Freenet / Hyphanet

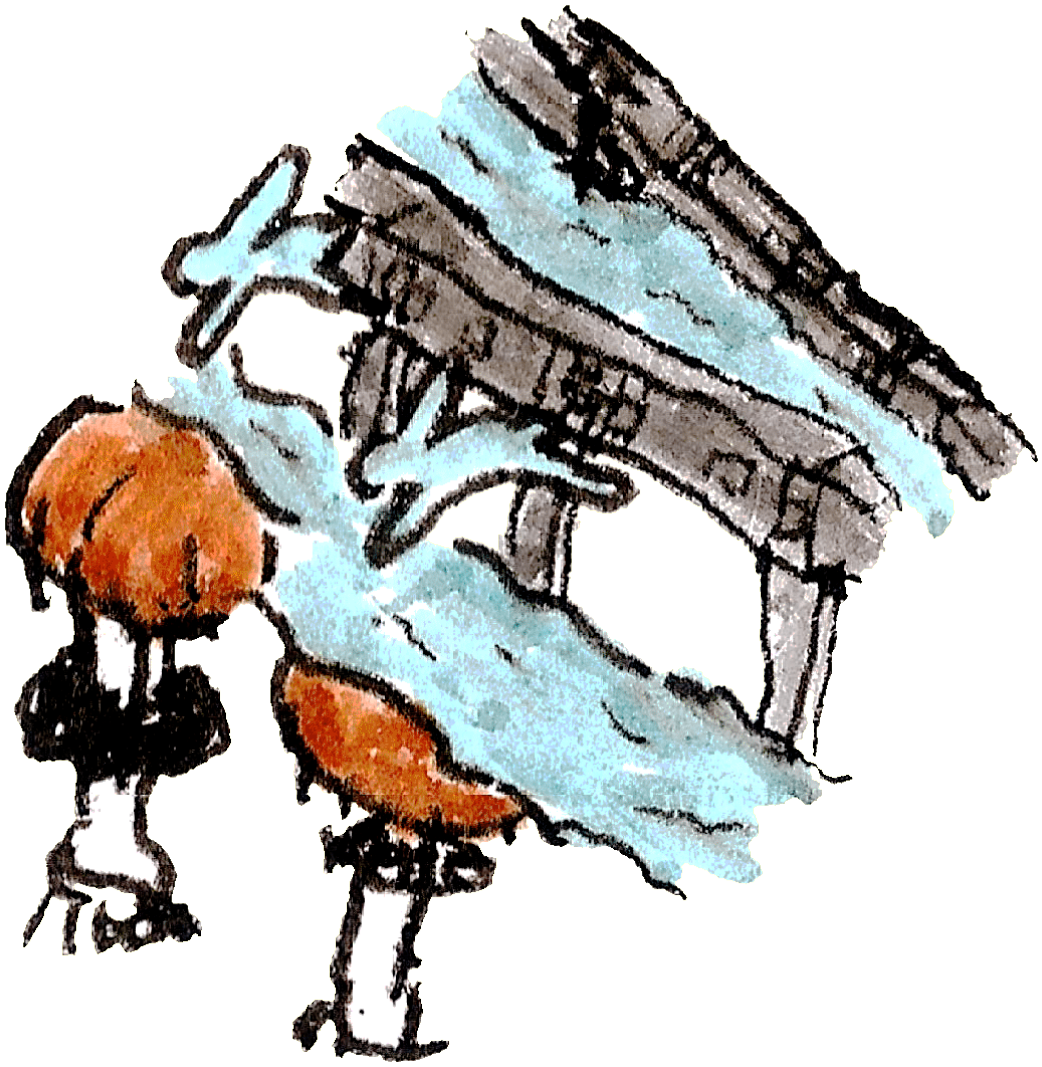
With version 1489 Freenet / Hyphanet gained a filter for m3u files — playlists — which enables its use as streaming platform.

Update 2021-07: I now created a streaming-website generation tool which automates the process of streaming video-on-demand. It's called Guile Media Site. You can clone it from sourcehut or get it directly from Freenet:

- `https://hg.sr.ht/~arnebab/guile-media-site/browse/site/gms.scm`
- `USK@KxGwMvg~cXm5hs1ZX4NSH~I8fYyqcQD-~8dDdtmDs18,gKSJ4JQ4E1s2Pi-C1iKnfcpWp2pTmc`

For streaming audio, use the **multi-node version** below.

In this article I want to show you how you can forward an arbitrary stream into Hyphanet. To listen to a stream by its USK-key, download and run your own Freenet / Hyphanet node.



TLDR: For audio, just use re-stream to Hyphanet:

```
 /tmp
git clone https://github.com/hyphanet/re-stream-into-freenet
 re-stream-into-freenet
 -e   ./re-stream-to-freenet.sh theradiocc
```

Requirements: pyFreenet3 (`pip3 install --user pyFreenet3`) and openjdk.

It directly creates a freesite where people can listen to the audio. Watch for output like this:

```
Creating minimal streaming site:
http://127.0.0.1:8888/freenet:USK@fLh9vDtv4yliMDD6tJSb~GMLdC2EQuf~Wpc1PxyFMXo,30P5T9r
```

To embed such a stream on your Freesite, you can use an audio-tag with the `/SSK@.../stream.m3u` in `src`. You can simply copy a working tag from the minimal site.

Contents

1 Requirements

Watching this stream requires a Freenet / Hyphanet node running at least build 1492. For uploading the stream you need pyFreenet. And GNU Linux, but you guessed that, right?

- Install Freenet / Hyphanet: `install-freenet-linux.org`
  (as of build 1492, Freenet supports at most openjdk 15. Version 16 and 17 still need module system fixes)
- Install pyFreenet: Get Python3, then: `pip3 install --user pyFreenet3`

The stream has around 15 minute delay, if your node is fast enough. Given that a practical streaming setup for semi-professional free streaming uses around 20s delay, that's not too shabby over a confidential decentralized network. With a stronger internet connection, you might be able to push this further down. I'm limited to 40Mbps in theory, the nodes for this use much less.

As source I use theradio.cc, a radio station that streams creativecommons licensed media.

To get gap-less playback for mp3, almost without artefacts, use `mpv --ytdl=no --prefetch-playlist`.

For ogg, use a site in Freenet with an audio-tag, because there's some issue with our file splitting that trips up mpv playlist playing with ogg.

2 Restreaming theradio.cc with a fully automated script

To use this script, either download it, inspect it, then run it as a bash script, or, if you're reckless, run it directly:

```
sudo apt install openjdk-17-jdk wget
pip3 install --user pyFreenet3
wget -O stream-radiocc-into-freenet.sh
  https://www.draketo.de/software/stream-the-radio-cc-into-freenet.txt
bash stream-radiocc-into-freenet.sh
```

```
http://stream.theradio.cc:8000/trcc-stream-lq.ogg
```

```
mktemp -d XXXXXXXX
mkdir
```

```
wget http://ftp.lysator.liu.se/pub/freenet/fred-releases/build01494/new_installer_off
  -O freenet-installer.jar
mkdir node1
 node1

 follow the prompts

java -jar ../freenet-installer.jar -console
./run.sh stop

cat > freenet.ini
```

```
./run.sh restart
 giving Freenet some  to start
sleep
 stop the node to replicate it
./run.sh stop


 i  ..5  cp -r node1 node



 i  ..5
    sed -i s/fproxy.port.*/fproxy.port/  node/freenet.ini
    sed -i s/fcp.port.*/fcp.port/  node/freenet.ini
    sed -i s/node.listenPort.*// node/freenet.ini
    sed -i s/node.opennet.listenPort.*// node/freenet.ini
    sed -i s/node.inputBandwidthLimit.*/node.inputBandwidthLimit70k/ node/freenet.ini
    sed -i s/node.outputBandwidthLimit.*/node.outputBandwidthLimit70k/ node/freenet.i


 i  ..5   node ./run.sh start  -

grep -i .port node*/*ini
```

```
 creating the first  segments,  minutes, before starting to upload.
 Please give them enough time.
rm -rf /FREESTREAM-split/
mkdir /FREESTREAM-split/
 /FREESTREAM-split/

timeout  nohup ffmpeg -i
        -c copy -map
        -segment_time :00:30
        -f segment
        -reset_timestamps
 First  segments . Later segments are created  the background.
 This setup can live-stream radio  more than  days.




timeout  nohup ffmpeg -i
        -c copy -map
        -segment_time :06:00
        -f segment
        -reset_timestamps

fcpgenkey -P   tail -n
fcpinvertkey -P
 i  ..9
      >> stream.m3u
```

```
i  ..999
     >> stream.m3u



fcpupload -P  -p  stream.m3u stream.m3u

 Streaming radio to fcpinvertkey -P  stream.m3u



 To create a streaming site, open the Sharesite plugin site
 http://localhost:8180/Sharesite/
  your browser, click
 open your freesite and enter the following:
 fcpinvertkey -P  stream.m3u
 Then change the Insert Hour to -1 to insert at once.
 Save it, go back to the freesite menu,  tick its checkbox and click insert.
 Once a link appears next to your site, your streaming page is ready. You can use tha

 mpv --ytdlno --prefetch-playlist http://127.0.0.1:8888/freenet:fcpinvertkey -P  stre
 i  -- ..99




     ! -e   sleep
   date


   fcpupload -P  -p
     ! -e   sleep
   fcpupload -P  -p
     ! -e   sleep
   fcpupload -P  -p
     ! -e   sleep
   fcpupload -P  -p
     ! -e   sleep
   fcpupload -P  -p
```

```
      ! -e   sleep
    fcpupload -P
      ! -e   sleep
    fcpupload -P
      ! -e   sleep
    fcpupload -P
      ! -e   sleep



    fcpupload -w -P


    sleep
    fcpupload -P

sleep
 All the files are uploading. Stopping the streaming nodes.
 i  /node*    ./run.sh stop  -
```

3 Background: Stream audio with a single node

To stream live, you need a sufficiently fast connection to upload 350KiB **to Freenet** in 30s.

On my node this is not real-time yet: It takes around 11 minutes to insert 3 minutes of music.

The following code uses some optimizations: The uploaded segments fit into a single SSK splitfile (around 300KiB) and multiple files are inserted in parallel to reduce the impact of wait times for detecting successful uploads (the first with higher priority to reduce the time until the stream can start). It still has room for improvement, though. It could, for example, use multiple insert-nodes, one per file in a batch, and reducing the batch-size to 5. That might allow pushing the delay down to 2 minutes.

```
.0.0.1


fcpgenkey -H  -P   tail -n
fcpinvertkey -H  -P

rm -rf /tmp/FREESTREAM-split/
mkdir /tmp/FREESTREAM-split/
```

```
 /tmp/FREESTREAM-split/
 i  ..999
      >> stream.m3u



fcpupload -H  -P  -p  stream.m3u stream.m3u

 Streaming radio to fcpinvertkey -H  -P  stream.m3u


timeout  nohup ffmpeg -i http://mp3.theradio.cc/
        -c copy -map
        -segment_time :00:20
        -f segment
        -reset_timestamps

 you can listen to the stream with
 mpv --ytdlno --prefetch-playlist http://:/freenet:fcpinvertkey -H  -P  stream.m3u
 i  ..99




      ! -e   sleep

    sleep
    date


    fcpupload -H  -P  -p

     j  ..8

        fcpupload -H  -P



    fcpupload -w -H  -P
```

If you have Freenet 1489-pre1 running at port 8888, you can listen in on a test-stream with mpv:

```
mpv --ytdlno --prefetch-playlist http://127.0.0.1:8888/freenet:SSK@d8lB5dqTaAt~39aFk6
```

(also works in vlc, but with small gaps between the segments)

We have Freenet Radio!

It's imperfect and you'll hit some gotchas, but it works, and that's seriously cool!

4 Background: Multi-node version

To speed this up more, you can install multiple nodes. This recipe can livestream audio over Freenet with 5-10 minutes delay if you have at least 400kiB/s upload speed.

First install a template node:

```
/tmp/freenet-streaming-nodes
mkdir

mkdir node1
 node1

wget
java -jar new_installer_offline_1492.jar -console
```

follow the prompts, then open the Freenet web interface:

```
lynx .0.0.1:8888
```

Now **stop the node**:

Next just copy the folder:

```
 i  ..5  cp -r node1 node
```

Then remove the listen-port lines in `freenet.ini`. This causes Freenet to create unique listen ports on the next run.

Also adjust the fcpport, fred port, and optionally the bandwidth:

```
 i  ..5
    sed -i s/fproxy.port.*/fproxy.port/  node/freenet.ini
    sed -i s/fcp.port.*/fcp.port/  node/freenet.ini
    sed -i s/node.listenPort.*// node/freenet.ini
    sed -i s/node.opennet.listenPort.*// node/freenet.ini
    sed -i s/node.inputBandwidthLimit.*/node.inputBandwidthLimit70k/ node/freenet.ini
    sed -i s/node.outputBandwidthLimit.*/node.outputBandwidthLimit70k/ node/freenet.i
```

And start them:

```
 i  ..5   node ./run.sh start  -
```

```
$ grep -i .port node*/*ini
node1/freenet.ini:fproxy.port
node1/freenet.ini:fcp.port
node2/freenet.ini:fproxy.port
node2/freenet.ini:fcp.port
node3/freenet.ini:fproxy.port
node3/freenet.ini:fcp.port
node4/freenet.ini:fproxy.port
node4/freenet.ini:fcp.port
node5/freenet.ini:fproxy.port
node5/freenet.ini:fcp.port
```

Now we have 5 running nodes on FCP ports 9180 to 9580. We'll use them to insert:

```
http://stream.theradio.cc:8000/trcc-stream-lq.ogg
fcpgenkey -P   tail -n
fcpinvertkey -P

rm -rf /tmp/FREESTREAM-split/
mkdir /tmp/FREESTREAM-split/
 /tmp/FREESTREAM-split/
 i  ..999
```

```
      >> stream.m3u



fcpupload -P  -p  stream.m3u stream.m3u

 Streaming radio to fcpinvertkey -P  stream.m3u









timeout  nohup ffmpeg -i
        -c:a libmp3lame -vn  -map :a:0
        -abr  -b:a 50k
        -segment_time :00:50
        -f segment
        -reset_timestamps

 you can listen to the stream with
 mpv --ytdlno --prefetch-playlist http://127.0.0.1:8888/freenet:fcpinvertkey -P  stre
 i  ..99




      ! -e   sleep
    date


    fcpupload -P  -p
      ! -e   sleep
    fcpupload -P  -p
      ! -e   sleep
    fcpupload -P  -p
      ! -e   sleep
```

```
fcpupload -P  -p
  ! -e   sleep
fcpupload -P  -p


  ! -e   sleep
fcpupload -P
  ! -e   sleep
fcpupload -P
  ! -e   sleep
fcpupload -P
  ! -e   sleep



fcpupload -w -P


sleep
fcpupload -P
```

---

5 Video-Streaming

Wouldn't it be great to replace youtube with fully decentralized streaming? Can Freenet do it? Are you up for the test? :-)

You won't be live-streaming yet, bandwidth does not suffice with the currently supported codecs, but you can provide video that starts playing right-away.

We'll prepare the nodes just like before, and choose a video file to stream.

```
/tmp/36C3_-_Climate_Modelling-Q_ysS7C8IBw.mkv
fcpgenkey -P   tail -n
fcpinvertkey -P

rm -rf /tmp/FREESTREAM-split/
mkdir /tmp/FREESTREAM-split/
 /tmp/FREESTREAM-split/
 i  ..999
      >> stream.m3u
```

```
fcpupload -P  -p  stream.m3u stream.m3u

 Streaming radio to fcpinvertkey -P  stream.m3u


timeout  nohup ffmpeg -i
        -c:v libtheora -c:a libvorbis -map
        -q:v  -q:a
        -vf :480
        -segment_time :00:15
        -f segment
        -reset_timestamps

 you can listen to the stream with
 mpv --ytdlno --prefetch-playlist http://127.0.0.1:8888/freenet:fcpinvertkey -P  stre
 i  ..99




     ! -e   sleep
   date


   fcpupload -P  -p
   fcpupload -P  -p
   fcpupload -P  -p
   fcpupload -P  -p
   fcpupload -P  -p

     ! -e   sleep

   sleep
   fcpupload -P
   fcpupload -P
   fcpupload -P
   fcpupload -P

   fcpupload -w -P
```

This has some glitches at the beginning, but you can already watch it:

```
mpv --ytdlno --prefetch-playlist http://127.0.0.1:8888/freenet:SSK@I3yBAkjMtFmZPQS088
```

And with that we have video-streaming over Freenet!

6 Background: Split a video file by silence

Depending on your player, playback might not be completely gapless. You can lessen this problem by splitting the video file by silence. While this will likely fail catastrophically for music, it should work pretty well for recorded talks.

Here's a one-line example from Vi at Stackoverflow, adapted to prouce scaled down ogg theora for Freenet, splitting at places with at least 0.3 seconds of -40dB signal:

```
ffmpeg -i infile.mkv -filter_complex  -map outa -f s16le -y /dev/null   perl -ne   ba
```