Streaming server

Die Planung

Für ein Projekt in meinem privaten Umfeld gab es seit Längerem einen Stream, der mittels No23Player und Windows-PC auf die Beine gestellt wurde. Von einem Tag auf den anderen gab es plötzlich sporadische Streamabbrüche, was ziemlich nervig ist, wenn sich der Stream dann nicht wieder von allein aufbaut. Weil wir auch nach Monaten keine Ursache finden konnten, habe ich mir den Raspi als Ersatz gedacht uî d ha" e angefangen zu forschen, was für unseren Zw%ck d! s Richtige sein könnte. Nach tagelanger Suche im Internet war eines klar: Es gab keine für uns maßgeschneiderte Lösung. Ich musste aus mehreren Bauprojekten eines machen.

Für mich ist Streaming eine komplizierte Sache. Da gibt es die verschiedensten Protokolle und Pakete, die über TCP und UDP laufen können. Dann gibt es in dem Zusammenhang End-to-End-Verbindungen, wo der Hörer ständig mit dem Sender Quittungen austauscht, wie auch Unicast und Multicast, wobei dort die Daten einfach nur in die Weiten des Internet gepumpt werden, wenn ich es richtig verstanden habe. Manche der Verfahren konnten von ganz vielen Playern wiedergegeben werden und manche nur von wenigen, wie z. B. dem VLC. Es gab also sehr viel zu bedenken, wenn man dem Benutzer nicht zu viel Veränderungen zumuten wollte!

Wir nutzten eine TCP-End-to-End-Verbindung mit MMS-Stream, hinter dem sich das Microsoft-WMA verbirgt, womit dann auch sehr viele Player klarkommen, obwohl es schon recht alt ist. Als Übertragungsrate haben wir auf Grund der geringen Qualitätsanforderungen und der maximalen „Sende-Bandbreite" (Upload) nur 64kBit in Mono. Es war also etwas gefragt, was Ähnliches bietet!

Zusätzlich zu den bisher genannten technischen Themen kommt nun noch ein Berg von möglicher Streaming-Software! Ich habe mich näher mit **Icecast**, **FFMPEG** und **VLC** beschäftigt. Ein anderer Freak hatte Ähnliches getestet und war bei FFMPEG hängengeblieben. Das hatte ich leider nicht wirklich gut zum Laufen bekommen. VLC wäre schön gewesen, weil ich damit sogar MMS hätte machen können, aber mir ist nur RTP gelungen, was zwar „moderner" und vermutlich sinnvoller ist, aber nur relativ wenige Player hat. Schließlich müssen wir auch an Smartphone, iPhone, GoDi-Player (Raspi-Basis mit VLC-Player) und Windows-PCs denken.

Die Installation

Ich habe dann nach viel Testerei **Icecast als Server und Darkice als „Sound-Zulieferer"** zum Laufen bekommen. Der Link fängt dann zwar nicht mehr mit MMS, sondern mit HTTP an, aber da gibt es genügend Player, die das können. Außerdem hatte ich noch die Idee, den Streamserver portabel zu halten, um auch Streams außerhalb des üblichen Standortes senden zu können. Da LTE mit PrePaid ohne monatliche Bindung noch in den Kinderschuhen steckt, habe ich an UMTS gedacht, um einen Stream an einen Server im Internet zu senden, der ihn dann an mehrere Benutzer weiter streamt. Das wäre mit Icecast machbar.

Ich habe genau dieses mehrfach im Internet gefunden und daraus die folgende Anleitung zusammengestellt. Meine **Hardware** ist ein **Raspi B+** mit einer USB-Soundkarte von Behringer namens **UCA222**, die laut mehreren Personen keine Probleme am Raspi macht und so war es auch. An den Eingang der Soundkarte wird dann der Ausgang eines Mischpults angeschlossen, was wiederum für die Mischung und Verstärkung der angeschlossenen Mikrofone zuständig ist.

Grundvoraussetzung ist wieder alles, was in den Kapiteln zum Streaming-Client steht. So setze ich hier voraus, dass man den Editor Nano bedienen kann, ein aktuelles Raspbian **Jessie Lite** installiert und die ersten Schritte durch die „Raspi-Config" inklusive der Aktivierung des SSH gemacht hat. Dann geht es jetzt los!

Nach erfolgter „Raspi-Config" nutze ich nicht mehr die angeschlossene USB-Tastatur und den HDMI-Monitor, sondern Putty (siehe Streaming-Client Kapitel 4), um die Möglichkeit zu haben, Befehle aus dem Internet einfach und wenig fehlerbehaftet zum Raspi zu übertragen. Ich mache Befehle kenntlich, indem ich vor einen Befehl ein $ stelle und alles auch noch gelb markiere. Grau markierte Texte sind meistens Texte, die im Editor „Nano" eingegeben werden müssen.

Es beginnt mit einem Reboot und wer noch kein Update oder Upgrade gemacht hat, macht es jetzt::

```
$sudo reboot
$sudo apt-get update
$sudo apt-get upgrade -y
```

Anschließend dauert es ein paar Minuten. Wenn der Prompt wieder zu sehen ist, erfolgt ein Reboot bei gesteckter USB-Soundkarte:

```
$sudo reboot
```

Ob die Soundkarte erkannt wurde, sieht man einmal beim Prüfen des USB-Anschlusses nach dem letzten Reboot:

```
$lsusb
```

Bei meiner Soundkarte beinhaltet eine Zeile „Texas Instruments PCM2902 Audio Codec", was auf die Soundkarte hinweist. Um auch mit der Soundkarte aufnehmen zu können, sollte beim folgenden Befehl ein brauchbares Ergebnis herauskommen:

```
$arecord -l
```

Dort kommt „List of Capture Hardware Devices" und bei mir ist es „card 1" mit dem „subdevice #0". Wer nichts Vergleichbares bekommt, prüft seinen USB-Anschluss oder hat möglicherweise eine Soundkarte, die keine Aufnahmefunktion hat. (Im Alsamixer taucht für meine Soundkarte kein Capture-Device auf, aber das stört mich nicht.)

Beginnen wir jetzt mit der Installation von Darkice. Bei Jessie habe ich es gewagt, darkice auf die übliche Art zu installieren, was unter Wheezy nicht funktioniert hatte:

$sudo apt-get darkice

Darkice braucht eine Konfigurationsdatei. Ich habe die originale Datei extrem verkürzt und für diese Doku minimal verändert:

$sudo nano /etc/darkice.cfg

```
# sample DarkIce configuration file, edit for your needs before using
# see the darkice.cfg man page for details

# this section describes general aspects of the live streaming session
[general]
duration = 0 # duration of encoding, in seconds. 0 means forever
bufferSecs = 60 # size of internal slip buffer, in seconds
reconnect = yes # reconnect to the server(s) if disconnected

# this section describes the audio input that will be streamed
[input]
device = hw:1,0 # OSS DSP soundcard device for the audio input
sampleRate = 44100 # sample rate in Hz. try 11025, 22050 or 44100
bitsPerSample = 16 # bits per sample. try 16
channel = 2 # channels. 1 = mono, 2 = stereo

# this section describes a streaming connection to an IceCast2 server
# there may be up to 8 of these sections, named [icecast2-0] … [icecast2-7]
# these can be mixed with [icecast-x] and [shoutcast-x] sections
[icecast2-0]
bitrateMode = cbr # average bit rate
format = mp3 # format of the stream: ogg vorbis
bitrate = 128 # bitrate of the stream sent to the server
server = raspberry.internet.de # host name of the server
port = 8000 # port of the IceCast2 server, usually 8000
password = hackme # source password to the IceCast2 server
mountPoint = raspi # mount point of this stream on the IceCast2 server
name = Raspi # name of the stream
description = Raspi Stream # description of the stream
url = http://raspberry.internet.de # URL related to the stream
genre = mygenre # genre of the stream
public = no # advertise this stream?
localDumpFile = temp.mp3 # local dump file
[]
```

Es folgt die Installation des eigentlichen Streamingservers „Icecast2“:

$sudo apt-get install icecast2

Statt „localhost“ gibt man den tatsächlichen Namen an, wie z. B. servername.domain.de oder raspberry.fritz.box. Die Abfrage nach den drei Passwörtern habe ich im ersten Versuch auf Default gelassen. Es sollte sich alles auch später ändern lassen, wenn man in der Darkice.cfg das Passwort richtig setzt.

Es wird alles noch einmal aktualisiert mit nachfolgendem Reboot:

$sudo apt-get upgrade
$sudo reboot

Nun kommt der große Test! Der Stream wird gestartet!

$sudo service icecast2 start
$sudo darkice

Bei dieser Aktion läuft Darkice im Vordergrund, was sich später noch ändern wird. Nun kann man Icecast testen indem man die oben genannte Adresse in einem Browser aufruft. In meinem Beispiel wäre es:

http://raspberry.internet.de:8000

Es sollte ein funktionierender Stream angezeigt werden, wenn man zuerst „Public home“ und dann „Server status“ ausgewählt hat:

Wenn es deutlich weniger Einträge sind, als auf dem Bild, ist etwas falsch gelaufen und der Link wird nicht funktionieren! Besonders, wenn die ersten beiden Zeilen leer sind!

Anschließend kann man den Stream abspielen lassen indem man dem Link noch die Freigabe anhängt:

http://raspberry.internet.de:8000/raspi

Wenn man vorhat diesen Stream über das Internet laufenzulassen, sollte man unbedingt nur GEMA-freie Musik laufen lassen. Außerdem muss im Router (evtl. Fritz-Box) der Port 8000 für die interne IP-Adresse des Raspberry freigegeben werden!

Das war sicherlich nicht einfach, aber bei mir hat es funktioniert, wenn die darkice.cfg wirklich in Ordnung ist!

Generell sollte man sich überlegen, ob man bei funktionierendem Stream das SSH aus Sicherheitsgründen nicht wieder deaktiviert! Und dass man nicht das vorgegebene Passwort des Users Pi nutzt, ist eh klar, da es bereits im Juni 2017 einen Virus gab, der das ausgenutzt hatte! Der User Pi hat mittel „sudo“ Root-Rechte! Schlimmer geht’s nimmer!

Der Autostart

Ich wollte erreichen, dass der Streamingkram gleich nach dem Booten von alleine funktioniert, ohne dass sich jemand anmelden muss. Bisher hatte ich beim Streaming-Client den Autostart so gebaut, dass ein automatisches Login eines bestimmten Users erfolgte und daraufhin mit dessen Berechtigungen ein automatisches Skript abläuft. Beim Streamingserver bin ich ohne Login ausgekommen, weil ich den passenden Artikel im Internet gefunden habe! Und das auch noch für **Jessie**! 🙂

Ich trage also den Start-Befehl einfach am Ende einer „autostart-Datei" ein, aber vor „exit 0":

$ sudo nano /etc/rc.local

sudo darkice &

Damit startet nun der Stream automatisch. Das war doch nicht so schwierig, oder? 😉

Das Starten eines Webbrowsers, wie es zu Zeiten von Wheezy auf dieser Seite zu finden war, wurde vorerst entfernt, da es vermutlich nicht so ohne Weiteres mit Jessie Lite läuft. Da wird eine schwerere Jessie herhalten müssen! 😀