

Slug! - Die NSLU2 Modifikation

Inhaltsverzeichnis

Über die NSLU2	<i>Seite 2</i>
Neue Firmware	<i>Seite 5</i>
Was ist, wenn es mal in die Hose geht?	<i>Seite 6</i>
Unslingen: Die Kiste wird aufgemacht	<i>Seite 10</i>
Eine neue Bash installieren	<i>Seite 14</i>
Midnight Commander installieren	<i>Seite 16</i>
FTP-Server aufsetzen	<i>Seite 17</i>
Apache Aufsetzen	<i>Seite 20</i>
PHP Erweiterung	<i>Seite 21</i>
Mysql Datenbank und gesicherte Verzeichnisse	<i>Seite 23</i>
Crontabs: Alles zur seiner Zeit	<i>Seite 28</i>
Chkrootkit: Der Kammerjäger kommt!	<i>Seite 30</i>
Kurzer Prozess mit htop	<i>Seite 32</i>
Festplatte spin(n)t hoch und runter	<i>Seite 33</i>
Die Slug als Downloadmaschine	<i>Seite 36</i>
Sicher ist sicher... die Secureshell	<i>Seite 40</i>
1001 Konsolen mit Screen	<i>Seite 40</i>
MP3 Server mit Firefly – Die Mücke und der Elefant	<i>Seite 41</i>
Firewall und IP Falle	<i>Seite 47</i>
Phönix aus der Asche: Kaputt für mehr Power	<i>Seite 69</i>
Hinweise	<i>Seite 72</i>

Was ist eine NSLU2?

... zumindest dachte ich mir das, als ich zum ersten Mal davon hörte. NSL steht als Abkürzung für Network Storage Link. Das U selbst kann Unit oder auch USB bedeuten. Dieses spezielle Gerät, das hier auf dieser Seite etwas näher besprochen wird, wird von der Firma Linksys vertrieben.

Derartige Geräte werden vom Grund her als Server für Datenspeicher innerhalb eines Netzwerkes verwendet. Das heißt an einer NSL-Box wird ein Datenträger angeschlossen, z.B. eine externe USB-Festplatte oder ein Flashspeicher, der für alle Geräte innerhalb eines Netzwerkes zur Verfügung steht, je nachdem der Zugriff auf den Datenspeicher geregelt ist. Vielerorts werden für solche einfachen Aufgaben ganze Rechner in einem Netzwerk abgestellt. Der Vorteil solcher Lösungen liegt klar auf der Hand: Sie sind einfach zu konfigurieren, nehmen weniger Platz als ein kompletter Rechner ein, verbrauchen weniger Strom als ein "echter" Server und sind natürlich kostengünstig. Derzeit kostet ein neues NSLU2 um die 80,- Euro. Wer eine ausrangierte Festplatte zur Verfügung hat, der muss nur noch ca. 40,- Euro für ein USB-Gehäuse berappen. Somit kommt eine derartige Lösung auf rund 120,- Euro.

Das folgende Bild zeigt das Größenverhältnis der kleinen Box:



(Abb. 1: Die NSLU2 im Verhältnis zu einem Feuerzeug)

Sie lässt sich somit recht einfach verbauen und stört auch nicht durch irgendwelche Laufgeräusche, da das Gerät selbst keine Lüfter verwendet. Hörbar ist nur die externe Platte selbst. Aber der Geräuschpegel hängt ebenso stark von der Qualität der verbauten Platte ab. Ich selbst verwende eine alte ATA100er IBM Harddisk, mit einer Umdrehung von 7200Rpm und einer Speicherkapazität von 120 GB.



(Abb.2 : Die NSLU2 mit einer angeschlossenen externen USB Festplatte)

Betrieben wird das Gerät dann an einem Switch oder auch Router, über den sämtliche Zugriffe innerhalb des Netzwerkes geregelt werden. Bei Anschaffung eines Routers, der vermutlich Anfragen auf ein DSL-Modem weiterleiten soll, sind für den erweiterten Betrieb einer NSLU2 (Web- Und/oder FTP-Server) folgende Eigenschaften von Vorteil:

- Firewall/Virtual Server Routing: Hier wird die Anfrage von außerhalb gesteuert. Möchte man z.B. auf eine Website zugreifen, dann leitet der Router diese Anfrage auf eine festgelegte interne IP-Nummer weiter. Diese ist dann eben die für den NSLU2 festgelegte IP. Die NSLU2 kann durch die Installation einiger zusätzlicher Softwarepakete nachdem diese modifiziert wurde einige Serverdienste zur Verfügung stellen. Doch damit man auf diese auch von außen Zugriff bekommt, müssen die entsprechenden Anfragen via Port-Redirection an die kleine Box weiter gegeben werden. Das kann der Port 80 für den http Dienst als auch der Port 21/22 für den ftp sein. Sollte der Router keine Redirection beherrschen, so wird es nicht möglich sein, die Box für den externen Betrieb sicher zu konfigurieren.

- Dynamic DNS: Da man bei der Wiederanwahl in das Internet in der Regel eine neue IP bekommt, wird der Zugriff auf einen eigenen Server unnötig erschwert. Hierzu legt man sich dann z.B. über eine Service einen eigenen Domain-Namen fest, der dann bei der Wiedereinwahl die eigene IP-Nummer automatisch übermittelt bekommt. Somit ist man immer über den festgelegten Domain-Namen erreichbar und muss somit nicht mehr mit

komplizierten IP-Nummern hantieren. Es gibt hierfür einige Anbieter wie das DynDNS, 2mydns oder MyIP um nur drei aus einer Vielzahl an Möglichkeiten zu nennen. Eine Ausführliche Liste findet sich unter folgendem Link:

<http://www.technopagan.org/dynamic/>

Dennoch kann nicht jeder Router mit allen Serviceanbietern kommunizieren. Von daher sollte man sich auch vor dem Kauf eines Routers informieren, welche Anbieter unterstützt werden, wenn es denn ein bestimmter Anbieter sein muss.

-Auto Reconnect: Der Router sollte sich selbständig wieder an das Internet verbinden, sobald die Verbindung seitens des Providers getrennt wird. Hier bitte zwei Dinge beachten: Zum einen ist dies natürlich nur interessant, wenn ein zeit- und volumenunabhängiger Tarif abgeschlossen wurde, also eine klassische Flatrate. Alles andere wird TEUER!!! Zum anderen bitte darauf achten, ob hierbei nicht vertragliche Regelungen seitens Eures Providers derartige Nutzung untersagen. Bitte informiert Euch über beide WICHTIGEN Dinge. Denn beides kann Geld kosten!

Wieso nun gerade dieses Gerät?

Nach kurzer Zeit hat sich herausgestellt, dass im Inneren der NSLU2 ein normales Linux schlummert, das nun viel mehr kann, als nur Daten über das Netz zu schaufeln. Muss auch so sein, denn das Gerät wird über ein Web-Interface administriert, was zumindest schonmal einen Webserver im Inneren vermuten lässt. Und noch viel mehr! Der Datenspeicher ist zudem ein Samba-Server, der nun betriebssystemunabhängig verschiedene Betriebssysteme versorgen kann. Nachdem findige Leute die Maschine sich zur Brust genommen haben, war es sehr schnell möglich, weitere Programme hierauf zum Laufen zu bringen. Somit kann das Gerät auch als normaler Webserver, als FTP-Server, Mail-Server und auch für den Anschluss einer Webcam genutzt werden. Dabei verliert es nicht die Funktionalität, die es von Haus aus mitbringt: Ein Datenserver im Netz zu sein.

Ich werde nun hier diverse Bearbeitungsschritte protokollieren, die ich selbst an diesem Gerät anhand diverser Anleitungen im Netz (div. FAQs und Newsgroup-Beiträge) durchprobiert habe, um zu demonstrieren, wieviel Power in diesem erstmal recht unscheinbar wirkenden Gerät steckt. Genug, um hierbei auch ein kleines Büro mit Daten zu versorgen und über das Internet für einen Datenaustausch erreichbar zu machen.

Ich wünsche viel Spaß mit meinem Dokument, der Slug und Euch beim Bateln an der kleinen Box!

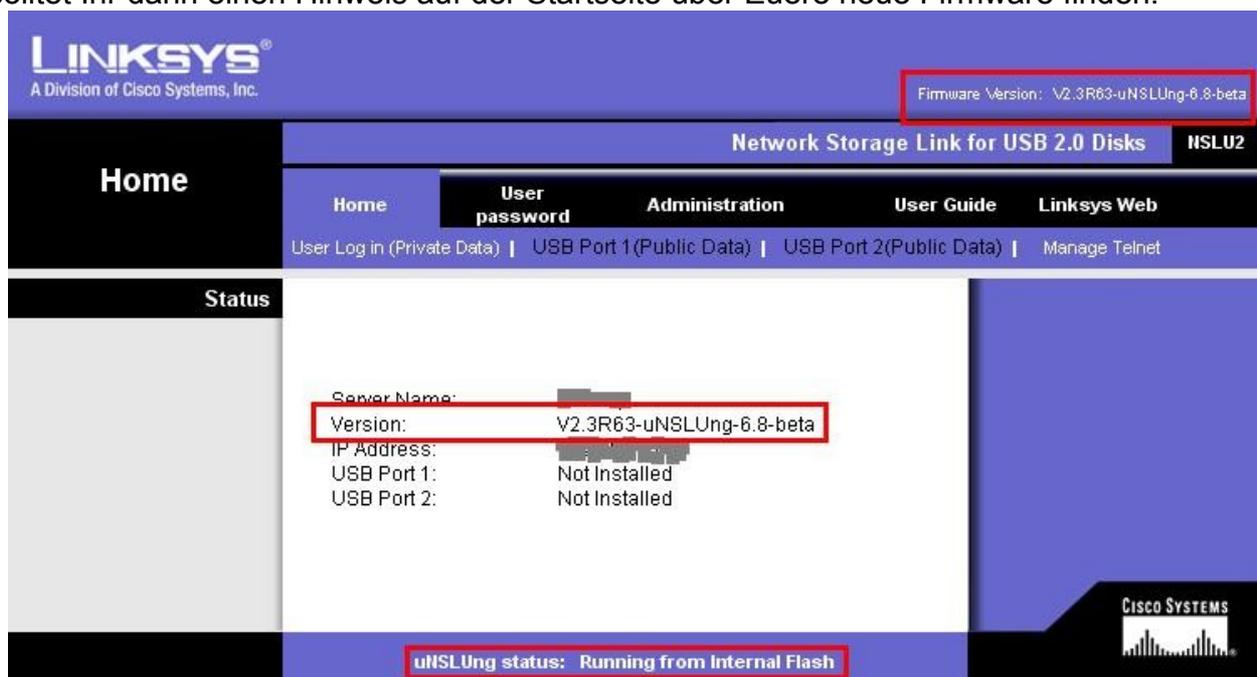
All gates are open now!

Wie flashe ich das Gerät nun?

Dies könnt Ihr im Administratorenpanel auf dem Webinterface mittels dieses Tools durchführen. Nachdem Ihr das Gerät wie in der mitgelieferten Anleitung in Betrieb genommen habt zuerst bitte die Festplatten abziehen. Beim Update darf die Festplatte **NICHT** angeschlossen sein. Jetzt als Admin (**Administration** anklicken) anmelden und auf **Advanced**. Hier dann **Upgrade** anklicken. Die Firmware als *.bin- File kann dann dort angegeben werden.



Danach auf "**Start Upgrade**" klicken und abwarten. Das Tool startet die NSLU2 dann selbständig. Bitte den Updatevorgang NICHT unterbrechen. Es dauert in der Tat ein wenig, bis Ihr eine Erfolgsmeldung bekommt und das Gerät neu gestartet wird. Nach dem Reboot solltet Ihr dann einen Hinweis auf der Startseite über Euer neue Firmware finden:



Aktuelle Firmware bekommt Ihr unter folgender Internetadresse:

<http://www.slug-firmware.net/u-dls.php>

Die Originalfirmware ist hier erhältlich:

<http://www.linksys.com/>

Sollte das Gerät dabei "zerflasht" werden ist Ruhe zu bewahren! Man glaubt es fast kaum, aber ich persönlich habe den Eindruck, dass man mit einem großen Hammer auf die Kiste hauen muss, damit es tatsächlich kaputt ist. In den meisten Fällen lässt sich ein fehlgeschlagener Flash auch wieder retten, ohne dass das Gerät eingeschickt werden muss. Dazu bitte mein Kapitel "Notoperation!" lesen. Wer sich es zutraut, kann auch auf diese Art und Weise gleich die Firmware auf den internen Speicher schreiben.

Notoperation!

Sollte irgendwann mal nichts mehr gehen, dann muss eine kleine Notoperation durchgeführt werden. Symptom: Das Gerät lässt sich nichtmehr ansteuern, kein Ping funktioniert und auch das mitgelieferte Setup-Programm von CD erkennt das Teil nicht mehr. Um das zu reparieren muss das Gerät neu geflasht werden.

Variante 1 (Windows bis maximal Windows XP/Vista 32bit evtl.)

Ihr benötigt dazu ein Biosfile mit der Version 2.3R25, das Ihr hier herbekommt:

<ftp://ftp.linksys.com/pub/network/nslu2-fw-2.3r25.zip>

Dazu einen Flasher, den Ihr hier herunterladen könnt:

<http://www.nslu2-linux.org/wiki/Main/SercommFirmwareUpdater>

Den Flasher installiert dann gleich mal bevor Ihr weiter macht.

Bitte geht nun wie folgt vor:

1.) Zuerst muss das Netzwerk auf **192.168.0.x** umgestellt werden. Dabei darf die Nummer **192.168.0.1** von keinem weiteren Gerät belegt sein, denn diese benötigt unsere kleine Box.

2.) Steckt nun das Gerät an, schaltet es aber noch nicht ein! Jetzt eine Telnetsession mit **telnet 192.168.0.1 9000** aufmachen, aber **NOCH NICHT** mit Enter abschicken, sondern erstmal nur in die Dos-Shell eingeben!

3.) Jetzt muss es ganz schnell gehen! Schickt den Telnetbefehl ab und drückt dabei gleichzeitig die Einschalttaste am Gerät. Wenn die Anzeige "**Verbinden....**" in eine erste Biosmeldung umschaltet müsst Ihr sofort die Taste **STRG+C** drücken. Wenn Ihr nun ein "RedBoot>" am Prompt erscheinen seht, dann seid Ihr "drin". Falls nicht muss die Prozedur wiederholt werden, bis Ihr den richtigen Moment trefft.

4.) Nun gebt "**upgrade**" ein und drückt **Enter**. Jetzt müßte ein Status-LED am Gerät grün/orange blinken.

5.) Startet nun das installierte Upgrade-Utility. Wählt hier Eure Netzwerkkarte aus und auf "**Browse**"->"**Browse Targets**" klicken. Nun müsstet Ihre eine MAC-Adresse sehen, die Ihr anklickt. Dann auf Files und Euer Bios (*.bin Datei) auswählen. Auf **Öffnen** klicken und mit **OK** bestätigen. Jetzt startet den Upgradevorgang indem Ihr auf den Button **Upgrade** drückt.

6.) Das alles dauert nun ne kleine Weile. Zuerst wird das Eeprom gelöscht, danach neu aufgespielt und dann überprüft. Es versteht sich von selbst, dass hier nicht unterbrochen werden darf! Wenn alles gut gelaufen ist, dann sollte die Meldung "**Upgrade successfully**" erscheinen, die Ihr mit "**OK**" wegklickt.

7.) Das Gerät startet von selbst neu. (Also wartet bis zum nächsten Piepsignal)

8.) Falls Ihr nun unter der alten IP nicht mehr auf Euer Gerät kommt, dann wurde es auf **192.168.1.77** zurückgestellt. Das deutet allerdings auf einen Fehler hin, der sich mit einer Fehlermeldung im Adminpanel zeigt: "**Can't get Samba Information**". Wenn das der Fall ist, dann muss noch der Configbereich im Eeprom gelöscht werden. Macht nochmal einen RedBoot wie oben beschrieben. Gebt dann am Prompt folgendes ein:

```
fis erase -f 0x50040000 -l 0x20000
```

(Hinweis: das -l ist ein kleines L !!!)

Bitte genau dies eingeben! Bei einen Tippfehler kann das Eeprom beschädigt werden! Danach ein

```
reset
```

eingeben. Dann funktioniert auch die Eingabe der Serverinfos im Adminstartionspanel wieder.

HINWEIS: Es kann leider sein, dass neuer Netzwerkkarte vom UGUTIL nicht erkannt werden. Hierzu ist dann eine ältere Standard-Karte zu diesem Zwecke einzusetzen. Bei mir wurde der Onboardchip eines Nforce4 Chipsatzes nicht erkannt. Dafür der Netgear WG111v2 USB Wlan-Stick. Ob ein Update über Funk ratsam ist sollte jeder für sich entscheiden.

HINWEIS 2: Bevor unter Windows ein Update vorgenommen wird, sollte die Softwarefirewall kurzfristig dafür abgeschaltet werden. Dies kann nur nervige Sperrungen hervorrufen, die das Updaten verzögern.

HINWEIS 3: UGUTIL läuft leider nicht unter Vista 64bit

Variante 2: Via upslug2 unter Linux

Der meine ich etwas einfachere Weg geht über Linux. Zunächst benötigt man das Programm upslug2. Dieses kompilieren wir selbst nachdem wir uns das Quellpaket besorgt haben:

```
svn co http://svn.nslu2-linux.org/svnroot/upslug2/trunk upslug2
```

Das erzeugt ein neues Unterverzeichnis upslug2. In dies wechseln wir dann:

```
cd upslug2
```

Danach starten wir den Kompile mit folgenden Befehlen:

```
autoreconf -i  
./configure  
make
```

Jetzt liegt das Programm upslug2 im Buildverzeichnis. Kopiert nun die aktuelle Firmware in das gleiche Verzeichnis.

Nun stellt Euer Netzwerk auf folgende IP um:

192.168.0.xxx

D.h. : Die NSLU2 wird wieder die IP 192.168.0.1 verwenden, somit muss der Linuxrechner eine andere Endnummer bekommen.

Jetzt versetzen wir die NSLU2 mit einem einfachen Trick in den Redboot- Modus: Schalte die Slug aus und nehmt eine Büroklammer zur Hand. Drückt damit auf der Rückseite des Gerätes den Reset-Button. Diesen haltet dann fest und drückt vorne auf den Einschaltknopf. Den Einschaltknopf lasst los und haltet hinten noch den Reset-Knopf gedrückt. Dann beobachtet die obere LED. Diese wird zuerst orange und nach ca. 10 Sekunden rot. Jetzt den Reset-Knopf loslassen! Wenn nun die LED Rot-Grün blinkt, dann ist diese bereit für den Flashvorgang.

Unslingen und User mit root-Rechte einrichten

Zuerst wird über das Administrationspanel die neue Firmware eingespielt. Das funktioniert menügesteuert. Danach müssen wir das Gerät "entsichern", damit eine Telnet-Session zugelassen wird und wir in das geheiligte Innere der Maschine gelangen können.

1. Startet die Kiste ohne angeschlossener Festplatte. Ruft jetzt im Browser unter Angabe der IP nebst des eventuell von Euch neu eingestellten voreingestellten Ports folgende URL auf:

<http://192.168.1.77:8080/Management/telnet.cgi>

Hier werdet Ihr nach dem Adminpasswort gefragt. Sollte das nicht verstellt sein, dann lautet es werkseitig admin:admin . Jetzt könnt Ihr bequem auf einen Button klicken, um die Telnetsession zu "**enablen**" (aktivieren). Vergesst bitte nicht, wenn Ihr mit Euren Arbeiten an der Slug fertig seid, hier wieder auf "**disable**" zu klicken, damit die Maschine vor Zugriffen gesichert ist.

Hinweis: Ab de unslug Version 6.8 kann man dies auch komfortabel auf der Startseite des Administrationspanel erledigen!!!

Schaltet nun von Euerer Shell aus auf eine Telnetsession

telnet [IP der Slug]

Hinweis: Ein sehr schönes Tool hierzu ist der Putty. Die Projektseite dazu findet Ihr hier:

<http://www.putty.nl/>

Richtet ein Profil für die NSLU2 im Putty ein (in der Konfiguration telnet anklicken, die IP der Slug eintragen und als neues Profil mit einem Namen -z.B. nslu2 - abspeichern) und legt eine Verknüpfung an. Diese erweitert Ihr mit dem Parameter -load [profilname] , also unter Windows mit unserem nslu2 Profil z.B. putty.exe -load nslu2 . Dann genügt ein Doppelklick auf die Verknüpfung und Ihr habt schon eine Telnetverbindung zum gewünschten Client vor Euch auf dem Bildschirm.

2. Telnet auf die Box und hier nun mit root: **uNSLUng** anmelden. Wenn Ihr als Root angemeldet seid, erhaltet Ihr in etwa diese Meldung:

```
login: root
Password:
No directory, logging in with HOME=/

Welcome to Unslung V2.3R63-uNSLUng-6.8-beta

----- NOTE: RUNNING FROM INTERNAL FLASH -----

This system is currently running from the internal flash memory,
it has NOT booted up into "unslung" mode from an external drive.

In this mode, very few services are running, and available disk
space is extremely limited. This mode is normally only used
for initial installation, and system maintenance and recovery.

BusyBox v0.60.4 (2005.03.22-06:52+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.
```

Danach die Festplatte wieder anschließen. Wartet nun ab, bis die Box die Festplatte als angeschlossen anzeigt (grünes LED an der Box, oder über das Web-Adminpanel). Jetzt seid Ihr in der Box schonmal drin. Damit nun das Betriebssystem auf die Festplatte ausgelagert wird (unslingen) gebt nun folgenden Befehl an der Konsole ein:

/sbin/unsling

Hinweis: Ab der Firmwareversion **6.8** wird die Festplatte hierzu an den USB-Port 2 angeschlossen. Der Befehl lautet dann

/sbin/unsling disk2

Danach kann hier sogar ein root-Passwort eingegeben werden, was dann für das System generell gilt. Wenn das Unslingen fertig ist das Gerät neu starten. Die Platte bleibt am USB-Port2!!!

Es dauert eine Weile, bis die Sache erledigt ist. Jetzt kann aufgrund des wesentlich größeren Speicherplatzes später entsprechend neue Software installiert werden.

3. Ruft in Euerem Browser das Administrationspanel auf und legt einen neuen User an (z.B. **Testi**). Dieser soll auch ruhig ein Heimatverzeichnis bekommen, also hier das im Konfigmenü auswählen. Speichern und beenden.

4. In der Telnetkonsole dann **vi /etc/passwd** eingeben. Mit "i" in den Editmodus wechseln.

5. Sucht nun die Zeile mit

```
Testi:xyzverschlüsseltes.passwort:2002:501:::/dev/null
```

Der rot markierte Bereich muss nun so abgeändert werden, dass Ihr aus dem **Testi** einen echten root-Account macht. Also schreiben wir vom root-Account den hinteren Teil einfach ab.D.h. die Zeile mit dem Testi muss nun so aussehen:

```
Testi:xyzverschlüsseltes.passwort:0:0:root:/root:/bin/sh
```

6. Mit "ESC" den Editmodus verlassen. mit ":w" speichern und ":q" beenden.

7. Zum Schluß updaten wir die Slug gleich und sehen dabei, ob unser neuer User wirklich Rootrechte hat. Hierzu folgendes ausführen:

```
ipkg update  
ipkg list  
ipkg upgrade
```

Danach werden automatisch die neuesten Pakete gezogen und installiert.

Zudem sollten noch folgende Einstellungen im Webadmin gemacht werden:

Ruft dazu Eure Konfiguration auf und geht auf **Administration - > System**. Dann entfernt die Haken vor

- > Enable Guest Logins
- > Enable FTP-Server
- > Enable UPnP Support

Wichtig ist, dass Ihr noch den Port des Webadmins von 80 auf einen anderen ändert (8282 oder ähnliches). Den Port 80 benötigt Ihr dann für Euren Apache.

Die Konfiguration sollte dann in etwa wie folgt aussehen:

unslung  Firmware Version: V2.3R63

System Network Storage Link for USB 2.0 Disks NSLU2

Home User password Administration User Guide Linksys Web

LAN | System | Users | Status | Advanced

Identification

Server Name:

Comment:

Workgroup:

Location

Language Support:

Time Zone:

Local Date: , 2007 (mth, day, year)

Local Time: :

WINS

Enable WINS

WINS Server: . . .

Location

Enable Guest Logins

Convert failed logins to "guest" logins (Windows networks)

Enable FTP Server

Allow anonymous FTP login ("guest" rights)

Enable UPnP Support

Port number for HTTP (Web Browser) connections to this server:

Password for 'guest' account that is used during no EXT3 disk available:

Save Cancel Restore Default Config Help

CISCO SYSTEMS 

Hinweis: *Ihr müsst nicht jedes mal den Webadmin bemühen, um die Slug neu zu starten. Ebenso ist es nicht notwendig, jedes Mal aufzustehen, und das Knöpfchen direkt am Gerät zu drücken. Wenn Ihr eh schon in der Konsole eingeloggt seid, dann geht das über zwei lockere Befehle am Prompt:*

```
sync  
reboot
```

bash installieren

Ziel ist es, die bash soweit zu installieren, dass jeder User entsprechend die gleiche bash verwendet. Ich beziehe mich dabei auf die Anleitung von unslung.de, passe diese allerdings hie und da einwenig an, um vielleicht das eine oder andere einwenig chronologischer in der Beschreibung darzustellen.

1) Alles passiert natürlich unter telnet. Loggt Euch mit Euerem root-Account ein, und installiert zuerst die bash:

```
ipkg install bash
```

2) Falls noch nicht geschehen legen wir unter etc eine Datei profile an und editieren diese gleich:

```
vi /etc/profile
```

(Hinweis zum vi: Editmodus: Taste "i" drücken. Editmodus verlassen: "ESC" drücken. Speichern: Tasten ":w" und beenden mit Tasten ":q")

3) Nun fügen wir folgende zwei Zeilen in die Datei ein:

```
export PS1="\[\033[1;33m\]\u@\h\[\033[0;37m\]:/> "
```

und GANZ ZUM SCHLUSS

```
test -f /opt/bin/bash && exec /opt/bin/bash
```

Wer es etwas freundlicher beim Begrüßen mag, der kann oben in der **etc/profile** noch folgende Zeilen ergänzen:

```
echo -----  
echo "Hallo $USER, willkommen auf der Slug! ;-)"  
echo -----  
echo
```

4) Als nächstes legen wir im root-Verzeichnis die **.bashrc** an:

`vi /.bashrc`

In diese Datei können wir einige hilfreiche Aliase und Umgebungsvariablen setzen. Ich empfehle folgende Zeilen einzugeben:

```
alias "d=ls -la"  
alias "dir=ls -la"  
export PATH=$PATH:~/bin:.
```

5) Die **.bashrc** muss nun in jedes Userverzeichnis kopiert werden, das sich unter **/share/hdd/data/** befindet.

Jetzt macht eine neue Telnet-Session auf und meldet Euch an. Nun dürfte alles etwas vertrauter aussehen, zumindest für die eingefleischte Linux Fans.

Midnight Commander installieren

1) Zuerst muss das Programm installiert werden:

```
ipkg install mc
```

(falls vorher noch nicht geschehen, dann zuerst ipkg update eingeben!)

2) Falls noch nicht geschehen, die Datei /etc/profile anlegen und editieren:

```
vi /etc/profile
```

3) Dort zwei Zeilen ergänzen:

```
export TERMINFO=/opt/lib/terminfo
export TERM=xterm-pcolor
```

4) System neu starten.

5) Midnight Commander mit

```
/opt/bin/mc
```

starten. Falls Ihr die bash installiert habt, könnt Ihr auch in der .bashrc folgendes ergänzen:

```
alias "mc=/opt/bin/mc"
```

Das macht die Sache recht praktisch, denn so wird das Programm ganz einfach mit einem **mc** in der shell gestartet.

FTP-Server einrichten (vsftpd)

1) Zuerst legt man ein leeres Verzeichnis an:

```
mkdir -p /usr/share/empty
```

2) Vsftp-Daemon installieren:

```
ipkg install vsftpd
```

3) unslung- Verzeichnis anlegen:

```
mkdir -p /share/hdd/conf/unslung
```

4) rc.xinetd Datei anlegen und mit Inhalt füllen:

```
vi /share/hdd/conf/unslung/rc.xinetd
```

In den Firmwareversionen über 5.x ist das rc.xinetd Script im Verzeichnis /unslung direkt unter Root abzulegen. Somit entfällt Schritt 3) und 4) und es genügt folgender Befehl:

```
vi /unslung/rc.xinetd
```

Dann im vi-Editor den Editmodus mit Drücken der Taste "i" aktivieren und folgendes abtippen:

```
#!/bin/sh

if ( [ ! -f /etc/inetd.conf ] || !(grep vsftpd /etc/inetd.conf -q) ) then
  echo "ftp stream tcp nowait root /opt/sbin/vsftpd /opt/etc/vsftpd.conf"
  >>/etc/inetd.conf
fi

return 1
```

Bitte HAARKLEIN GENAU abschreiben! Bitte zweimal kontrollieren!
Dann Editit-Modus mit "ESC" beenden und mit

:w

abspeichern und mit

:q

beenden

5) vsftpd.conf editieren, damit jeder angemeldete User nur seinen eigenen Bereich sehen kann, und nicht in Toplevel wechseln kann:

```
vi /opt/etc/vsftpd.conf
```

Hier dann im Edit-Modus (Taste "i" !) folgende Zeile am Schluss ergänzen:

```
chroot_local_user=YES
```

Abspeichern und beenden!

6) Logdatei für den ftp-Daemon erstellen:

```
mkdir -p /opt/var  
mkdir -p /opt/var/log  
touch /opt/var/log/vsftpd.log
```

7) Ersten User anlegen:

Im normalen Adminpanel einen neuen User anlegen. Nennen wir ihn hier als Beispiel "upload". Dieser bekommt auch beim Erstellen ein Homedirectory auf der Platte zugesprochen. Jetzt müssen wir noch die **passwd** anpassen:

```
vi /etc/passwd
```

Den User "upload" suchen. Die Zeile sieht dann in etwa so aus:

```
upload:xyz.verschlüsseltespasswort:2003:501:::/dev/null
```

Das ändern wir dann in folgender Weise ab:

```
upload:xyz.verschlüsseltespasswort:2003:501::/share/hdd/data/upload:/bin/sh
```

In der Version **6.8** bitte folgende Verzeichnisse verwenden:

```
upload:xyz.verschlüsseltespasswort:2003:501::/upload:/bin/sh
```

Da auf der Slug nun alles auf dem USB-Port2 liegt, wäre das korrekte Verzeichnis /share/flash ... Wir umgehen das damit, indem wir den die Verlinkung unter /root verwenden.

Somit wird dem User ein festes Heimatverzeichnis zugewiesen, das sich eben auf der Platte befindet. Das alles wie oben schon beschrieben dann speichern und beenden.

8) Die NSLU dann neu starten und mit einem ftp-Clients oder via Webbrowser

```
ftp://username:deinpasswort@ipnummerderslu
```

ausprobieren.

Noch ein Tipp: Wenn man ein laufendes Protokoll sich anzeigen lassen möchte, dann macht eine Telnet-Session auf und gebt folgendes ein:

```
tail -f /opt/var/log/vsftpd.log
```

Dann habt Ihr immer im Blick, was gerade auf dem Server passiert.

Hinweis :

Solltet Ihr einen Tippfehler bei der Erstellung der **/share/hdd/conf/unslung/rc.xinetd** gebaut haben und startet das Dingens via Neustart der Kiste oder mit dem Befehl **/etc/rc.d/rc.xinetd**, dann kann es sein, dass Ihr eben keinen Zugriff via ftp bekommt. Ihr entdeckt dann zwar den Fehler im **/share/hdd/conf/unslung/rc.xinetd** aber auch nach Korrektur startet der FTP-Server nicht. Das liegt nun daran, dass beim ersten Start die Zeile echo"ftp... " in das **/etc/rc.d/rc.xinetd** Script MIT ÜBERNOMMEN wurde. Also hier dann den Fehler, sollte er in der echo-Zeile entstanden sein, auch ausbessern! Dann klappt es! Die Festplatte zu löschen und neu zu unslingen bringt auch nichts, da das **etc/rc.d/rc.xinetd** auf dem Flashspeicher liegt. Hier müsste man die Firmware dann komplett neu aufspielen. Aber da lässt sich sicher besagte Datei einfacher und schneller editieren.

Indianer! Der Apache 2.0 Webserver

Der Apache Webserver ist auf der Slug schnell installiert. Hierzu wird sich erstmal als root per telnet eingeloggt.

Danach führen wir bis zur Version 6.8 den Befehl

```
ipkg install unslung-feeds
```

aus. Und nun folgenden Befehl:

```
ipkg install apache
```

Bei der Firmware ab Version **6.8** reicht letzterer Befehl vollkommen aus. Das System wird automatisch konfiguriert und der Webserver gestartet. Dieser liegt allerdings auf dem Port **8000**. Ruft den Server nun mit

http://[eure ip adresse]:8000

auf. Es ergibt dann in etwa folgendes Bild:



Um den Apache nun anzupassen rufen wir die Konfigurationsdatei des Apache auf:

```
vi /opt/etc/apache2/httpd.conf
```

Wichtige Zeile/ Einträge:

Listen 8000

gibt den Standardport an. Ändert dies auf **80**, um den normalen Port eines Webservers zu verwenden.

DocumentRoot "/opt/share/www" legt fest, wo Eure Webfiles liegen. Idealerweise legt hier einen User an, den Ihr per ftp erreichen könnt. Auf dieses Userverzeichnis schreibt diese Zeile um. Z.b. nennt sich der User Webserver . Also lautet die Zeile dann:

```
DocumentRoot "/webserver"
```

<Directory "/opt/share/www"> muss ebenso angepasst werden (z.B. <Directory "/webserver">

DirectoryIndex index.html index.html.var Diese Zeile ergänzt dann noch durch ein "index.htm und index.php" , damit auch Pages mit der htm und php Extention geladen werden können.

Das sollten die wichtigsten Punkte sein. Im Internet finden sich noch eine Vielzahl an weiteren Konfigurationsmöglichkeiten für den Apache. Speichert Euerer Änderungen ab.

Kopiert nun eine Testpage auf Euer Webverzeichnis und startet die Slug neu. Nun müsste nach einer kurzen Wartezeit der Webserver mit Euerer normalen IP abrufbar sein.

***Hinweis:** Bei Eurem Webserver bitte die Files immer via ftp und nicht dem Samba-Share hoch laden. Denn nur über einem ftp-Upload ist sichergestellt, dass Ihr keinen Access-Error wegen mangelnder Zugriffsrechte bekommt.*

PHP Erweiterung für Apache

Das geht schneller, als wir vermuten. Zuerst loggen wir uns wieder als root via einer Telnetsession ein. Danach installieren wir zwei Pakete:

```
ipkg install php-apache eaccelerator
```

Jetzt starten wir den Apache neu, indem wir folgenden Befehl absetzen:

```
/opt/etc/init.d/S80apache
```

Jetzt legen wir folgenden Code auf unsere Homepage:

```
<?php
  phpinfo()
?>
```

Das File speichern wir als "phpinfo.php" ab. Dieses rufen wir dann über unseren Browser auf:

[deine ip nummer]/phpinfo.php

Wenn alles funktioniert sollte dies in der Art zu sehen sein:



System	Linux Hellboy 2.4.22-xfs #1 Mon Apr 10 18:17:11 PDT 2006 armv5b
Build Date	Apr 16 2006 21:58:09
Configure Command	'./configure' '--build=i386-pc-linux-gnu' '--host=armv5b-softfloat-linux' '--target=armv5b-softfloat-linux' '--prefix=/opt' '--with-config-file-scan-dir=/opt/etc/php.d' '--with-layout=GNU' '--disable-static' '--enable-maintainer-zts' '--disable-dom' '--disable-xml' '--enable-libxml' '--with-apxs2=/home/slug/optware/nslu2/staging/opt/sbin/apxs' '--without-pear' '--without-iconv'
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	/opt/etc/php.ini
Scan this dir for additional .ini files	/opt/etc/php.d
additional .ini files parsed	/opt/etc/php.d/eaccelerator.ini
PHP API	20031224
PHP Extension	20041030
Zend Extension	220040412
Debug Build	no
Thread Safety	enabled
IPv6 Support	enabled

Dieses Script gibt nun Auskunft, welche Version des Apache und des PHP installiert ist.

Hinweis! Sollte der php-Aufruf nur einen Plain-Text bringen, dann kann es sein das in einer möglichen neuern Apache Version der php-Aufruf in der `/opt/etc/apache2/httpd.conf` fehlt. Ergänzt hierfür am Ende der Datei folgende Zeilen:

```
LoadModule php5_module libexec/libphp5.so
AddType application/x-httpd-php .php .html
```

Mysql Datenbank installieren

Mysql selbst installieren wir über ein Konsolenfenster als root. Hierzu gebt folgenden Installationsbefehl ein:

```
ipkg install php-mysql
```

Nachdem die Pakete installiert sind setzen wir sofort ein root- Passwort für die Datenbank:

```
/opt/bin/mysqladmin -u root password dein_passwort
```

HINWEIS! Bei neueren Versionen kann es beim Aufruf zu einer Meldung kommen, dass die Library libmysqlclient.so.14 nicht gefunden wird. Ist anscheinend bei neueren Installationen der Fall. Die Meldung umschifft Ihr mit einem Symlink:

```
ln -s /opt/lib/mysql/libmysqlclient.so.14.0.0 /lib/libmysqlclient.so.14
```

(sicherheitshalber die Slug neu booten!)

Jetzt legen wir eine neue Datenbank mit den Namen *Beispiel* an:

```
/opt/bin/mysqladmin -p create beispiel
```

Hier muss nun Euer Passwort eingegeben werden

Das wären die ersten wirklichen Fingerübungen am Prompt, was die Datenbank betrifft. Damit der Webserver die Datenbank auch sauber integriert startet die Slug einfach neu:

sync
reboot

Ruft nun nochmal Euer phpinfo-Script auf, wie ich es im PHP-Abschnitt beschrieben habe. Im Bereich "**additional .ini files parsed**" muss jetzt die mysql.ini stehen:

additional .ini files parsed	/opt/etc/php.d/eaccelerator.ini, /opt/etc/php.d/mysql.ini
-------------------------------------	---

Wer jetzt natürlich nicht unbedingt sich die Finger beim Eingeben von cryptischen Befehlszeilen am Prompt brechen möchte, wenn er seine Datenbank administriert, der braucht auch nicht auf seiner kleinen Slug auf die Vorzüge der grafischen Administration via **phpmyadmin** zu verzichten. Das Programmpaket kann man sich unter

<http://www.phpmyadmin.net>

herunterladen. Das entpackt man dann auf seinen Webserver und legt wie gehabt seine **config.inc.php** an. Bitte konsultiert die beiliegende Doku, um den genauen Installationsvorgang in Erfahrung zu bringen. Weiteres würde über den mir gesteckten Rahmen dieser Dokumentation hinaus gehen. Wenn Ihr die Dateien oben habt und die Zugriffe in der config geregelt habt, dann solltet Ihr das phpmyadmin wie gewohnt benutzen können:

The screenshot shows the phpMyAdmin interface. On the left sidebar, there's the phpMyAdmin logo and a 'Datenbank' dropdown menu set to 'beispiel (0)'. Below that, it says 'beispiel (0)' and 'Es wurden keine Tabellen in der Datenbank gefunden.' The main content area shows 'localhost ► beispiel' and a toolbar with icons for 'Struktur', 'SQL', 'Suche', 'Abfrageeditor', 'Exportieren', 'Importieren', 'Operationen', 'Rechte', and 'Löschen'. A message states 'Es wurden keine Tabellen in der Datenbank gefunden.' Below this is a form to create a new table: 'Neue Tabelle in Datenbank beispiel erstellen'. It has input fields for 'Name:' and 'Anzahl der Felder:', and an 'OK' button. At the bottom right, there's a link 'Neues phpMyAdmin Fenster'.

Hinweis: Hier ist es ratsam, über ein htaccess File das Verzeichnis zu schützen, auf dem das phpmyadmin abgelegt ist. Sonst kann ja jeder in der Datenbank sein Unwesen treiben.

Exkurs: htaccses anlegen

Das ist einwenig kniffliger, aber ich denke, dass man das, wenn man sich Schritt für Schritt an meine Anleitung hält machbar. Zuerst müssen wir dem Apache erklären, dass er nicht nur Friedenspfeifen rauchen soll, sondern auch gesicherte Verzeichnisse zu verwalten hat. Dazu müssen wir seine Konfigurationsdatei "nachbehandeln". In einer telnet-Session rufen wir diese als root im vi-Editor auf:

```
vi /opt/etc/apache2/httpd.conf
```

Die Zeile

```
AllowOverride None
```

muss auf

```
AllowOverride All
```

(i oder **Einfg.** zum Editieren drücken, **ESC** zum Verlassen des Editmodus)

gesetzt werden. Dann mit **:w** speichern und mit **:q** verlassen.

Jetzt startet den Apache wieder neu, damit die neue config eingelesen wird:

```
/opt/etc/init.d/S80apache
```

Somit sind wir entsprechend vorbereitet, die gewünschten Verzeichnisse zu sichern. Wir wollen nun das Verzeichnis des Webservers /phpmyadmin absichern. Dieses liegt auf der Slug Beispielsweise unter /webserver/phpmyadmin . Legt nun eine Datei .htaccess mittels vi an

```
vi /webserver/phpmyadmin/.htaccess
```

und füllt diese mit folgenden Zeilen:

```
AuthType Basic
AuthName Phpmyadmin
AuthUserFile /webserver/phpmyadmin/.htpasswd
require valid-user
```

Hinter "**AuthName**" könnt Ihr eine beliebige Bezeichnung setzen. Diese wird dann nur eben später am Eingabefenster des Browsers angezeigt, wenn nach dem Passwort gefragt wird. Man kann natürlich noch einige Erweiterungen in die htaccess mit aufnehmen, allerdings sollte unsere Konfiguration für das Erste genügen.

Dann legen wir die Passwortdatei .htpasswd an. Dazu liefert uns der Apache ein Tool, das uns auch die Passwörter entsprechend verschlüsselt.

```
htpasswd -C .htpasswd username
```

Gebt nun das Passwort ein. Einen weiteren User legt Ihr mit

```
htpasswd .htpasswd neueruser
```

an. Macht auch diese Datei lesbar:

```
chmod 644 .htpasswd
```

Sollte der Aufruf htpasswd nicht funktionieren, so habt Ihr in der **etc/profile** den entsprechenden Suchpfad nicht eingetragen. Ihr könnt Euch auch kurzfristig damit behelfen:

```
export PATH=$PATH:/opt/sbin:.
```

Ansonsten solltet Ihr eh folgende Zeilen in die **/etc/profiles** aufnehmen:

```
export PATH=$PATH:/opt/bin:/opt/sbin:/opt/usr/bin:/opt/usr/sbin:
export LD_LIBRARY_PATH=/opt/lib
```

Es funktionieren dann auch komplexere Anwendungen, wie z.B. das bekannte **phpbb**-Forum. Den Download findet Ihr hier:

<http://www.phpbb.com/>

Hinweise zur Installation entnehmt bitte der beiliegenden Dokumentation.

Testinstallation:

yourdomain.com
A _little_ text to describe your forum

FAQ Search Memberlist Usergroups Register
Profile Log in to check your private messages Log in

The time now is Tue Apr 18, 2006 9:58 pm [View unanswered posts](#)

yourdomain.com Forum Index

Forum	Topics	Posts	Last Post
Test category 1			
Test Forum 1 This is just a test forum.	1	1	Sat Oct 21, 2000 12:01 am admin →

All times are GMT

Who is Online

Our users have posted a total of **1** article
We have **1** registered user
The newest registered user is [admin](#)

In total there is **1** user online :: 0 Registered, 0 Hidden and 1 Guest [Administrator] [Moderator]
Most users ever online was **1** on Tue Apr 18, 2006 9:58 pm
Registered Users: None

This data is based on users active over the past five minutes

Log in

Username: Password: Log me on automatically each visit

New posts No new posts Forum is locked

Zeitgesteuerte Events - Die Crontab

Ihr könnt natürlich wie unter jedem Linux-System zeitgesteuerte Events, wie z.B. Sicherungen über die Crontab steuern. Im Internet findet Ihr eine Vielzahl an Seiten, die den genauen Aufbau der Crontab beschreiben. Da wir einen Webserver haben und mit einer Datenbank arbeiten, macht es Sinn, die Datenbank regelmäßig auf ein extra Verzeichnis zu speichern. Wir gehen in unserem Beispiel davon aus, dass wir die Datenbank "beispiel" alle 30 Minuten speichern wollen.

Zuerst legen wir ein Script an, das unsere Datenbank wegsichert. Idealerweise habt Ihr einen weiteren user (z.B. test) angelegt, auf den Ihr ftp-Zugriff habt. In dessen Verzeichnis werden wir die Datenbank "dumpen".

Die Datei legen wir wieder mit dem guten alten vi an:

```
vi /usr/sbin/sqldump.sh
```

Der Code sieht im Groben wie folgt aus:

```
#!/bin/sh
/opt/bin/mysqldump --password=(passwort) datenbankname >
/zielverzeichnis/datenbankname.sql --default-character-set=utf8
(bitte unter #!/bin/sh alles in eine Zeile schreiben!)
```

In unserem Beispiel also:

```
#!/bin/sh
/opt/bin/mysqldump --password=(passwort) beispiel > /test/beispiel.sql --default-
character-set=utf8
(bitte unter #!/bin/sh alles in eine Zeile schreiben!)
```

Als Passwort (dann ohne die Klammern!!!) gebt Ihr Euer admin-Passwort ein, das Ihr bei der mysql-Installation festgelegt habt.

Speichert die Datei ab. Jetzt müssen wir die Datei noch ausführbar machen. Das funktioniert über ein lockeres

```
chmod +x /usr/sbin/sqldump.sh
```

Testet das Script einfach, indem Ihr das Datenbank-Dumping per Hand startet:

```
/usr/sbin/sqldump.sh
```

Nun editieren wir die Crontab. Das machen wir alte Linux-Hasen wieder über den vi:

```
vi /etc/crontab
```

fügt ans Ende der Datei folgende Zeile ein:

```
30 * * * * root /usr/sbin/sqldump.sh &>/dev/null
```

Die Crontab sollte nun ungefähr so aussehen:

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=""
HOME=/
# ----- Default is Empty ----- #
0 0-23/8 * * * root /usr/sbin/CheckDiskFull &>/dev/null
0 0 * * * root /usr/sbin/WatchDog &>/dev/null
1 * * * * root /usr/sbin/hwclock -s &>/dev/null
30 * * * * root /usr/sbin/sqldump.sh &>/dev/null
```

Speichert Euere Änderungen ab. Danach startet die Slug neu, damit die Crontab auch wieder neu eingelesen wird:

```
sync
reboot
```

Prüft dann bei Gelegenheit nach, ob Ihr tatsächlich immer neue Sicherungen in Euer Sicherungsverzeichnis "gedumpt" bekommt.

Wanzenjagd mit Chkrootkit

Wer seine kleine Schachtel ständig am Netz hängen lässt ist vor Angreifern natürlich nicht sicher. Deswegen ist Umsichtigkeit sowohl in den Logs als auch bei den systemkritischen Dateien erstes Gebot! Die Linuxwelt spricht weniger von Viren, sondern bekannter sind Rootkits, die wichtige Systemdateien verändern damit der Angreifer Kontrolle über das System erlangt.

Aber auch hierfür gibt es Tools, um zu prüfen, wie der Stand der Dinge ist. Ein regelmäßiger Check seines Servers ist auf jeden Fall erstes Gebot.

In den Repositories der NSLU2 gibt es leider hierzu kein entsprechendes Tool, also werden wir selbst Hand anlegen. Wir werden dazu das bekannte Werkzeug **chkrootkit** verwenden. Die dazugehörige Page findet Ihr hier:

<http://www.chkrootkit.org>

Damit auf der Slug überhaupt etwas kompiliert werden kann müssen die Build-Tools installiert werden. Diese benötigen ca. 150 MB auf der Festplatte.

```
ipkg install optware-devel
```

Danach legen wir uns ein beliebiges Arbeitsverzeichnis an:

```
mkdir work
```

und wechseln in dieses rein:

```
cd work
```

Jetzt holen wir uns das Quellpaket ab:

```
wget ftp://ftp.pangeia.com.br/pub/seg/pac/chkrootkit.tar.gz
```

Das Paket entpacken wir mit einem lässigen

```
tar -zxvf chkrootkit.tar.gz
```

Der Quellcode liegt nun in einem neuen Verzeichnis. Wie dieses heisst erfahrt Ihr mit einem

```
ls -l
4096 Feb  2 14:20 chkrootkit-0.47
37791 Feb  2 13:59 chkrootkit.tar.gz
```

Hier sieht man, dass das Verzeichnis **chkrootkit-0.47** heißt. Wechselt nun in dieses Verzeichnis:

```
cd chkrootkit-0.47
```

Danach müssen wir noch einen Suchpfad setzen, damit das mit dem Compiler klappt:

```
export PATH=$PATH:/opt/bin:.
```

Jetzt legen wir mit einem

```
make
```

los. Der Vorgang dauert nicht lange. Damit das Programm auch arbeitet brauchen wir noch das netstat Tool:

```
ipkg install net-tools
```

Jetzt können wir das Tool aus dem gleichen Verzeichnis heraus mit

```
./chkrootkit
```

starten. Der Output ist dann denke ich selbsterklärend.

Ich weiÙe darauf hin, dass bei eventuell gefundenen Infektionen erstmal geschaut werden muss, welche Datei davon betroffen ist. Da die Software auf der NSLU2 angepasst ist muss das nicht zwangsläufig bedeuten, dass tatsächlich eine Infektion vorliegt, sondern die eine oder andere Datei von der Suchroutine als infiziert markiert wird, weil diese vielleicht vom Standardcode anderer Linuxversionen abweicht (Falschmeldung). Sollte sich aber im Laufe der Zeit zusätzlich etwas verändern, dann ist es **HÖCHSTE** Eisenbahn, sein System vom Netz zu nehmen und auf die Suche zu gehen.

Zusätzlich: Nehmt die Quellcodes wieder von der Platte, damit hier nichts manipuliert werden kann. Das kompilierte Programm könnt Ihr in ein beliebiges Verzeichnis verschieben.

Festplatten in den Ruhemodus versetzen (spin-down)

Nachdem Klimaschutz und Energiesparen in aller Munde ist, sollten wir uns auch Gedanken um den Stromverbrauch unseres Mini-Servers machen. Unsere externe Platte soll nun nach einer gewissen Zeit in den Ruhemodus versetzt werden und herunterfahren (**spin-down**). Wenn die Maschine wieder einen Zugriff von außen bekommt soll die Platte wieder hochfahren (**spin-up**).

Nur ist es leider nicht so einfach, den Stromsparmomodus zu aktivieren und es existieren sehr viele Ansätze, die teilweise funktionieren, aber oft viel Konfigurationsarbeit voraussetzen.

Wenn man sich durch diverse Anleitungen arbeitet, gibt es im Grunde drei Wege, die zum gewünschten Ziel führen:

- a) Die Harddisk beherrscht das Herunterfahren von sich aus
- b) Wir geben das Timeout der Platte von außen mit
- c) Die NSLU2 wird softwareseitig über Scripte gesteuert

Ich möchte auf das alles nicht wirklich recht tief eingehen, denn im Netz finden sich entsprechende Anleitung. Ich möchte hier nur meinen Ansatz vorstellen, der auf der 2. Basis (Timeout von außen) basiert und recht einfach umzusetzen ist, wenn die Randbedingungen stimmen.

a) Die Harddisk beherrscht das Herunterfahren von sich aus

Mache Festplatten oder fertige USB Lösungen haben bereits ein vorprogrammiertes Powermanagement. D.h. sie drehen nach einer gewissen Zeit der Inaktivität herunter oder schalten sich ab. Empfohlen hierfür wird gerne die USB-Box "[One Touch](#)" von Maxtor. Zudem bieten manche Hersteller Tools zum Einstellen ihrer Festplatten an, wie Hitachi mit ihrem "[Feature Tool](#)". Diese Lösung hat allerdings leider bei mir auf zwei verschiedenen IBM/Hitachi nicht funktioniert.

b) Wir geben das Timeout der Platte von außen mit

Unter **Linux** bietet sich das Tool `hdparm` zum Regulieren des Spin-Downs an. Hierzu kann man die USB Box öffnen und steckt die Disk an einem IDE-Kabel direkt an seinem Rechner an und schaltet die Box an. Danach bootet man Linux und holt sich die Rootrechte an einer Konsole:

```
su  
[PASSWORT EINGEBEN]
```

Dann setzen wir den Spin-Down fest:

```
hdparm -k1 -K1 -S120 /dev/hda
```

Für `/dev/hda` nehmen wir entsprechend unserer angeschlossenen Platte das korrekte Device:

Primary Master: **hda**
Primary Slave: **hdb**
Secondary Master: **hdc**
Secondary Slave: **hdd**

`-S120` legt die Zeit auf 10 Minuten fest. Zum Testen kann man den Wert auch auf 10 Sekunden (`-S10`) festlegen. Die Parameter `-k1` und `-K1` versuchen den Wert in die Festplatte dauerhaft zu schreiben, damit nachdem man die Kiste vom Stromnetz genommen hat sich auch noch an den Spin-Down Timer erinnert. Das klappt aber leider nur in den wenigsten Fällen. Von daher muss die Platte im laufenden Betrieb vom IDE Kabel gezogen werden und darf nicht abgeschaltet werden. Dann wird der interne IDE Anschluss der USB Box wieder angeschlossen und an die NSLU2 gestöpselt. Schaltet man die Festplatte ab, so muss dieser Vorgang wiederholt werden. Das ist nicht wirklich komfortabel, aber bringt fast immer den gewünschten Erfolg.

Da man eigentlich keine IDE-Platte vom laufenden Rechner trennen sollte (der Controller könnte einem das übel nehmen) empfehle ich eine andere Variante:

Hierzu benötigt man eine SATA Platte und ein USB Gehäuse für SATA Platten, das sowohl einen SATA Anschluss als auch USB Anschluss mitbringt. Da SATA hotplugfähig ist, kann das Kabel relativ gefahrlos gezogen werden. Ich habe an meinen zweiten SATA Controller meines Mainboards das der USB Box beiliegende Slotblech (SATA Bracket) für einen externen SATA Anschluss angeschlossen. So kann ich die SATA Platte nun via meines SATA Controllers erstmal an meinen Rechner bringen, ohne dass ich das USB Gehäuse aufschrauben und irgendwelche Kabel trennen muss. Ich stecke einfach die Platte im laufenden Linuxbetrieb an und checke erstmal, ob diese erkannt

wurde, indem ich folgenden Befehl an der Konsole absetze:

```
dmesg
```

Hier sehe ich auch gleich, als welches Gerät (Device) die Platte erkannt wurde. Dies wird in der Regel /dev/sd... sein. Nun kann ich mittels hdparm den gewünschten Spin-down Timer einstellen:

```
hdparm -k1 -K1 -S120 /dev/sd...
```

Dann ziehe ich einfach den SATA Anschluss ab und stecke den USB-Anschluss wieder an die NSLU2. Die Platte merkt sich nun solange dieses Setting bis das Gerät vom Strom getrennt wird. Damit das natürlich einfach und schnell zu bewerkstelligen ist sollte die NSLU2 nebst USB Platte in der Nähe des stationären Rechners stehen. Das empfiehlt sich auch deswegen, da man auch schnell mal die Harddisk über die SATA Schnittstelle mit größeren Datenpaketen füttern kann. Ansonsten bestünde noch die Möglichkeit dies mittels eines Notebooks zu erledigen, das einen externen SATA Anschluss hat (fertig eingebaut oder via PCMCIA Controller).

Hier mein Setup mit einem SATA2 USB-Gehäuse, das beide Schnittstellen bereit hält:

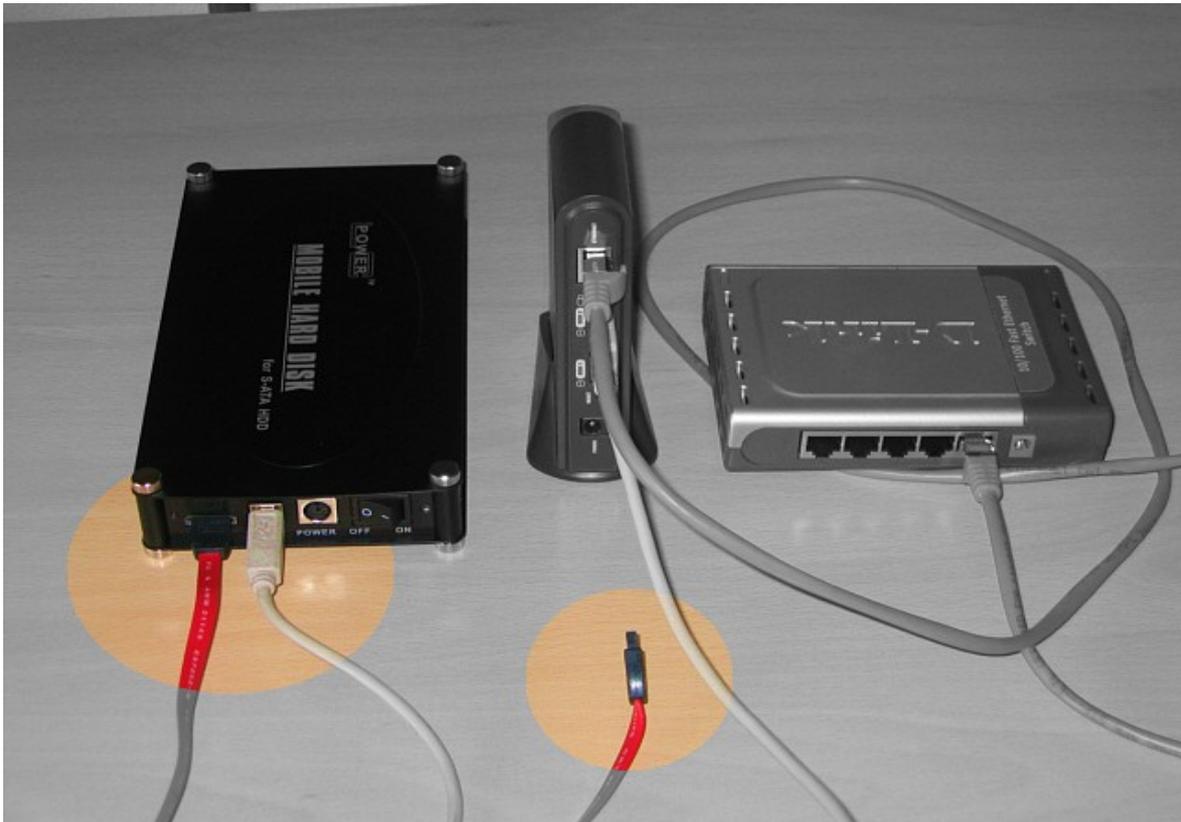


Abb. 3: NSLU2 mit externer Festplatte und SATA Anschluss

c) Die NSLU2 wird softwareseitig über Scripte gesteuert

Diverse Scripte, die die Box intern steuern können findet Ihr unter folgendem Link:

<http://www.nslu2-linux.org/wiki/FAQ/SpinDownUSBHarddisks>

Hier sei nur gesagt, dass das Setup teilweise nicht trivial ist und einiges an Experimentierfreude voraussetzt.

Hinweis: Der Spin-Down der Platte funktioniert natürlich erst dann, sobald kein Zugriff auf die Platte erfolgt. Hier muss dafür gesorgt werden, dass etwaige Prozesse nicht auf die Platte greifen. Zu überlegen ist hierbei, das OS auf einen USB-Stick zu unslingen und die Platte nur als Datenträger der Homeverzeichnisse zu verwenden.

CURL zum Download verwenden

Unsere kleine Schnecke eignet sich natürlich auch wunderbar dafür, Daten aus dem Netz zu saugen, auch wenn wir mal nicht an einem Rechner sitzen. Große Files werden dann einfach über nacht von der kleinen Box selbständig gezogen und das verbraucht natürlich deutlich weniger Strom, als wenn wir das an unserer big Machine erledigen.

Gerade ist beispielsweise das neue Debian Etch 4.0 erschienen. Diese Distribution kann man entweder auf über 20 CDs aus dem Netz saugen oder auch wahlweise als eine DVD Ausgabe, die sich über satte drei Scheiben erstreckt und schonmal ein Downloadvolumen von rund 12GB hat.

Ein einfaches wget stößt dann schon bei 2GB an seine Grenzen. Dann muss einfach etwas mächtigeres her. Hierfür bietet sich das Kommandozeilen-Tool Curl an, das auch mehrere Jobs hintereinander erledigen kann. Die Projektseite dazu findet sich hier:

<http://curl.haxx.se>

Leider gibt es für die Slug selbst keine fertige Version, sodass wir hier selbst Hand anlegen müssen. Aber auch dies soll uns nicht davor abschrecken, das Tool einzusetzen.

Als erstes müssen wir die Build-Tools der Slug installieren (falls noch nicht geschehen):

```
ipkg install optware-devel
```

Zuerst modifizieren wir die `/etc/profile`, damit wir einen sauberen Kompile hinbekommen:

```
vi /etc/profile
```

Ergänzt dann folgende Zeilen und speichert die Datei ab:

```
export PATH=$PATH:/opt/bin:/opt/sbin:/opt/usr/bin:/opt/usr/sbin:.  
export LD_LIBRARY_PATH=/opt/lib
```

Meldet Euch erstmal aus Eurer Slug ab und dann wieder an, damit die Suchpfade übernommen werden.

Jetzt legen wir ein Arbeitsverzeichnis an:

```
mkdir work
```

und wechseln gleich dort hin:

```
cd work
```

Nun saugen wir uns den Quellcode (noch mit wget):

```
wget http://curl.haxx.se/download/curl-7.16.2.tar.gz
```

Nun packen wir den Code aus:

```
tar -zxvf curl-7.16.2.tar.gz
```

Daraufhin entsteht ein neues Verzeichnis in das wir gleich wechseln:

```
cd curl-7.16.2
```

Und schon können wir loslegen:

```
./configure --prefix=/opt
```

Das bereitet uns den Quellcode vor und stellt sicher, dass alle Files in den Standardpfad für neue Anwendungen der NSLU2 landen. Da unsere Schnecke nicht die schnellste ist, kann das durchaus eine Weile dauern, bis das Konfigurationsscript durch ist.

Der Kompile wird dann mit einem

```
make
```

angestoßen. Auch das wird sich einwenig hinziehen. Danach installiert Ihr alles mit einem

```
make install
```

Nun können wir uns als root abmelden und melden uns als ein normaler User wieder an. Danach starten wir einen Testdownload, der beispielsweise so aussehen kann:

```
curl -C - -O http://laotzu.acc.umu.se/debian-cd/4.0_r0/i386/iso-cd/debian-40r0-i386-netinst.iso
```

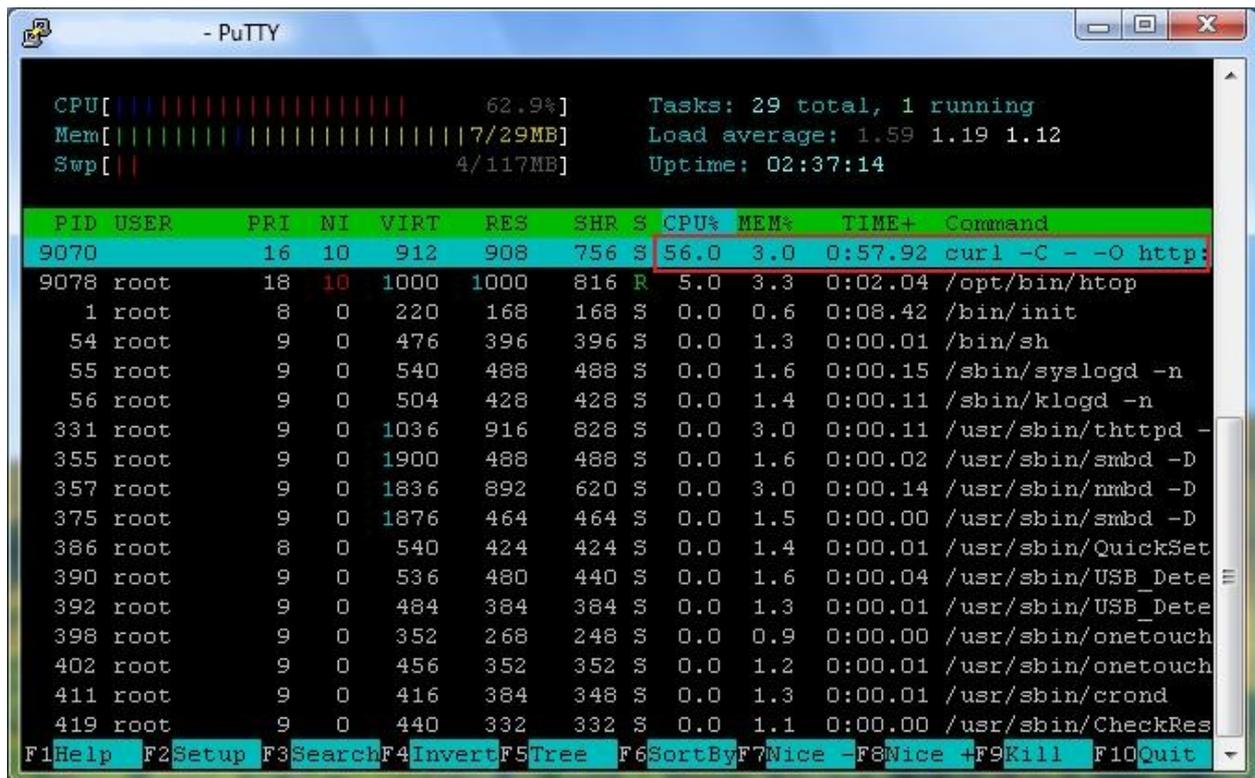
Weitere Links werden einfach mit einem weiteren **-O** angehängt, dann arbeitet Curl die Downloads der Reihe nach ab.

Jetzt haben wir natürlich noch das Problem, dass beim Ausloggen aus der telnet- oder ssh- Session der curl-Prozess wieder abbricht. Um das zu verhindern, müssen wir den Prozess abkoppeln. Das erledigen wir einfach, indem wir den Befehl **nohup** (no hangup) vor unserem curl - Befehl stellen. Allerdings schreibt uns der nohup Befehl den kompletten Statistik- Output in eine Log-Datei (nohup.out). Das macht vielleicht bei Programmen Sinn, bei denen wir auf eine spezielle Anzeige warten (beim Kompilieren oder einen Virenschann etc.) . Bei einer Downloadstatistik bekommen wir mal schnell ein paar MB zusammen, was wir nicht unbedingt haben wollen.

Also stellen wir noch hinter den curl- Befehl den Parameter **--silent** . Somit sieht der komplette Befehl wie folgt aus:

```
nohup curl -C - -O http://laotzu.acc.umu.se/debian-cd/4.0_r0/i386/iso-cd/debian-40r0-i386-netinst.iso --silent
```

Zum Test können wir uns nach dem Start des Downloads ausloggen und als anderer User (root) einloggen. Dort kontrollieren wir mittels **htop** ob der Prozess noch läuft:



```
CPU[||||||||||||||||||||||||||||| 62.9%] Tasks: 29 total, 1 running
Mem[||||||||||||||||||||||||||| 7/29MB] Load average: 1.59 1.19 1.12
Swp[|| 4/117MB] Uptime: 02:37:14

  PID USER   PRI  NI  VIRT   RES   SHR  S  CPU% MEM%   TIME+  Command
  9070          16  10   912    908   756  S  56.0  3.0   0:57.92 curl -C - -O http:
  9078 root     18  10  1000   1000   816  R   5.0  3.3   0:02.04 /opt/bin/htop
    1 root     8   0   220    168   168  S   0.0  0.6   0:08.42 /bin/init
   54 root     9   0   476    396   396  S   0.0  1.3   0:00.01 /bin/sh
   55 root     9   0   540    488   488  S   0.0  1.6   0:00.15 /sbin/syslogd -n
   56 root     9   0   504    428   428  S   0.0  1.4   0:00.11 /sbin/klogd -n
  331 root     9   0  1036    916   828  S   0.0  3.0   0:00.11 /usr/sbin/thttpd -
  355 root     9   0  1900    488   488  S   0.0  1.6   0:00.02 /usr/sbin/smbd -D
  357 root     9   0  1836    892   620  S   0.0  3.0   0:00.14 /usr/sbin/smbd -D
  375 root     9   0  1876    464   464  S   0.0  1.5   0:00.00 /usr/sbin/smbd -D
  386 root     8   0   540    424   424  S   0.0  1.4   0:00.01 /usr/sbin/QuickSet
  390 root     9   0   536    480   440  S   0.0  1.6   0:00.04 /usr/sbin/USB_Dete
  392 root     9   0   484    384   384  S   0.0  1.3   0:00.01 /usr/sbin/USB_Dete
  398 root     9   0   352    268   248  S   0.0  0.9   0:00.00 /usr/sbin/onetouch
  402 root     9   0   456    352   352  S   0.0  1.2   0:00.01 /usr/sbin/onetouch
  411 root     9   0   416    384   348  S   0.0  1.3   0:00.01 /usr/sbin/crond
  419 root     9   0   440    332   332  S   0.0  1.1   0:00.00 /usr/sbin/CheckRes

F1Help F2Setup F3Search F4Invert F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit
```

Perfekt! Hier sehen wir, dass curl ganz oben munter werkelt und ordentlich CPU Last verursacht. Das lag allerdings in meinem Beispiel nur daran, dass ich den **--silent** Mode nicht verwendete und dabei noch kräftig in die Log-Datei geschrieben wurde. Normal benötigt der curl bei solch einer Tätigkeit um die 15% CPU Last.

Alternativ kann man natürlich auch den curl zeitgesteuert loslaufen lassen. Dazu legt den Befehl (ohne nohup) in ein bash-script und ruft diesen über einen cron-job zur gewünschten Zeit auf. So kann der große Download auch mitten in der Nacht loslegen, wenn ihr die volle Bandbreite Eurer Internet-Verbindung eh nicht benötigt.

Die Secure-Shell

Wer nicht ständig über das doch ungeschützte telnet auf seine Box zugreifen möchte, kann dies auch über die Secure-Shell (ssh) erledigen. Dazu installiert diese einfach mit einem

```
ipkg install openssh
```

Der Vorteil hierbei ist, dass nicht jedes Mal der Telnet aktiviert werden muss und die Angelegenheit entsprechend verschlüsselt über den Äther geht.

Mehrere Konsolen mit Screen

Das kleine Programm [Gnu Screen](#) ermöglicht den User auf einer Konsole mehrere Screens, also Konsolen zu aktivieren, zwischen denen man bequem umschalten kann. So kann beispielsweise eine Log-Datei mit tail oder der htop mitlaufen, während man am nächsten Screen weitere Arbeiten tätigt.

Installiert wird Screen einfach durch ein

```
ipkg install screen
```

Das Programm wird dann an der Konsole mit einem

```
/opt/bin/screen
```

gestartet. Soltet Ihr **/opt bin** bereits als Suchpfad integriert haben, dann genügt ein

```
screen
```

Damit auch im "Screen" die bash verwendet wird, legt in Eurem Home-Verzeichnis des Users, der eben screen starten darf eine Konfigurationsdatei an:

```
vi .screenrc
```

Dort ergänzt Ihr die folgende Zeile:

shell bash

Speichert diese und startet screen neu.

Wenn Ihr den Willkommensscreen abstellen wollt, dann schreibt folgende Zeile in die .screenrc:

startup_message off

Folgende Befehle soltet Ihr Euch merken:

Strg+a c Startet eine neue Konsole

Strg+a leer schaltet zwischen den Konsolen um

exit beendet die Konsole (wenn eine Konsole übrig wird screen beendet)

Strg+a d beendet screen, screen läuft aber im Hintergrund weiter

screen -R startet wieder die im Hintergrund noch laufende Screen-Session

Der wesentliche Vorteil von screen ist, dass alle Programme, die Ihr innerhalb der Screen-Session startet, auch dann weiter laufen, wenn Ihr Euch aus der Slug ausloggt. Somit könnt Ihr beispielsweise einen großen Download mittels curl starten, müsst aber nicht den nohup Befehl verwenden, sondern könnt jeder Zeit wieder Euch einloggen und mittels **screen -R** die Session aufnehmen und beispielsweise den statistischen Output von curl Euch ansehen.

MP3 Server mit Firefly

MP3s im heimischen Netz streamen? Na kein Problem! Mittels dem daap- Server Firefly kann Eure MP3 in Eurem Netz verfügbar gemacht werden. Ob am lokalen Rechner, am Laptop via WLAN oder mittels dafür geeignete/kompatible Stalonegeräte, die Slug sorgt für genug Futter.

Die offizielle Seite des Firefly findet sich hier:

<http://www.fireflymediaserver.org>

Zunächst legt über Eure Webadmin-Konfiguration einen neuen User für Eure Musik an (mp3 oder musik etc.) . Dieser sollte ausschließlich als Samba Share verwaltet werden und **keinesfalls** für einen FTP-Zugriff im Internet freigegeben werden. Sollte von außen auf Eure mp3 Sammlung ein Zugriff möglich sein, dann sind schlimmste Probleme vorprogrammiert! In unserem Beispiel nennen wir ihn **musik**. Dementsprechend hat er als Heimatverzeichnis das Verzeichnis **/musik**.

Der Server selbst ist sehr schnell installiert. Hierzu loggt Euch auf Eure Slug ein und führt folgende Befehle zur Installation aus:

```
ipkg update
ipkg install mt-daapd
```

Nun müssen wir die Konfigurationsdatei editieren:

```
vi /opt/etc/mt-daapd/mt-daapd.conf
```

Folgende Zeilen sind interessant:

```
[...]
port      3689 (Standard, kann geändert werden, muss aber nicht)
[...]
admin_pw  beliebiges Administratorpasswort (WICHTIG!)
[...]
mp3_dir   /musik (Das Verzeichnis für die Musikdateien)
[...]
```

Das war es im Grunde auch schon. Speichert Eure Änderung ab und startet den Server neu:

```
/opt/etc/init.d/S60mt-daapd
```

Wenn Ihr eine Logdatei mitlaufen lassen wollt müsst Ihr hierfür wieder die Konfiguration bearbeiten:

```
vi /opt/etc/mt-daapd/mt-daapd.conf
```

Ändert die Zeile

```
# logfile /var/log/mt-daapd.log
```

in

```
logfile /var/log/mt-daapd.log
```

um. Legt dann noch eine leere Logdatei an:

```
touch /var/log/mt-daapd.log
```

Auch für diese Änderung muss der Server neu gestartet werden:

```
/opt/etc/init.d/S60mt-daapd
```

Die Logdatei könnt Ihr Euch dann an der Konsole mittels

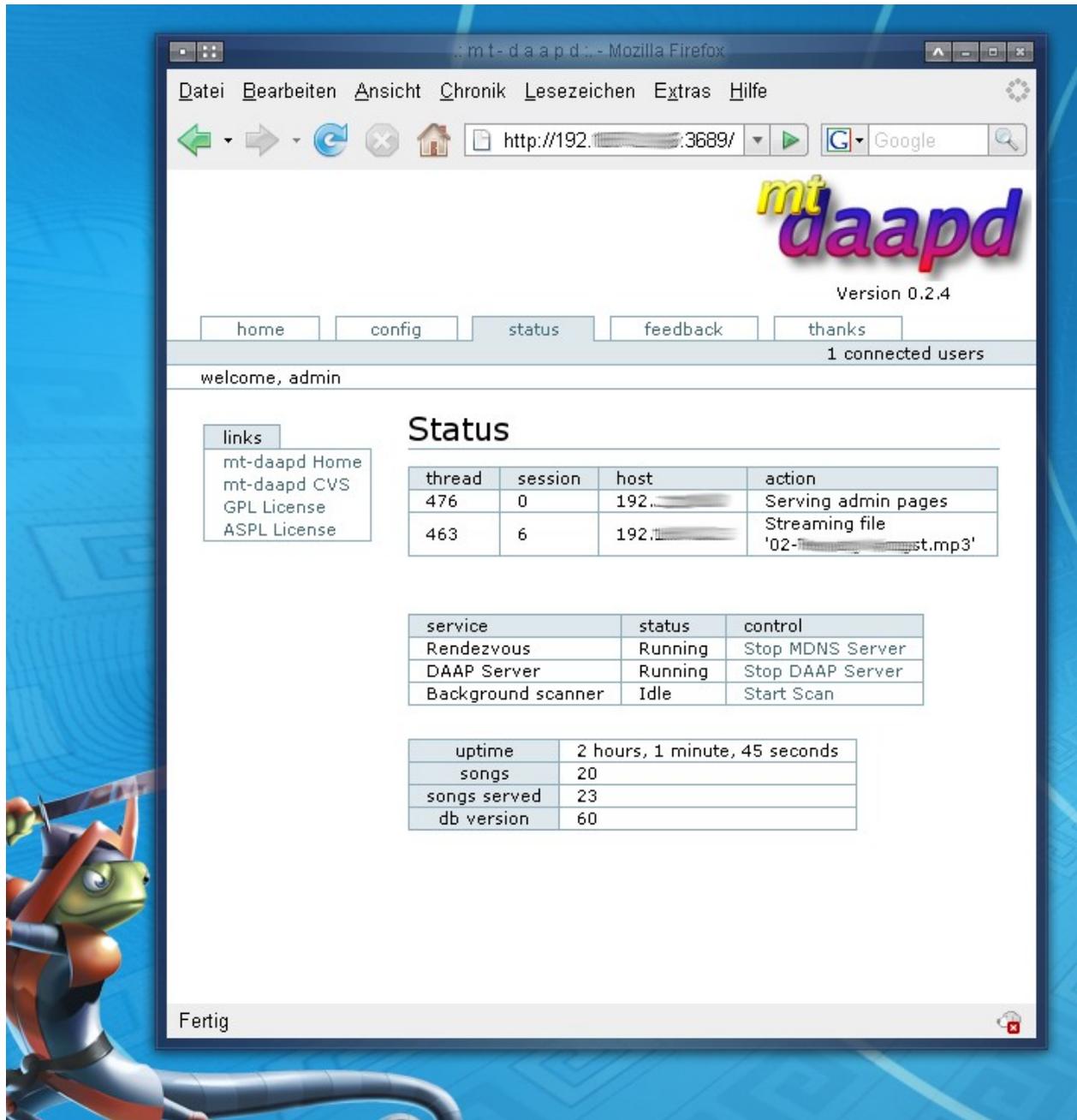
```
tail -f /var/log/mt-daapd.log
```

ansehen.

Ladet nun eine Test-mp3 auf Euer Musikverzeichnis und startet den Webadmin (mit dem in Eurer Config festgelegtem Port) in Eurem Webbrowser:

```
http://[IP-Nummer der Slug]:3689
```

Meldet Euch dann mit dem Usernamen **admin** mit Eurem Passwort an
Klickt nun auf "**status**" :

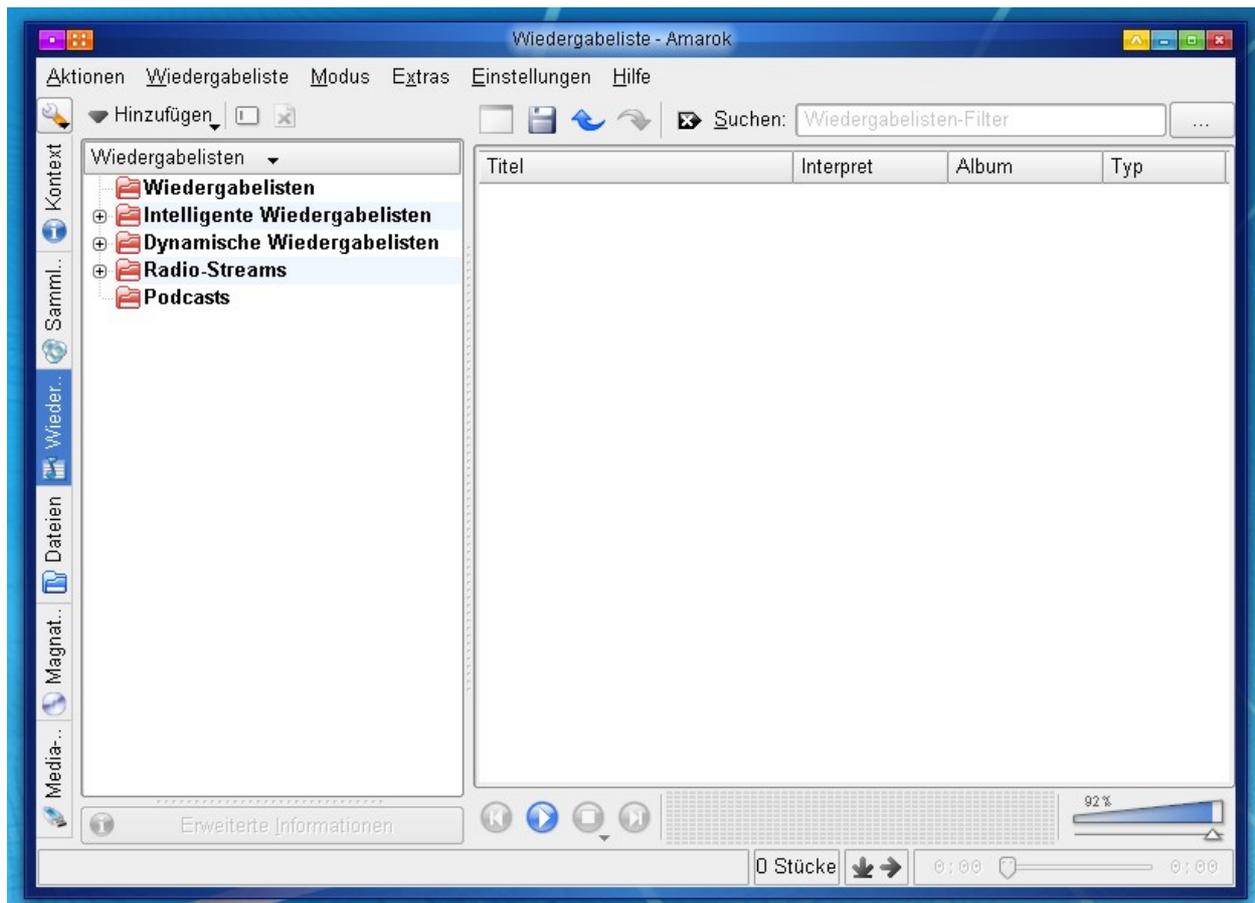


Klickt dann auf Start Scan, damit Eure Dateien eingelesen werden. Kurz darauf sollte die Anzahl der auf dem Server geladenen Dateien angezeigt werden.

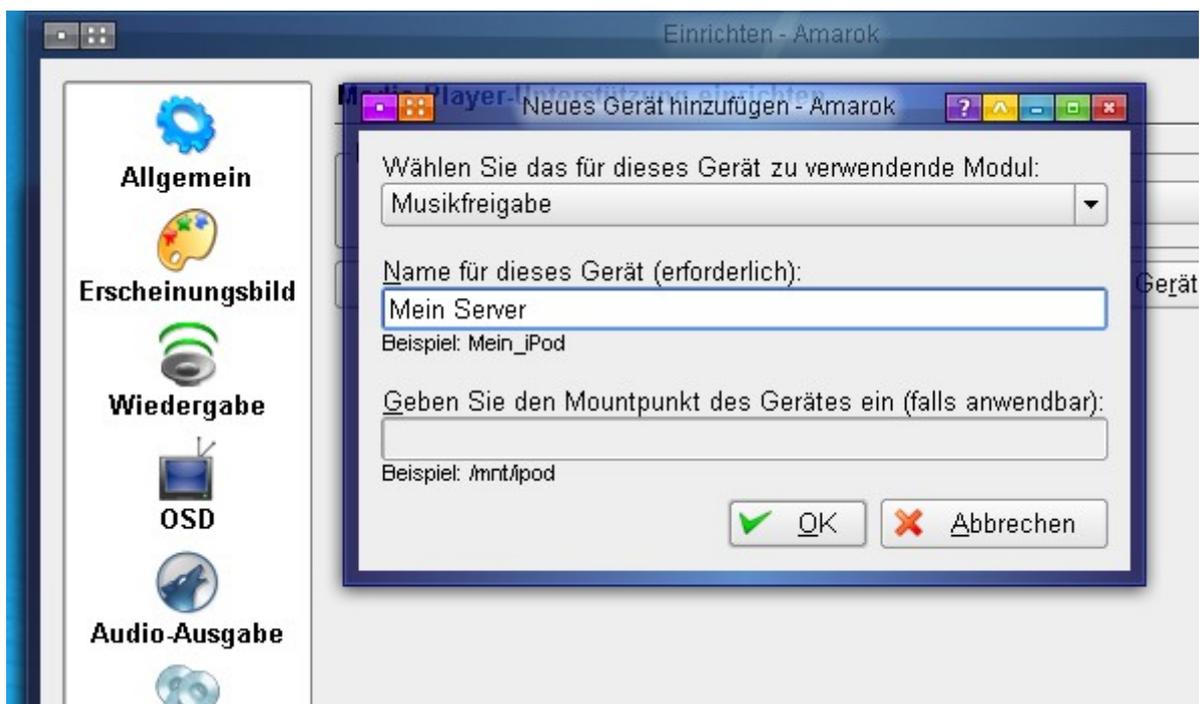
Mittels einem geeigneten Programm (iTunes, Amarok um nur zwei zu nennen) können nun die Musikdateien abgespielt werden.

Hier ein kurzes Beispiel, wie der Amarok unter Linux dafür konfiguriert wird:

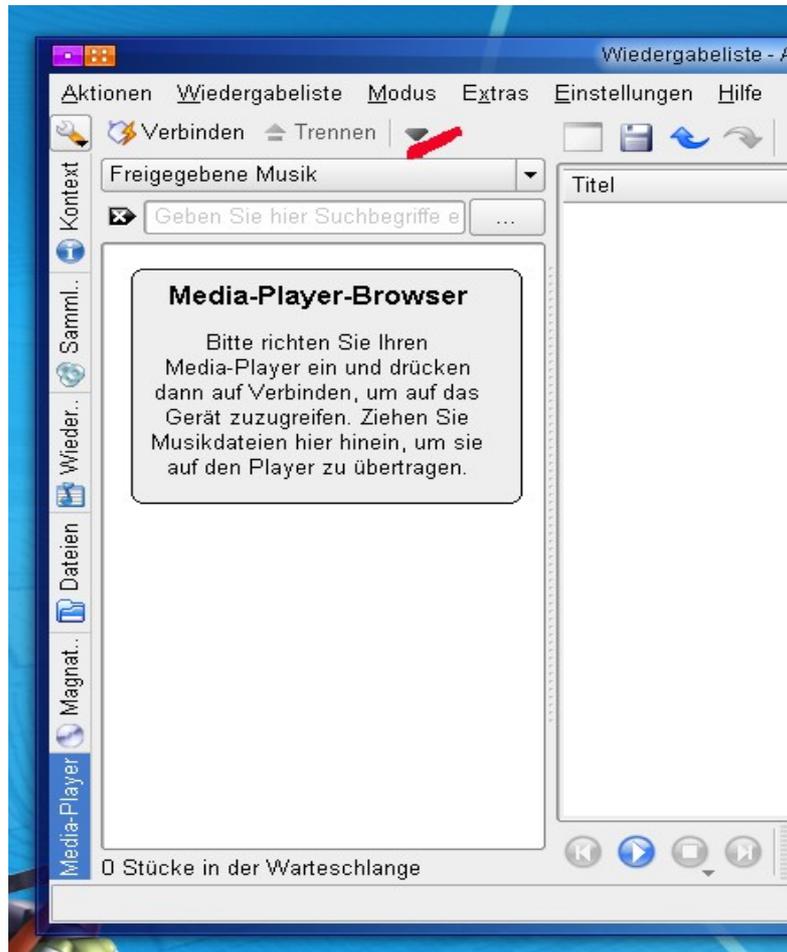
Startet dazu Euren Amarok:



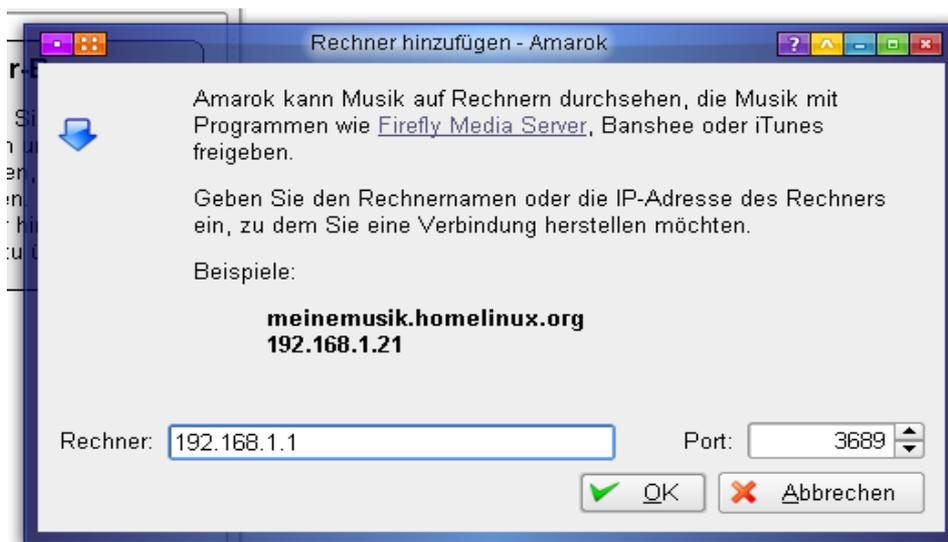
Dann klickt auf „Einstellungen -> Amarok einrichten ...“ Dann links auf "Media-Player" und "Gerät hinzufügen". Im Dropdownmenü wählt Ihr **Musikfreigabe** aus und vergibt einen beliebigen Namen:



Ihr beendet das kleine Menü mit **OK** und schließt das Konfigurationsfenster mit **"Anwenden"**. Danach könnt Ihr im Dropdownmenü oben links das Gerät **"Freigegebene Musik"** auswählen und klickt auf den kleinen Pfeil rechts neben den Buttons Verbinden/Trennen (rot markiert):



Nun müsst Ihr die IP Eures Rechners im Netz mit dem DAAP Server eingeben. Den Port stellt falls nötig gemäß Eurer Serverkonfiguration ein:



Das dann auch mit **OK** beenden. Klickt nun auf **Verbinden**. Danach solltet in der Liste darunter Euer Server auftauchen (IP) und Eure MP3s gelistet werden.

Jetzt könnt Ihr anfangen, Eure Musikauswahl zu starten.

Firewall und IP-Falle

Wer im Netz einen Webserver betreibt, der wird sich immer wieder Attacken gegen seinen Server ausgesetzt fühlen. Das sind nicht oft gezielte Angriffe, sondern eher Suchmechanismen, die an diversen Standardports herausfinden, ob diverse Dienste wie http oder ssh freigegeben sind. An solchen freigegeben Ports werden dann entsprechende Einbruchsversuche gestartet. Dies geschieht über verschiedene Wege. Die einfachste aber am häufigsten verwendete Methode ist der sogenannte Brute Force Angriff. Hier versucht der Angreifer mittels vorgefertigter Passwortlisten das korrekte Passwort zu erraten. Dabei werden anhand dieser Liste Logins solange probiert, bis das passende Passwort gefunden ist. Ähnliches kann natürlich auch am FTP Server passieren.

Dieses Tutorial soll nun eine Möglichkeit aufzeichnen, solchen Brut Force Attacken zu begegnen und zusätzlich die NSLU2 über ein Firewall-Script soweit abzudichten, bis eben auf die für die NSLU2 benötigten Ports. Das Tutorial soll allerdings keinesfalls eine falsche Sicherheit vorgaukeln. Eine Firewall und eine IP Falle, wie wir sie im nachfolgenden Text über das Programm Fail2Ban realisieren werden kann nur eine gewisse Sicherheit bieten. Wichtig ist zudem weiterhin, seine Log-Dateien regelmäßig im Auge zu behalten und auch zu kontrollieren, welche Dateien sich auf der BÜchse befinden. Bekommen die Dateien ungebetenen Zuwachs, dann ist wohl der ftp-Server geknackt worden.

Das von mir verfasste Tutorial fasst einige Arbeitsschritte zusammen, die für sich betrachtet auch noch ausbaufähig sind. Aufgrund der Komplexität des Themas kann dies auch nicht wirklich vollständig sein und stellt in dieser Hinsicht eine Art Grundgerüst dar, auf das freilich weiter aufgebaut werden kann.

Da das Konfigurieren einer Firewall nicht wirklich trivial ist sei noch gesagt: Bevor Ihr irgendwelche Scripte automatisiert, startet diese IMMER erst manuell und testet die Funktionalität der Scripte. Sperrt Ihr Euch aus Versehen durch eine Falschkonfiguration aus und die Scripte werden nach einem Reboot automatisch gestartet, weil diese bereits in den Initscripten integriert wurden habt Ihr wenig bis gar keine Chance mehr, auf Eure BÜchse zu kommen. Hier muss das Gerät dann wieder mit einer frischen Firmware geflasht werden und Ihr könnt das Gerät wieder von vorne neu einrichten.

Ziel dieses Tutorials ist es nun, eine IP Falle mit dem Programm Fail2ban zu installieren. Fail2ban findet Ihr unter folgender URL:

http://www.fail2ban.org/wiki/index.php/Main_Page

Mittels dieser "Falle" können Loginversuche limitiert werden. IP-Nummern über die nun versucht wird, ein Passwort zu erraten, werden nach einer vordefinierten Anzahl von Fehlversuchen einfach gesperrt. Die Sperre wird dann nach einer gewissen Zeit wieder aufgehoben. Hierbei empfiehlt sich eine Sperre von mehreren Tagen, damit der Versuch, einen Einbruch über eine Passwortliste zu unternehmen ad Absurdum führt. Zudem sollen Einbruchversuche uns per E-Mail gemeldet werden. Zum Schluss wollen wir noch eine Firewall für die restlichen Ports unseres Servers einrichten.

Also lest alles in Ruhe durch, nehmt Euch Zeit für Euer Projekt, dann werdet Ihr denke ich auch eine Menge Spaß dabei haben!

Basis Pakete installieren

Zuerst sollten wir - falls nicht schon aufgrund meiner vorangegangenen Tutorials geschehen - die NSLU2 durch die Entwicklerpakete ergänzen, damit wir bei Bedarf etwas kompilieren können:

```
ipkg install optware-devel  
ipkg install python
```

Danach modifizieren wir die `/etc/profile`, damit wir einen sauberen Kompile hinbekommen:

```
vi /etc/profile
```

Ergänzt dann folgende Zeilen und speichert die Datei ab:

```
export PATH=$PATH:/opt/bin:/opt/sbin:/opt/usr/bin:/opt/usr/sbin:.  
export LD_LIBRARY_PATH=/opt/lib
```

Meldet Euch erstmal aus Eurer Slug ab und dann wieder an, damit die Suchpfade übernommen werden.

Syslog ändern

Damit wir ein paar mehr wichtige Informationen wie eben auch die Sicherheitsmeldungen des openssh Servers geliefert bekommen, müssen wir den bisherigen Syslog Dämonen durch einen anderen ersetzen. Dies ist schnell erledigt!

Zuerst das neue Syslogpaket installieren

```
ipkg install syslog-ng
```

Danach in **/etc/inittab** folgende Zeilen auskommentieren:

```
# slog:unknown:/sbin/syslogd -n  
# klog:unknown:/sbin/klogd -n
```

Alte Logdaemons abschießen:

```
killall syslogd  
killall klogd
```

Syslog dann mit

```
/opt/etc/init.d/S01syslog-ng
```

starten. Beim nächsten Reboot wird das Programm automatisch gestartet.

Damit die Logdateien nicht auf Dauer zu groß werden und eventuell gerade bei der Verwendung von etwas knapp bemessenen USB-Sticks zu vollen Datenträgern zu führen, empfehle ich die Verwendung des logrotate. Dieser sorgt dafür, dass die schnell wachsenden System-Logs in separaten gepackten Dateien gespeichert werden und die aktuelle Log-Datei herunter gekürzt werden. Dies geschieht rotierend, d.h. die gepackten Dateien werden durchnummeriert und die älteste jeweils gelöscht.

Hierzu installiert bitte den logrotate mit folgendem Befehl:

```
ipkg install logrotate
```

Jetzt editiert die **/opt/etc/logrotate.conf** mit dem

```
vi /opt/etc/logrotate.conf
```

und ändert diese wie folgt ab:

```
compress

/opt/var/log/messages opt/var/log/syslog opt/var/log/kern.log opt/var/log/d**** {
    rotate 5
    postrotate
        /bin/killall syslogd
        /bin/killall klogd
        /bin/killall -HUP syslog-ng
    endscrip
}

include /opt/etc/logrotate.d
```

rotate 5 bedeutet, dass Ihr 5 Sicherungen bekommt, bis die älteste bei der 6. Sicherung gelöscht wird. Damit das auch automatisiert funktioniert, editiert Eure crontab:

```
vi /etc/crontab
```

und fügt folgende Zeile ein:

```
30 23 * * 6 root /opt/sbin/logrotate -f /opt/etc/logrotate.conf &>/dev/null
```

Das löst jeden Samstag um 23:30 Uhr eine Bereinigung der Logfiles aus. Somit habt Ihr die Files zumindest 5 Wochen zur Verfügung, solltet Ihr irgendwelchen Unregelmäßigkeiten auf die Spur kommen wollen. Die gepackten Dateien befinden sich dann unter **/opt/var/log** und können mit dem Midnight Commander beispielsweise eingesehen werden.

Iptables / Firewall installieren

Bevor wir uns nun an das Eingemachte machen können, müssen wir noch diverse Kernelmodule nachinstallieren. Dann kommen die iptables dran, mit denen wir im weiteren Verlauf des Tutorials noch viel Freude haben werden. Folgende 4 Pakete werden nun installiert:

```
ipkg install kernel-module-ip-tables -force-depends
ipkg install kernel-module-iptables-filter -force-depends
ipkg install kernel-module-iptables-log
ipkg install iptables
```

Startet nun sicherheitshalber Eure Slug neu, damit alles sauber initialisiert wird.

Erweiterung der Firewall mit IPCHAINS

Wer seine Slug noch weiter abdichten möchte, der kann hier mit einem sehr guten Script arbeiten, das man auf der NSLU2-Linux.org findet. Dieses fügt weitere Regeln hinzu und sperrt die Ports. Wobei Standarddienste wie der ftp und der http Server offen bleiben. Sollen weitere Ports geöffnet werden, so muss das in dem folgenden Script hinterlegt werden.

Jetzt ein Firewallscript anlegen:

```
touch /opt/etc/iptables.sh
```

Das machen wir ausführbar:

```
chmod +x /opt/etc/iptables.sh
```

Dann füllen wir den Inhalt mit folgender ausgezeichneten Firewallregel, welches ich auf der nslu2-linux.org gefunden habe (Link folgt nach dem Script):

```
#!/bin/sh
#####
#####
#
# Local Settings
#

# IPTables Location - adjust if needed

IPT="/opt/sbin/iptables"
IPTS="/opt/sbin/iptables-save"
IPTR="/opt/sbin/iptables-restore"

# Internet Interface
INET_IFACE="ixp0"

# Path to iptables modules
IPT_MODPATH="/lib/modules/2.4.22-xfx/kernel/net/ipv4/netfilter"

# CHANGE THIS TO MATCH YOUR SLUG IP ADDRESS
# currently not used
INET_ADDRESS="10.0.0.123"

# Localhost Interface

LO_IFACE="lo"
LO_IP="127.0.0.1"

# Save and Restore arguments handled here
if [ "$1" = "save" ]
then
    echo -n "Saving firewall to /etc/sysconfig/iptables ... "
    $IPTS > /opt/etc/iptables
    echo "done"
    exit 0
elif [ "$1" = "restore" ]
then
    echo -n "Restoring firewall from /etc/sysconfig/iptables ... "
    $IPTR < /opt/etc/iptables
    echo "done"
    exit 0
fi
# Load Modules

echo "Loading kernel modules ..."
```

```

insmod $IPT_MODPATH/ip_tables.o
insmod $IPT_MODPATH/iptables_filter.o
# if you need logging, uncomment line above
# (be aware you have installed the kernel-module-ipt-log
#insmod $IPT_MODPATH/ipt_LOG.o

# Flush Any Existing Rules or Chains

echo "Flushing Tables ..."

# Reset Default Policies
$IPT -P INPUT ACCEPT
$IPT -P FORWARD ACCEPT
$IPT -P OUTPUT ACCEPT

# Flush all rules
$IPT -F

# Erase all non-default chains
$IPT -X

if [ "$1" = "stop" ]
then
    echo "Firewall completely flushed! Now running with no firewall."
    exit 0
fi

#####
#####
# Rules Configuration

# Filter Table

# Set Policies

$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

# User-Specified Chains

echo "Create and populate custom rule chains ..."

# Create a chain to filter INVALID packets

$IPT -N bad_packets

# Create another chain to filter bad tcp packets

$IPT -N bad_tcp_packets

```

```

# Create separate chains for icmp, tcp (incoming and outgoing),
# and incoming udp packets.

$IPT -N icmp_packets

# Used for UDP packets inbound from the Internet
$IPT -N udp_inbound

# Used to block outbound UDP services from internal network
# Default to allow all
$IPT -N udp_outbound

# Used to allow inbound services if desired
# Default fail except for established sessions
$IPT -N tcp_inbound

# Used to block outbound services from internal network
# Default to allow all
$IPT -N tcp_outbound

# Populate User Chains

# bad_packets chain

# Drop INVALID packets immediately
# needs conntrack
#$IPT -A bad_packets -p ALL -m state --state INVALID -j DROP

# Then check the tcp packets for additional problems

$IPT -A bad_packets -p tcp -j bad_tcp_packets

# All good, so return
$IPT -A bad_packets -p ALL -j RETURN

# bad_tcp_packets chain
#
# All tcp packets will traverse this chain.
# Every new connection attempt should begin with
# a syn packet. If it doesn't, it is likely a
# port scan. This drops packets in state
# NEW that are not flagged as syn packets.

# needs conntrack
#$IPT -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j LOG \
  #--log-prefix "New not syn: "
#$IPT -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j DROP

# Stealth scans

```

```

$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL NONE -j DROP
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL ALL -j DROP
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

# All good, so return
$IPT -A bad_tcp_packets -p tcp -j RETURN

# icmp_packets chain
# ICMP packets should fit in a Layer 2 frame, thus they should
# never be fragmented. Fragmented ICMP packets are a typical sign
# of a denial of service attack.
#$IPT -A icmp_packets --fragment -p ICMP -j LOG \
  #--log-prefix "ICMP Fragment: "
$IPT -A icmp_packets --fragment -p ICMP -j DROP

# Echo - uncomment to allow your system to be pinged.
# Uncomment the LOG command if you also want to log PING attempts
#
# $IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j LOG \
#   --log-prefix "Ping detected: "
$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT

# comment out above and uncomment below to drop pings without logging.
#$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j DROP

# see ping reply packets
$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 0 -j ACCEPT

# Time Exceeded
$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT

# Not matched, so return so it will be logged
$IPT -A icmp_packets -p ICMP -j RETURN

# TCP & UDP
# Identify ports at:
#   http://www.chebucto.ns.ca/~rakerman/port-table.html
#   http://www.iana.org/assignments/port-numbers

#
# ADD UDP-based services here
#

# udp_inbound chain
# ports you want to accept udp packets on

# netbios/samba
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 137 -j ACCEPT

```

```

$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 138 -j ACCEPT

# Network Time Protocol (NTP) Server
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 123 -j ACCEPT

# External DHCP Server
# Allow DHCP client request packets inbound from external network
$IPT -A udp_inbound -p UDP -s 0/0 --source-port 68 --destination-port 67 -j
ACCEPT

# DNS in
#$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 53 -j ACCEPT
$IPT -A udp_inbound -p UDP -s 0/0 --source-port 53 -j ACCEPT

# Not matched, so return for logging
$IPT -A udp_inbound -p UDP -j RETURN

# udp_outbound chain
# ports you send udp packets to

# netbios/samba
$IPT -A udp_outbound -p UDP -s 0/0 --destination-port 137 -j ACCEPT
$IPT -A udp_outbound -p UDP -s 0/0 --destination-port 138 -j ACCEPT

# Network Time Protocol (NTP) Server
$IPT -A udp_outbound -p UDP -s 0/0 --destination-port 123 -j ACCEPT

# DHCP out
$IPT -A udp_outbound -p UDP -s 0/0 --destination-port 68 -j ACCEPT

# DNS out
$IPT -A udp_outbound -p UDP -s 0/0 --destination-port 53 -j ACCEPT

# No match, so ACCEPT
# make this DROP if you want to block any other outbound udp traffic
$IPT -A udp_outbound -p UDP -s 0/0 -j ACCEPT

# tcp_inbound chain
#
# This chain is used to allow inbound connections to the SLUG

# smb
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 137 -j ACCEPT
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 139 -j ACCEPT
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 445 -j ACCEPT

# HTTP
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT

# FTP
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port ftp -j ACCEPT

```

```

# Passive
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 33201:33210 -j ACCEPT

# DNS
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 53 -j ACCEPT

# sshd
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 22 -j ACCEPT

# telnet
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 23 -j ACCEPT

# Not matched, so return so it will be logged
$IPT -A tcp_inbound -p TCP -j RETURN

# tcp_outbound chain
#
# This chain controls what tcp traffic is allowed out

# http
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT
# DNS
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 53 -j ACCEPT
# sshd
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 22 -j ACCEPT

# No match, so ACCEPT
# Note, you could make this DROP to block any other outbound traffic

$IPT -A tcp_outbound -p TCP -s 0/0 -j ACCEPT

#####
#####
# INPUT Chain

echo "process INPUT chain ..."

# Allow all on localhost interface
$IPT -A INPUT -p ALL -i $LO_IFACE -j ACCEPT

# Drop bad packets
$IPT -A INPUT -p ALL -j bad_packets

# *****
# Inbound Internet Packet Rules

# Accept Established Connections
# Needs conntrack module
# $IPT -A INPUT -p ALL -i $INET_IFACE -m state --state ESTABLISHED,RELATED
-j ACCEPT

```

```

# packet filter accepts inbound packets that are replies to an outbound connection
# use until conntrack is available
# this blocks all new connection attempts except to those allowed below
$IPT -A INPUT -p TCP -i $INET_IFACE ! --syn -j ACCEPT

# Route the rest to the appropriate user chain
$IPT -A INPUT -p TCP -i $INET_IFACE -j tcp_inbound
$IPT -A INPUT -p UDP -i $INET_IFACE -j udp_inbound
$IPT -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets

# Drop without logging broadcasts that get this far.
# Comment this line if testing new rules that impact
# broadcast protocols.
#$IPT -A INPUT -m pkttype --pkt-type broadcast -j DROP

#####
#####
#
# OUTPUT Chain
#

echo "Process OUTPUT chain ..."

# Generally trust the firewall on output

# However, invalid icmp packets need to be dropped
# to prevent a possible exploit.
# needs conntrack
#$IPT -A OUTPUT -m state -p icmp --state INVALID -j DROP

# Localhost
$IPT -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPT -A OUTPUT -p ALL -o $LO_IFACE -j ACCEPT

# If you want to block outbound connections, uncomment first section below,
comment
# out second section, and add rules to tcp_outbound/udp_outbound

# To internet - filtered
#$IPT -A OUTPUT -p TCP -o $INET_IFACE -j tcp_outbound
#$IPT -A OUTPUT -p UDP -o $INET_IFACE -j udp_outbound

# To internet (unfiltered)
$IPT -A OUTPUT -p ALL -o $INET_IFACE -j ACCEPT

```

Das Script und einen Download findet Ihr unter folgender Adresse :

<http://www.nslu2-linux.org/wiki/HowTo/EnableFirewall>

Gestartet wird die Firewall mit

```
/opt/etc/iptables.sh
```

Anhalten mit

```
/opt/etc/iptables.sh stop
```

Um die Firewall bei jedem Start automatisch zu aktivieren müssen wir einen Link anlegen:

```
ln -s /opt/etc/iptables.sh /opt/etc/init.d/S30iptables
```

Ihr werdet hier jetzt bemerken, dass z.B. Euer Webadmin geblockt ist, weil dieser nicht auf dem Port 80 liegt, sondern einem anderen, den Ihr bereits in der Vergangenheit zugewiesen habt. Um nun den http auch für einen zweiten Port freizugeben ergänzt das Script durch eine neue Regel, die Ihr am besten unter die bisherige Port 80 Regel setzt. Sucht dazu folgende Zeilen im Script:

```
# http
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT
```

Diese erweitert Ihr folgendermaßen, wenn Ihr Eurem Webadmin z.B. den Port 9055 zugewiesen habt:

```
# http
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 9055 -j ACCEPT
```

Danach muss noch der inbound geändert werden:

```
# http
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT
```

Diese dann in

```
# http
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 9055 -j ACCEPT
```

Wenn Ihr z.B. den Firefly MP3 Server installiert habt, dann benötigt dieser auch einen freien Port auf 3689 beispielsweise (je nachdem, wie Ihr ihn konfiguriert habt). Gebt diesen Port frei, indem Ihr direkt bei den Portfreigaben im Script folgende Zeilen ergänzt:

```
# MT-DAAP
$IPT -A tcp_outbound -p TCP -s 0/0 --destination-port 3689 -j ACCEPT
$IPT -A INPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
$IPT -A OUTPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
```

Die IP 224.0.0.251 ist eine Multicast-Adresse, die hier Apples i-Tunes zum Rendezvous auf dem Port 5353 benötigt. Wird dieser geblockt findet der i-Tunes den NSLU2 Firefly Media Server nicht mehr im Netz.

Zum Schluss muss noch die **/opt/etc/vsftpd.conf** editiert werden, damit der passive Modus des vsftp FTP Server funktioniert und eine Verbindung wieder möglich wird. Ergänzt nun am Ende der **vsftp.conf** folgende Zeilen:

```
# for our firewall, only use this range of ports
pasv_min_port=33201
pasv_max_port=33210
```

Fail2ban installieren und konfigurieren

Zuerst benötigen wir ein Arbeitsverzeichnis. Das legen wir in dem Rootverzeichnis an:

```
mkdir work
```

Dann wechseln wir nach work

```
cd /work
```

Danach laden wir uns Fail2ban herunter:

```
wget http://downloads.sourceforge.net/fail2ban/fail2ban-0.7.9.tar.bz2?modtime=1177027414&big_mirror=0
```

Nun entpacken wir das Programm:

```
tar xvfj fail2ban-0.7.9.tar.bz2
```

Das erstellt uns ein neues Verzeichnis fail2ban-0.7.9 . Hier wechseln wir hinein:

```
cd fail2ban-0.7.9
```

Das wird dann mit

```
python setup.py install
```

installiert. Die wichtigen Dateien:

Die ausführbare Datei **fail2ban-client** liegt unter **/opt/local/bin** . Die dazugehörige Konfigurationsdatei **jail.conf** ist dann unter **/etc/fail2ban** zu finden, die dann noch editiert werden muss.

Öffnet nun die **/etc/fail2ban/jail.conf** und sucht nach folgender Sektion, um den ssh einzustellen:

```
[ssh-iptables]
enabled = false
filter = sshd
action = iptables[name=SSH, port=ssh, protocol=tcp]
        mail-whois[name=SSH, dest=yourmail@mail.com]
logpath = /var/log/sshd.log
maxretry = 5
```

Ich habe die z.B. wie folgt eingestellt:

```
[ssh-iptables]
enabled = true
filter = sshd
action = iptables[name=SSH, port=ssh, protocol=tcp]
        mail-whois[name=SSH, dest=yourmail@mail.com]
logpath = /opt/var/log/auth.log
maxretry = 3
bantime = 3600
```

Das bedeutet hier, dass beim Versuch via ssh sich einzuwählen beim dritten falschen Login (maxretry = 3) die IP für eine Stunde (bantime = 3600 ; wird in Sekunden angegeben) gesperrt ist. Das kann man natürlich je nach Gusto anpassen. Vergesst nicht, das **enabled = false** auf **true** umzustellen, damit der Dienst hier aktiviert wird.

Das Gleiche dann für unseren FTP Server (vsftpd):

```
[vsftpd-iptables]
enabled = false
filter = vsftpd
action = iptables[name=VSFTPD, port=ftp, protocol=tcp]
        mail-whois[name=VSFTPD, dest=yourmail@mail.com]
logpath = /opt/var/log/vsftpd.log
maxretry = 5
bantime = 600
```

in

```
[vsftpd-iptables]
enabled = true
filter = vsftpd
action = iptables[name=VSFTPD, port=ftp, protocol=tcp]
        mail-whois[name=VSFTPD, dest=yourmail@mail.com]
logpath = /opt/var/log/vsftpd.log
maxretry = 5
bantime = 3600
```

Hier erhöhen wir die Sperrdauer auf 1 Stunde und aktivieren den Dienst mit **enable = true** .

Speichert die Änderungen und startet nun den fail2ban indem Ihr zuerst mit

```
cd /opt/local/bin
```

in das korrekte Verzeichnis wechselt und dann

```
fail2ban-client start
```

eingibt. Nach erfolgreichen Start seht Ihr folgende Message:

2007-05-04 00:35:59,100 fail2ban.server : INFO Starting Fail2ban

Um zu sehen, ob auch die Dienste soweit online sind, setzt folgenden Befehl ab:

```
/opt/local/bin/fail2ban-client status
```

Der Output sollte folgender sein:

Status

|- Number of jail: 2

`- Jail list: ssh-iptables, vsftpd-iptables

Jetzt checken wir noch die IP Tables ob hier die korrekten Eintragungen vom fail2ban vorgenommen wurden:

```
iptables -L -v
```

Überprüft die Zeilen nun nach folgendem Output:

```
0 0 fail2ban-VSFTPD tcp -- any any anywhere anywhere tcp dpt:ftp  
181 16496 fail2ban-SSH tcp -- any any anywhere anywhere tcp dpt:ssh  
0 0 ACCEPT all -- lo any anywhere anywhere
```

Jetzt ist alles roger. Lasst Euch nun die Logfile des fail2ban anzeigen:

```
tail -f /var/log/fail2ban.log
```

Versucht Euch nun mit einem FTP Client einzuwählen und gebt bewusst ein falsches Passwort ein. Wenn Alles gut läuft, bekommt der FTP Client nach dem 5. Versuch keinen Connect mehr. Die Logdatei zeigt dann folgende Zeile:

```
2007-05-04 00:45:27,847 fail2ban.actions: WARNING [vsftpd-iptables] Ban  
192.168.0.2  
2007-05-04 00:45:28,775 fail2ban.actions.action: ERROR echo -en "Hi,\n"
```

```
The IP 192.168.0.2 has just been banned by Fail2Ban after
5 attempts against VSFTPD.\n\n
Here are more information about 192.168.0.2:\n
`whois 192.168.0.2`\n
Regards,\n
Fail2Ban"|mail -s "[Fail2Ban] VSFTPD: banned 192.168.0.2" yourmail@mail.com
returned 7f00
```

Hier seht Ihr, dass die IP gesperrt wurde. Wenn Ihr das ausprobieren solltet Ihr vorher vielleicht erstmal die Banddauer auf 60 Sekunden stellen, damit Ihr nicht ewig warten müsst, bis Ihr ge-unbant werdet:

```
2007-05-04 00:46:28,196 fail2ban.actions: WARNING [vsftpd-iptables] Unban
192.168.0.2
```

Natürlich kann man auch, um sich selbst nicht aus Versehen auszusperrern, seine eigene IP aus der Überwachung herausnehmen und in der **/etc/fail2ban/jail.conf** in die Zeile **ignoreip** eintragen.

Damit unser Filter auch beim nächsten Boot automatisch startet legen wir jetzt erstmal ein Script an:

```
vi /opt/bin/iptrap
```

Das füllen wir mit folgendem Script, das ich einmal kurz zusammengeschrieben habe:

```
#!/bin/bash
# Bootscript fuer den fail2ban Client

case "$1" in
  start)
    echo "Starting IP Trap..."
    exec /bin/fail2ban-client start
    ;;

  stop)
    echo "Stopping IP Trap... leaving machine wide open now ..."
    exec /bin/fail2ban-client stop
    ;;

  restart)
    $0 stop
    sleep 1
    $0 start
    ;;

  *)
    echo "Usage: $0 {start|stop|restart}"
    exit 1
    ;;
esac
```

Damit diese ausführbar ist, fügen wir das Executable Flag dazu:

```
chmod +x /opt/bin/iptrap
```

Diese kommt nun zu unseren init-Scripten:

```
In -s /opt/bin/iptrap /opt/etc/init.d/S35iptrap
```

Danach verlinken wir unsere Fail2ban Dateien nach **/bin**:

```
In -s /opt/local/bin/fail2ban-client /bin/fail2ban-client
In -s /opt/local/bin/fail2ban-regex /bin/fail2ban-regex
In -s /opt/local/bin/fail2ban-server /bin/fail2ban-server
```

Jetzt können wir die Kiste neu starten und hoffen, dass wir ein Stückchen sicherer geworden sind:

```
sync  
reboot
```

Meldung per E-Mail

Damit wir nicht immer regelmäßig die Log-Datei nach Sperrungen durchsuchen zu müssen, gibt es einen sehr komfortablen Weg, sich die Sperrungen per E-Mail zusenden zu lassen. Dabei ist es sogar möglich, gleich eine whois Abfrage zu starten und das Ergebnis in die Mail zu schreiben. Hierzu muss der Mailer **nail** und **whois** installiert werden:

```
ipkg install nail  
ipkg install whois
```

Danach verknüpfen wir das Programm nail mit dem Programmaufruf mail, weil die ganzen Scripte ein "mail" anstatt "nail" erwarten:

```
In -s /opt/bin/nail /opt/bin/mail
```

Jetzt müssen wir noch unseren smtp Server und Absender festlegen. Dazu müssen wir unter dem Heimatverzeichnis von /root die Datei .mailrc anlegen. Am besten mit vi aufmachen:

```
vi .mailrc
```

und folgende Zeilen eingeben (entsprechend Euren Server/Absender abändern):

```
set smtp=smtp. euerserver. de  
set from=meine @mail. blah
```

Wenn der SMTP Server SMTP-Auth verwendet (also einen Nutzer und Passwort abverlangt) dann ergänzt man die mailrc noch mit folgenden beiden Zeilen:

```
set smtp-auth-user=Loginname  
set smtp-auth-password=Passwort
```

Da hier das Passwort in klar lesbarer Form in der **.mailrc** hinterlegt ist muss sichergestellt sein, dass nur Root Leserechte an dieser Datei hat! SMTP-Auth ist manchmal nicht notwendig, wenn der SMTP Server zum Anbieter der eigenen Telefon/Internetanbindung gehört. In diesem Fall wird alleine durch den Zugang über das anbiertereigene Netz sicher gestellt, dass auch nur legitimitierte User Zugriff auf den SMTP Server haben.

Nun müssen wir noch Eure Empfängeradresse festlegen. Hierzu öffnet wieder die **/etc/fail2ban/jail.conf** und ändert in jedem Abschnitt, den Ihr auf **"true"** gesetzt habt die Platzhalter Adresse

dest=yourmail @mail. com

entsprechend Eurer gewünschten Empfängeradresse um.

Stoppt nun Eure Firewall und startet diese wieder neu, damit die Änderungen übernommen werden. Wenn alles richtig konfiguriert ist bekommt Ihr nun gleich die erste Mail, dass der Fail2ban Dienst gestartet wurde.

Wenn Ihr Eure IP-Falle allerdings beim Systemstart automatisch hochfahren lasst, kann es sein, dass Ihr keine E-Mails erhaltet. Das liegt daran, dass nicht der User "root" den Dienst startet, sondern das System selbst. Um dennoch auch beim automatischen Start die Mails zu bekommen fügt die SMTP Daten einfach in die **/opt/etc/nail.rc** ein, damit diese als Standardeinstellungen übernommen werden. Wenn Ihr die Slug nun neu startet sollte dann nach erfolgreichem Bootup die Meldung in Eurem Postfach landen, dass die IP Falle nun scharf ist:

Hi,

The jail SSH has been started successfully.

Regards,

Fail2Ban

An der Stelle komme ich nun zum Ende meines Vorschlages, die NSLU2 im Netz einwenig sicherer zu machen. Dennoch vergesst nie, wie ich es bereits eingangs schon erwähnte, die Augen offen zu halten und Eure Log-Dateien regelmäßig zu kontrollieren!

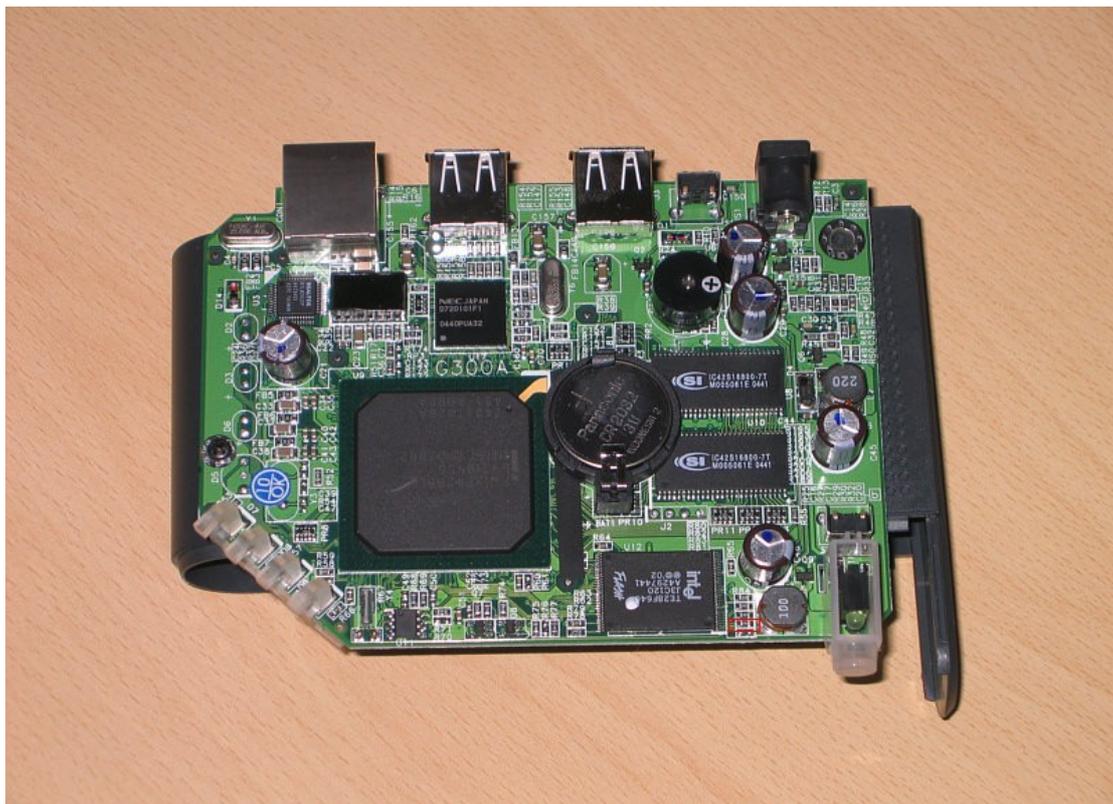
Phönix aus der Asche ... oder mehr Power

HINWEIS/UPDATE: Neue Geräte werden bereits mit der 266MHz Taktung ausgeliefert. Hier ist dieser Schritt nicht mehr notwendig!

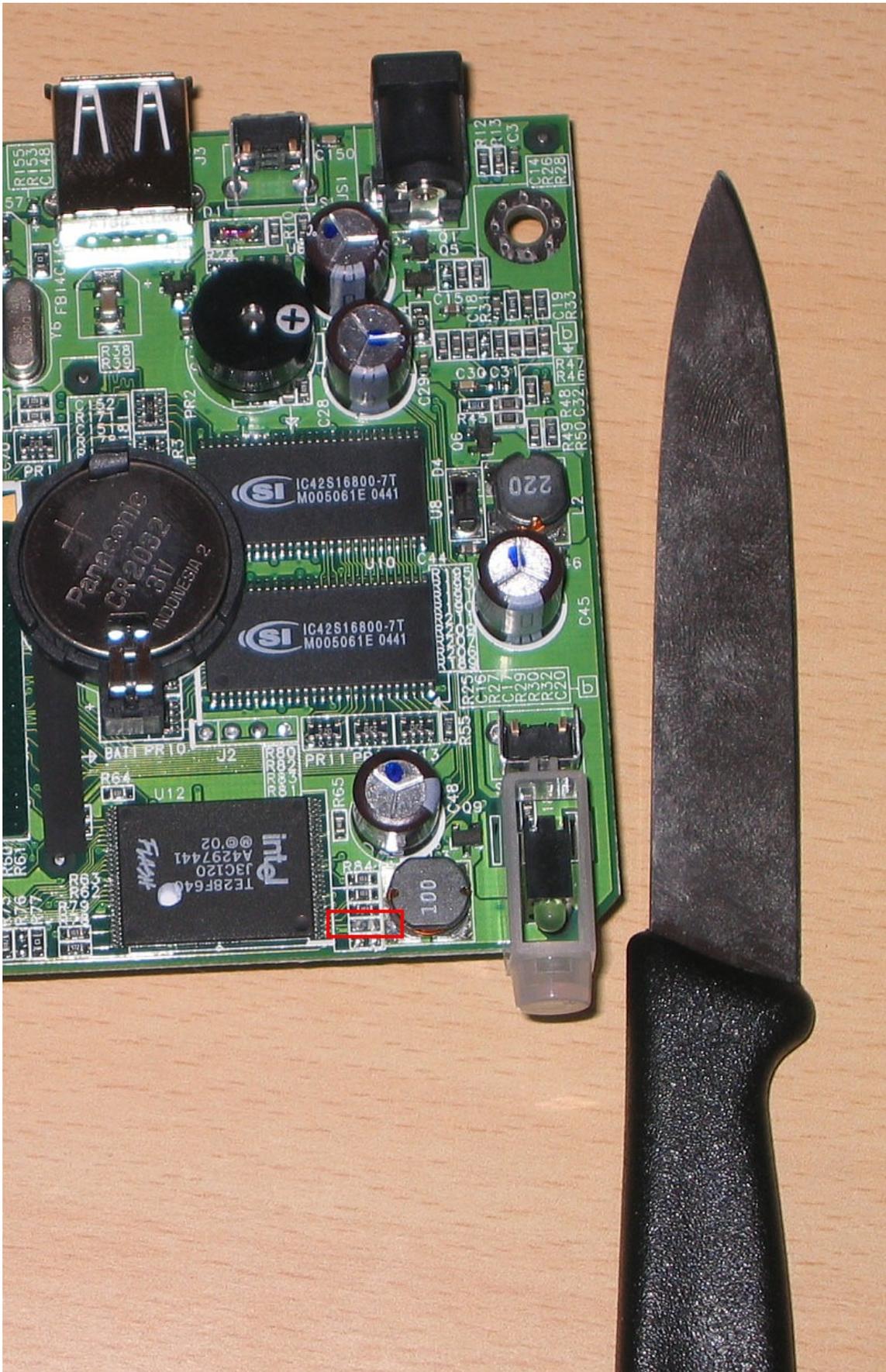
Nachdem ich im Netz einwenig recherchiert habe, habe ich herausgefunden, dass man durch einen kleinen Overclocker-Trick die NSLU2 merklich beschleunigen kann. Verbaut wurde eine 133MHz Intel ARM-CPU (XScale), die allerdings komischerweise laut der machbaren Spezifikation im Grunde nur auf halben Coretakt läuft. Möglich und ausgelegt ist die CPU auf 266 MHz. Durch einen kleinen Eingriff in das Gerät, kann die Sperre aufgehoben werden. Dazu muss das Gerät geöffnet und ein kleiner Widerstand entfernt werden.

Hinweis: Dadurch geht die Garantie verloren. Ebenso sind technische Defekte nicht auszuschließen, die eventuell auch Schäden anrichten können! Allerdings erhöht sich angeblich dadurch der Chip nicht merklich an Temperatur, bzw. die verbrauchte Leistung ist nicht wesentlich höher.

Hier das geöffnete Gerät und lokalisierter Widerstand:



Man kann den Widerstand auslöten (dazu benötigt man feines Werkzeug) oder man schneidet ihn mit einem scharfen Messer vorsichtig durch:



Bitte testet das Gerät nochmal genau durch, ob alle Funktionen erhalten geblieben

sind. Bei mir hat das soweit wunderbar funktioniert. Das muss aber nicht bedeuten, dass es überall so läuft! Also wie gesagt: Nur auf eigene Gefahr und die Garantie des Gerätes ist damit definitiv futsch!

Um nun sicher zu gehen, dass die Slug auch mit 266MHz taktet, kann man sich die BogoMIPS anzeigen lassen. Das passt zwar nicht haargenau, aber zeigt, dass der Takt nun entsprechend angehoben ist:

```
root@slug:~# cat /proc/cpuinfo
Processor       : XScale-IXP425/IXC1100 rev 1 (v5b)
BogoMIPS       : 263.78
Features        : swp half thumb fastmult edsp

Hardware       : Intel IXDP425 Development Platform
Revision       : 0000
Serial         : 0000000000000000
```

Das Datenblatt zum XScale-IXP425 kann bei Intel hier eingesehen werden:

<http://www.intel.com/design/network/manuals/252480.htm>

Dort steht es als downloadbares pdf-Dokument offen zur Verfügung.

Hinweise

Diese Dokumentation wurde mit besten Wissen und Gewissen erstellt. Tippfehler und sonstige Errata sind durchaus möglich. Gebt mir bitte Bescheid, wenn Ihr hierzu Verbesserungen habt. Zudem bitte ich Euch, erst nach Rückfrage Teile oder die komplette Doku zu kopieren. Normal ist das kein Problem, nur möchte ich schon wissen, für was und wo Ihr das verwendet.

Die kommerzielle Verwendung der Doku (Verkauf oder entgeltliche Veröffentlichung) ist ausdrücklich **NICHT** erlaubt!

Forchheim 2007, Pierre Kretschmer

(mail: pierre.kretschmer@pierre-kretschmer.de)

Links

NSLU2 Linux:

<http://www.nslu2-linux.org>

Dateiliste

<http://ipkg.nslu2-linux.org/feeds/unslung/wl500g/>

Aktuelle Firmware:

<http://www.nslu2-linux.org/wiki/Unslung/HomePage>

Diskussionsbeitrag im MF Forum zum Thema NSLU2

<http://forum.mindfactory.de/modem-netzwerke-allgemein/8362-nslu2-und-nun.html>

(Hier kann man mich auch direkt erreichen)

Meine (Gargi's) Homepage:

<http://www.gargi.org>