

Projektarbeit
in der Fakultät
Informatik

Access Point und Router mit embedded Board Banana Pi R1

Referent : Dr. Jiri Spale

Vorgelegt am : ??.??.2017

Vorgelegt von : Bastian
Elias
Jakob
Jonas
Tom

Abstract

English Abstract

Diese Dokumentation ist Teil des Projektes "Access Point und Router mit embedded Board Banana Pi R1", welches im Sommersemester 2017 an der Hochschule Furtwangen abgehalten wird. Das Projekt besteht daraus, das Vorgängerprojekt mit eigenen Implementierungen zu erweitern. Projektziele sind, das testen passender Betriebssysteme, sowie die Implementierung verschiedener Funktionen wie VPN, Radius, Samba, einem Mailserver und einer Displaystatusanzeige.

Inhaltsverzeichnis

Abstract	i
Inhaltsverzeichnis	iv
Abbildungsverzeichnis	v
Abkürzungsverzeichnis	vii
1 Projektübersicht	1
1.1 Kontext	1
1.2 Ziel	1
2 Betriebssystemeübersicht	3
2.1 OpenWRT	3
2.2 Bananian	3
2.3 IPFire	3
2.4 Armbian	3
3 Implementierung	5
3.1 Backup und Wiederherstellung des Betriebssystems	6
3.1.1 dd	6
3.1.2 rsync	8
3.1.3 tar	8
3.1.4 Automatisierung mittels Bash-Skript	9
3.2 Update des Banana Pi	10
3.3 Implementierung einer Displaystatusanzeige	11

3.4	Mail-Server	12
3.4.1	Einrichtung	12
3.5	Samba	14
3.5.1	Einrichtung	14
3.6	DDNS	16
3.6.1	Einrichtung No-IP	16
3.6.2	Einrichtung Custom-Domain	19
3.7	Routing, VLANs	21
3.8	WLAN	21
3.9	Radius	21
4	Benutzeranleitung	23
5	Lerneffekt - Eigenbewertung	25
6	Ausblick	27
	Literaturverzeichnis	29
	Eidesstattliche Erklärung	31

Abbildungsverzeichnis

Abbildung 1: Vorhandene Dateisysteme	6
Abbildung 2: Laufender Backupprozess mit Statusanzeige	7
Abbildung 3: Laufender Wiederherstellungsprozess mit Statusanzeige	7
Abbildung 4: Ausführung wöchentliches Backup	9
Abbildung 5: Verwendung von Tmux mit 'htop' und 'iftop'	11
Abbildung 6: noip Domain	16
Abbildung 7: noip Konfiguration	17
Abbildung 8: freenom Domain	19
Abbildung 9: freenom Nameserver	19
Abbildung 10: Cloudflare DNS	20

Abkürzungsverzeichnis

1 Projektübersicht

1.1 Kontext

1.2 Ziel

2 Betriebssystemeübersicht

2.1 OpenWRT

2.2 Bananian

2.3 IPFire

2.4 Armbian

3 Implementierung

3.1 Backup und Wiederherstellung des Betriebssystems

Um die Implementationen und Konfigurationen des Pi's abzusichern, wird eine Backup Lösung eingesetzt. Die Sicherung relevanter Daten soll hierbei einem möglichen Defekt oder Inkompatibilität durch Updates o.ä. entgegenwirken.

'FauBackup' und 'gitbac' bieten hierbei eine Lösung mittels externer Programme an. Debian bietet jedoch bereits standardmäßig einige Aufrufe welche zur Anlegung von Backups verwendet werden können. Diese wurden im Folgenden in Ubuntu anhand eines Bash Skripts zur Automatisierung getestet und implementiert.

3.1.1 dd

Als erste Ausführung wurde der Aufruf 'dd' verwendet. Hierbei wird der Inhalt der gesamten SD Karte als Image Datei abgespeichert.

Schritt 1:

Übersicht der vorhandenen Dateisysteme mittels des Terminalaufrufs 'df'

```

bastian@bastian-VirtualBox:~$ df
Dateisystem    1K-Blöcke  Benutzt  Verfügbar  Verw%  Eingehängt auf
udev           497668      4    497664      1%  /dev
tmpfs          101768     900    100868      1%  /run
/dev/sda1      9156984  5217368  3451424     61%  /
none            4          0         4          0%  /sys/fs/cgroup
none           5120          0     5120          0%  /run/lock
none          508832     76    508756      1%  /run/shm
none          102400     56    102344      1%  /run/user
none          455712764 373044796 82667968     82%  /media/sf_OrdnerWindows
bastian@bastian-VirtualBox:~$

```

Abbildung 1: Vorhandene Dateisysteme

Schritt 2:

Terminalaufruf für den Backupprozess

```
sudo dd if=INPUTPARTITION of=OUTPUTFILE
```

- sudo -> Backupprozess benötigt root-Rechte
- dd -> bit-genaues Kopieren der Dateien
- if=FILE -> Die Datei oder Partition welche integriert wird
- of=FILE -> Die Output Datei welche angelegt wird

Schritt 3:

Optionale Nutzung von Terminalaufruf 'pv' um den Fortschritt des Backup Prozesses zu sehen. Die mögliche Restzeit lässt sich nur durch das Hinterlegen der Größe der Partition anzeigen.

```
sudo dd if=INPUTPARTITION |pv| sudo dd of=OUTPUTFILE
```

```
bastian@bastian-VirtualBox:~$ sudo dd if=/dev/sda1 |pv| sudo dd of=/media/sf_OrdnerWindows/Backup.iso  
31,4MB 0:00:06 [5,28MB/s] [ <=> ]
```

Abbildung 2: Laufender Backupprozess mit Statusanzeige

Schritt 4:

Wiederherstellen eines hinterlegten Backups läuft ähnlich wie der ursprüngliche Prozess ab.

```
sudo dd if=OUTPUTFILE |pv| of=INPUTPARTITION
```

```
bastian@bastian-VirtualBox:~$ sudo dd if=/media/sf_OrdnerWindows/Backup.iso |pv -s 10G| sudo dd of=/dev/sda1 bs=4096  
56,8MB 0:00:12 [5,23MB/s] [> ] 0% ETA 0:35:52
```

Abbildung 3: Laufender Wiederherstellungsprozess mit Statusanzeige

Quelle: <https://wiki.ubuntuusers.de/dd/> [1]

3.1.2 rsync

Da der Vorgang mittels 'dd' als suboptimal angesehen wird, wurde alternativ der Aufruf 'rsync' verwendet. Dieser bietet die Möglichkeit eines inkrementellen Backups wodurch die Dauer des Prozesses erheblich reduziert werden kann. Hierbei werden die Größe und die Änderungszeit der Dateien in Quelle und Ziel miteinander verglichen. Eine Aktualisierung findet demnach nur statt, wenn Unterschiede vorzufinden sind.

```
rsync -aAXv --delete --exclude={"/dev/*","/proc/*","/sys/*",
    /*","/tmp/*","/run/*","/mnt/*","/media/*","/lost+found"
    "} / /path/to/backup/folder
```

- rsync -> Kopieren der Dateien
- aAX -> Übertragung im Archiv Modus wodurch alle symbolischen Verweise beibehalten werden
- delete -> Dateien die im Ursprungsverzeichnis nicht mehr existieren werden im Zielverzeichnis ebenfalls gelöscht
- exclude -> Dateien werden ausgelassen

Wiederherstellen des Rsync Backups durch folgenden Befehl:

```
rsync -aAXv /path/to/backup/location/* /mount/point/of/
    new/install/ --exclude={"/dev/*,/proc/*,/sys/*,/tmp/*,/
    run/*,/mnt/*,/media/*,/lost+found,/home/*}
```

3.1.3 tar

Eine weitere Anwendungsmöglichkeit bietet die 'tar' Archivierung. Vorteil dieses Aufrufs ist, dass durch Angabe von Parametern die Berechtigungen aller zu sichernden Daten ebenfalls beibehalten werden und die Archivierung Speicherplatz spart.

Wechsel in das Backupverzeichnis dann:

```
tar -cpzf Backup.tar ORDNER
```

- tar -> Archivieren von Daten
- c -> Archiv wird erzeugt (create)
- p -> Berechtigungen beibehalten (privilege)
- z -> Zusätzliche Komprimierung mit gzip
- f -> Archiv in Datei schreiben (finish)

3.1.4 Automatisierung mittels Bash-Skript

Damit der Nutzer die Aufrufe nicht händisch zu bestimmten Zeiten ausführen muss, wurden zwei Bash-Skripte zur Automatisierung geschrieben. Es gibt ein monatliches Backup mittels (tar) und wöchentliche inkrementelle Backups (rsync) auf die zurückgegangen werden kann.

Um das Zeitintervall der Backups einzustellen wird der 'Cron' Dienst verwendet. Hiermit können Skripte und Programme zu festgelegten Zeiten gestartet werden. Wenn ein hinterlegter Job täglich zu einer bestimmten Uhrzeit ausgeführt wird muss allerdings auch der Rechner zu dem Zeitpunkt aktiv sein. Ist dies nicht der Fall, startet der Prozess nicht. Um dies zu umgehen wird 'Anacron' verwendet. Durch ablegen des Skripts in eines der entsprechenden Verzeichnisse wird der Prozess entsprechend ausgeführt.

- /etc/cron.hourly/ - Stündlich ausführen
- /etc/cron.daily/ - Täglich ausführen
- /etc/cron.weekly/ - Wöchentlich ausführen
- /etc/cron.monthly/ - Monatlich ausführen

```
Woechentliches Backup wird ausgefuehrt. Diesen Vorgang bitte nicht abbrechen!  
sending incremental file list  
deleting security.update.log  
rsync: symlink "/media/sf_OrdnerWindows/Backup/initrd.img" -> "boot/initrd.img-3.13.0-119-generic" failed: Read-only file system (30)  
rsync: symlink "/media/sf_OrdnerWindows/Backup/initrd.img.old" -> "boot/initrd.img-3.13.0-117-generic" failed: Read-only file system (30)  
rsync: symlink "/media/sf_OrdnerWindows/Backup/vmlinuz" -> "boot/vmlinuz-3.13.0-119-generic" failed: Read-only file system (30)  
rsync: symlink "/media/sf_OrdnerWindows/Backup/vmlinuz.old" -> "boot/vmlinuz-3.13.0-117-generic" failed: Read-only file system (30)  
./
```

Abbildung 4: Ausführung wöchentliches Backup

3.2 Update des Banana Pi

Ziel war es das Betriebssystem und alle Programme immer auf dem aktuellsten Stand zu halten. Hierbei kann es jedoch zu Inkompatibilität bestimmter Funktionen oder Konfigurationen kommen. Daher wurde die Umsetzung auf die relevantesten Updates (Sicherheitsupdates) reduziert. Im Normalfall können mittels des Konsolenaufrufs 'apt-get update' die Updateliste und mit 'apt-get upgrade' die Programm Pakete selbst aktualisiert werden. Durch Nutzung des Aufrufs 'unattended-upgrade' wird auf Sicherheitsupdates des Systems überprüft und diese anschließend installiert.

Durch 'unattended-upgrade --dry-run -d' wird auf Verfügbarkeit von Updates geprüft ohne anschließende Installation. Nach jedem Durchgang wird eine Logdatei in /var/log/unattended-upgrades/ angelegt welche genauere Informationen zu den aktualisierten Dateien liefert.

3.3 Implementierung einer Displaystatusanzeige

Zur besseren Übersicht des Netzwerktraffics als auch der Ressourcen des Banana Pi sollte eine Displaystatusanzeige implementiert werden. Der Aufruf 'htop' bietet eine Übersicht aller laufenden Prozesse und deren Ressourcennutzung. 'iftop' zeigt die Netzwerkinterfaces und die eingehende und ausgehende Kommunikationen. Da das Terminal jedoch nur einen der Befehle zu einem Zeitpunkt ausführen kann wird ein Terminal-Multiplexer verwendet. Zur Auswahl stehen hierbei 'Terminator', 'screen', und 'Tmux'. Aufgrund der geringen Einarbeitungszeit und einfachen Anwendbarkeit wurde letzteres zur Implementation ausgewählt. Alle Multiplexer bieten die Möglichkeit Sitzungen zu erstellen. Leider kann dies nicht zur Implementierung der Displaystatusanzeige verwendet werden, da die Sitzung beim Herunterfahren des Betriebssystems gelöscht wird. Daher wird zum Systemstart ein Bash-Skript eingesetzt, welches automatisch die benötigten Fenster zur Überwachung anlegt.

```

bastian@bastian-VirtualBox:~/Dokumente/Studium_HFU/ProjektSS2017/BackupRestoreUpdate/Fertige_Skripte/Ubuntu$

```

```

1  [|||] 3.3% Tasks: 134, 194 thr; 1 running
2  [|||] 6.7% Load average: 0.26 0.13 0.10
Mem [|||||] 563/993MB Uptime: 01:02:01
Swp [|||] 2/1021MB

```

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Comm
2176	bastian	20	0	1389M	63908	15240	S	4.7	6.3	0:30.17	comp
1057	root	20	0	455M	90016	8088	S	3.3	8.8	0:28.12	/usr
3574	root	20	0	31440	1984	1400	R	2.7	0.2	0:00.43	htop
2492	bastian	20	0	638M	16612	5504	S	0.7	1.6	0:07.57	gnom
1096	mysql	20	0	609M	50116	900	S	0.7	4.9	0:05.85	/usr
1320	root	20	0	235M	1072	664	S	0.7	0.1	0:02.25	/usr
2230	bastian	20	0	857M	26360	13044	S	0.0	2.6	0:04.08	naut

```

TX: cum: 0B pearates: 0b 0b 0b
RX: 0B 0b 0b 0b
TOTAL: 0B 0b 0b 0b

```

ifftop: 16:29:22 May 17

Abbildung 5: Verwendung von Tmux mit 'htop' und 'iftop'

3.4 Mail-Server

Um Statusbenachrichtigungen zu erhalten, wurde ein Mail-Server auf dem Pi eingerichtet, welcher als Relay über Google Mail fungiert. Dazu wurde ein Google Mail Konto eingerichtet, auf welches Post Fix zugreift und Mails verschickt.

Alle System Mails werden Über das Google-Konto weitergeleitet, dazu gehören auch Statusmeldungen des Backup-Skripts.

Dieser Weg wurde wegen des geringen Aufwands gewählt. Ohne Relay würde man eine eigene Domain und sehr viel mehr Konfiguration benötigen. Da ein Google Mail Konto nie verfällt, war dies die beste und pflegeleichteste Möglichkeit.

Weiterhin können die Mails auch an jede beliebige Adresse verschickt werden, dies ist im Mail-Server frei konfigurierbar. Die Mails gehen momentan an die Google Mail Adresse.

3.4.1 Einrichtung

Voraussetzungen:

- Internetzugriff
- Google Mail-Konto
- Texteditor (Nano)
- SSH/Physikalischen Zugriff

1. Post Fix installieren:

```
apt-get update  
apt-get install postfix libsasl2-modules bsd-mailx
```

2. Das Konfigurationsfenster öffnet sich.

3. TLS/SSL aktivieren:

```
nano /etc/postfix/main.cf
```

3.1 Folgendes Einfügen:

```
mtp_sasl_auth_enable = yes  
smtp_sasl_security_options = noanonymous  
smtp_sasl_password_maps = hash:/etc/postfix/sasl_password  
# verschluesselung einschalten  
smtp_tls_security_level = may
```

4. Nutzerdaten des Google Mail-Kontos hinterlegen:

```
nano /etc/postfix/sasl_password  
smtp.gmail.com Bananapihfu:<Password>
```

5. Datei nur für root lesbar machen, da Klartext:

```
chmod 600 /etc/postfix/sasl_password
```

6. Postfix lookup Tabelle erstellen:

```
postmap hash:/etc/postfix/sasl_password
```

7. Postfix neustarten:

```
/etc/init.d/postfix restart
```

8. Für Weiterleitung der Systemnachrichten aliases bearbeiten:

```
nano /etc/aliases  
root: bananapihfu@gmail.com
```

Oder Wunschemail an welche Systemnachrichten gesendet werden

9. Änderungen an Aliases wirksam machen:

```
Newaliases
```

3.5 Samba

Um den Dateizugriff zu erleichtern, wurde ein Samba-Server auf dem Pi implementiert.

Samba ermöglicht es von nahezu jedem Gerät auf ein freigegebenes Verzeichnis auf dem Server zuzugreifen. Voraussetzung ist, dass das Client Betriebssystem das SMB-Protokoll unterstützt.

Die meisten modernen Betriebssysteme, wie Windows, MacOS und andere Unix-Systeme besitzen Samba Funktionalität.

3.5.1 Einrichtung

Voraussetzungen:

- Internetzugriff
- Texteditor (Nano)
- SSH/Physikalischen Zugriff

1. Samba installieren:

```
sudo apt-get update  
sudo apt-get install samba
```

2. Benutzer für Samba erstellen (Hat keinen Shell-Zugriff):

```
useradd sambaur --shell /bin/false
```

3. Passwort für den Benutzer in Samba setzen:

```
smbpasswd -a <user_name>
```

4. Verzeichnis im Home erstellen:

```
mkdir /home/sambaur  
mkdir /home/sambaur/samba
```

5. Berechtigungen setzen:

```
chown sambaur:sambaur /home/sambaur/  
chown sambaur:sambaur /home/sambaur/samba/
```


6. Backup der Samba Konfiguration im Homeverzeichnis machen:

```
cp /etc/samba/smb.conf ~
```

7. Config bearbeiten:

```
nano /etc/samba/smb.conf
```

7.1 folgendes am Ende der Konfiguration Einfügen:

```
[samba]  
path = /home/sambausr/samba  
valid users = sambausr  
read only = no
```

8. Service neustarten:

```
service smbd restart
```

9. Config testen:

```
testparm
```

3.6 DDNS

Um immer die aktuelle IP-Adresse des Pi zur Hand zu haben, wurde ein DynDNS-Client von no-ip implementiert, bei jedem Systemstart wird die bei der DNS hinterlegte IP-Adresse aktualisiert.

Nachteil hierbei ist jedoch, dass man alle 30 Tage diese Domain aktivieren muss, da diese sonst verfällt.

Um dies zu umgehen wurde die Kostenfreie Domain bananapihf.tk gebucht. Von Vorteil ist hier, dass diese Domain eine Laufzeit von einem Jahr hat und nach Ablauf auch ohne Mehrkosten verlängert werden kann.

Diese Domain wird bei Cloudflare verwaltet, da hier auch eine API angeboten wird, mit welcher man Theoretisch komplett auf eine Dynamische DNS bei No-IP verzichten kann.

3.6.1 Einrichtung No-IP

Voraussetzungen:

- Internetzugriff
- SSH/Physikalischen Zugriff
- NoIP.com Account

1. NoIP Hostnamen anlegen:

Man muss in No-IP den gewünschten Hostnamen einrichten, welcher später auf die aktuelle IP-Adresse verweist.

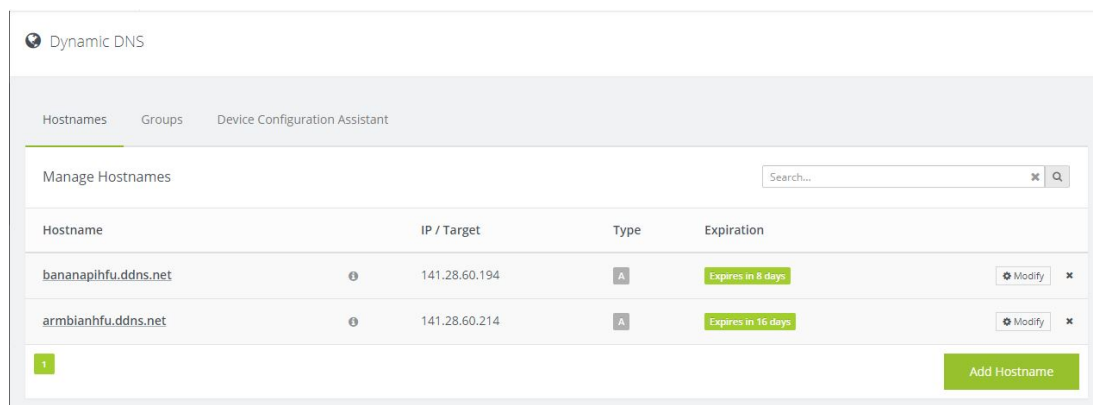


Abbildung 6: noip Domain

2. Verzeichnis für DDNS im Homeverzeichnis erstellen und wechseln:

```
mkdir DDNS
cd DDNS
```

3. NoIP-Client von NoIP.com beziehen:

```
wget https://www.noip.com/client/linux/noip-duc-linux.tar.gz
```

4. Das Archiv entpacken:

```
tar xvf noip-duc-linux.tar.gz
```

5. In den Entpackten Ordner wechseln:

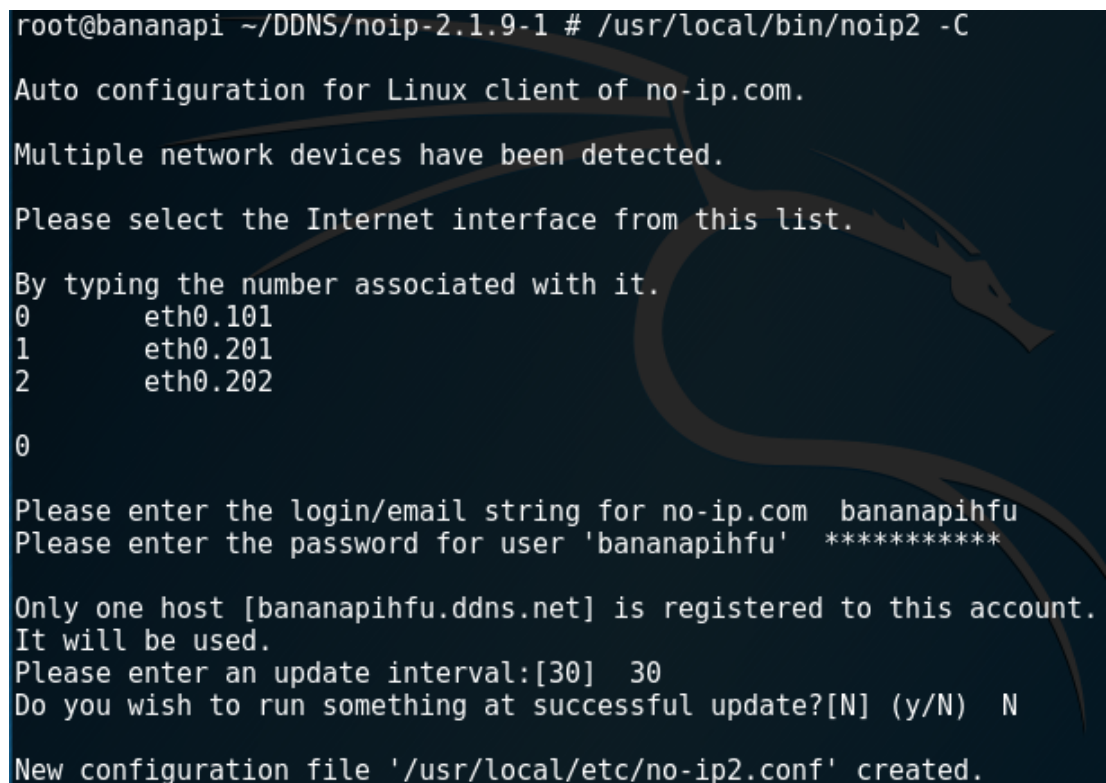
cd noip eingeben und mit TAB vervollständigen

```
cd noip-x.x.x-x
```

6. Client bauen und installieren:

```
make && make install
```

7. Konfiguration wird gestartet:

A terminal window showing the configuration process for the noip2 client. The prompt is root@bananapi ~. The user runs /usr/local/bin/noip2 -C. The program outputs: 'Auto configuration for Linux client of no-ip.com. Multiple network devices have been detected. Please select the Internet interface from this list. By typing the number associated with it.' It lists three interfaces: 0 eth0.101, 1 eth0.201, and 2 eth0.202. The user enters 0. Then it asks for the login/email string (bananapihf) and the password (masked with asterisks). It then states that only one host (bananapihf.ddns.net) is registered. It asks for an update interval (30) and if the user wants to run something at successful update (N). Finally, it says a new configuration file has been created at /usr/local/etc/no-ip2.conf.

```
root@bananapi ~/DDNS/noip-2.1.9-1 # /usr/local/bin/noip2 -C
Auto configuration for Linux client of no-ip.com.
Multiple network devices have been detected.
Please select the Internet interface from this list.
By typing the number associated with it.
0      eth0.101
1      eth0.201
2      eth0.202
0

Please enter the login/email string for no-ip.com  bananapihf
Please enter the password for user 'bananapihf'  *****

Only one host [bananapihf.ddns.net] is registered to this account.
It will be used.
Please enter an update interval:[30]  30
Do you wish to run something at successful update?[N] (y/N)  N

New configuration file '/usr/local/etc/no-ip2.conf' created.
```

Abbildung 7: noip Konfiguration

Um den Daemon automatisch bei Systemstart zu starten muss noch folgende Konfiguration vorgenommen werden:

1. Startscript unter `/etc/init.d/noip2` ablegen:

```
vim /etc/init.d/noip2
```

2. Script "noip2" kopieren und einfügen:

Verweis auf Anhang

3. Script ausführbar machen:

```
chmod a+rx /etc/init.d/noip2
```

3.6.2 Einrichtung Custom-Domain

Voraussetzungen:

- Account bei Cloudflare

1. Domain bei Freenom.com aussuchen:

Hier gibt es jede Menge kostenfreie Domains

2. Wunsch-Domain registrieren

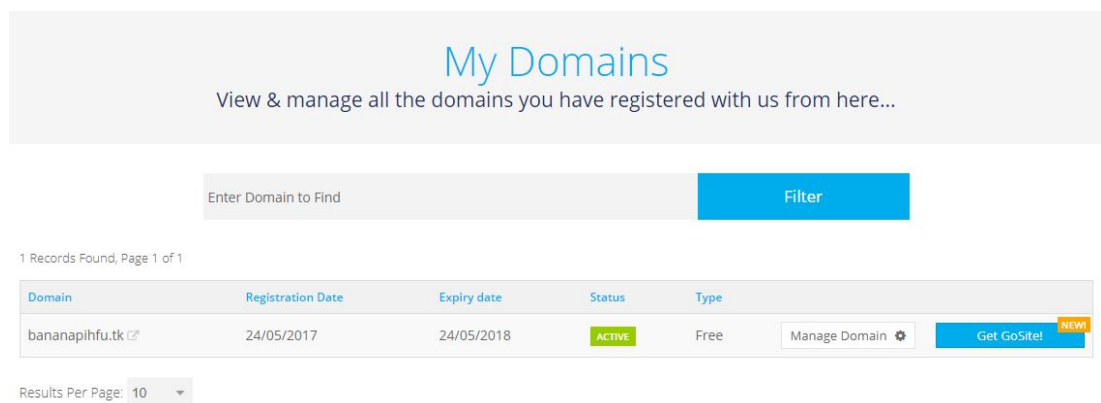


Abbildung 8: freenom Domain

3. Nameserver bei Freenom ändern:

Um die Domain bei Cloudflare zu verwalten, müssen die Nameserver angepasst werden. Diese erhält man, wenn man sich bei Cloudflare anmeldet.

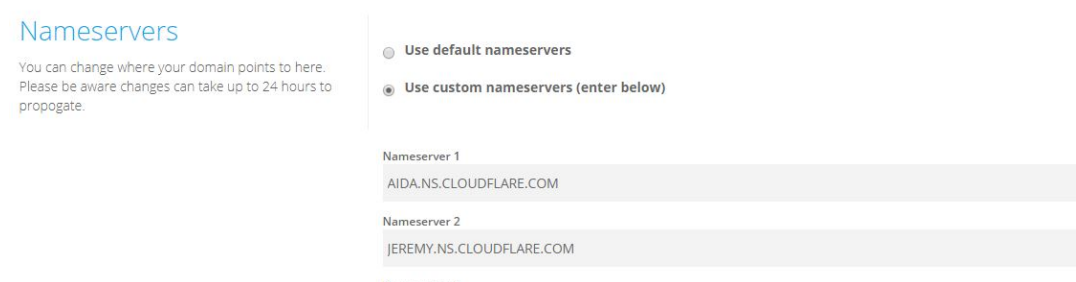


Abbildung 9: freenom Nameserver

3. CNAME Weiterleitung auf No-Ip einrichten

DNS

Manage your Domain Name System (DNS) settings.

DNS Records

A, AAAA, and CNAME records can have their traffic routed through the Cloudflare system. Add more records using this form, and click the cloud next to each record to toggle Cloudflare on or off.

- ⚠ An A, AAAA or CNAME record was not found for the **www** subdomain. The **www.bananapihf.tk** subdomain will not resolve.
- ⚠ An A, AAAA or CNAME record was not found pointing to the root domain. The **bananapihf.tk** domain will not resolve.
- ⚠ An MX record was not found for your root domain. An MX record is required for mail to reach **@bananapihf.tk** addresses.

A

Name

IPv4 address

Automatic TTL

Add Record

Type	Name	Value	TTL	Status
CNAME	armbian	is an alias of armbianhf.ddns.net	Automatic	
CNAME	bananian	is an alias of bananapihf.ddns.net	Automatic	

[Advanced](#) [API](#) [Help](#)

Abbildung 10: Cloudflare DNS

3.7 Routing, VLANs

3.8 WLAN

3.9 Radius

4 Benutzeranleitung

5 Lerneffekt - Eigenbewertung

6 Ausblick

Literaturverzeichnis

- [1] dd > wiki > ubuntuusers.de. <https://wiki.ubuntuusers.de/dd/>. Zuletzt
Aufgerufen: 24.06.2017.

Eidesstattliche Erklärung

Ich versichere, dass ich die vorstehende Arbeit selbständig verfasst und hierzu keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Stellen der Arbeit die wörtlich oder sinngemäß aus fremden Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt oder an anderer Stelle veröffentlicht.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

Furtwangen, den ??.???.2017