

# OpenWRT Open Source Router

PC-Treff-BB Aidlingen

Günter Waller

# Agenda

- Installation, Konfiguration, erste Schritte
- Schnittstellen
- Netzwerk-Übersicht, Firewall
- Zugriffe von innen und außen
- Dienste, Anwendungen
  - DHCP, DNS
  - Wake on LAN, Port Forwarding
  - Netzwerkdrucker
  - Zeitsteuerung
  - Künftige: Videoüberwachung, NAS, Owncloud

## Installation, Konfiguration, erste Schritte

- OpenWRT Wiki: Table of Hardware
  - Nach Herstellern gegliedert, große Zahl an Geräten
  - <http://wiki.openwrt.org/toh/start>
  - In meinem Fall:  
<http://wiki.openwrt.org/toh/tp-link/tl-wr1043nd#installation>
- Die Installationsmethode hängt vom Gerät ab. Es gibt 4 Methoden
  - Via Original Firmware Update-Funktion
  - Via Original Bootloader über Ethernet
  - Via Original Bootloader über Serial Port
  - Via JTAG (elektr. Kontakte auf dem Board) mit Spezialkabel vom PC-Drucker aus
- Einfach ist nur die erste Methode. Alles andere sehe ich nur als letztes Mittel in Notfällen.

# Installation

1. [obtain.firmware](#) and [Latest OpenWrt Release](#) - required image file is "openwrt-ar71xx-generic-tl-wr1043nd-v1-squashfs-factory.bin" from the "ar71xx" directory
2. [generic.flashing](#) Now write this firmware-file onto the flash-chip of your device

⚠ **NOTE:** If case you have a revision > v1.10, you need to flash an elder OEM firmware release first, and only then flash OpenWrt; tested and works 🙏 [credits go to sayboon](#) for his tutorial

## Flash Layout

Please read the article [Flash Layout](#) for a better understanding. It contains a couple of explanations. Then let's have a quick view at flash layout of this particular device:

TP-Link WR1043ND Flash Layout stock firmware					
<b>Layer0</b>	m25p80 spi0.0: m25p64 8192KiB				
<b>Layer1</b>	mtd0 <b>u-boot</b> 128KiB	mtd1 <b>firmware</b> 8000KiB		mtd3 <b>art</b> 64KiB	
<b>mountpoint</b>	none	/		none	
<b>filesystem</b>	none	SquashFS		none	
TP-Link WR1043ND Flash Layout					
<b>Layer0</b>	m25p80 wspi0.0: m25p64 8192KiB				
<b>Layer1</b>	mtd0 <b>u-boot</b> 128KiB	mtd5 <b>firmware</b> 8000KiB		mtd4 <b>art</b> 64KiB	
<b>Layer2</b>		mtd1 <b>kernel</b> 1280KiB	mtd2 <b>rootfs</b> 6720KiB		
<b>mountpoint</b>			/		
<b>filesystem</b>			<a href="#">overlayfs</a>		
<b>Layer3</b>			1536KiB	mtd3 <b>rootfs_data</b> 5184KiB	
<b>mountpoint</b>	none	none	<a href="#">/rom</a>	<a href="#">/overlay</a>	none
<b>filesystem</b>	none	none	<a href="#">SquashFS</a>	<a href="#">JFFS2</a>	none

## Installation, Konfiguration, erste Schritte

- Im Verzeichnis `ar71xx/generic` die Datei `openwrt-ar71xx-generic-tl-wr1043nd-v1-squashfs-factory.bin` herunterladen. Der Teilstring `factory` weist darauf hin, daß es hier um den Update von der Hersteller-Firmware nach Openwrt geht – also um die Erstinstallation. Für spätere Updates, wenn Openwrt schon installiert ist, verwendet man stattdessen `openwrt-ar71xx-generic-tl-wr1043nd-v1-squashfs-sysupgrade.bin` (**also** `sysupgrade` **statt** `factory`).
- Die Installation wird einfach per Browser mit der Funktion Firmware Upgrade vorgenommen. Danach erfolgt ein Reboot, der Router hat danach die IP-Adresse `192.168.1.1/24` und ist unter ihr per `http` mit `root` ohne Passwort erreichbar. Als erstes muß jetzt ein neues Passwort für `root` vergeben werden (nächste Seite).
- Achtung: Man muß noch auf die HW-Revision achten. Neuere Geräte muß man mit alter Firmware „austricksen“.

# Root Passwort, SSH, Zertifikate

OpenWrt | OpenWrt Backfire 10.03.1 | Load: 0.00 0.00 0.00

Status **System** Network Logout

System **Administration** Software Startup Scheduled Tasks LED Configuration Backup / Flash Firmware Reboot

Password successfully changed!

### Router Password

Changes the administrator password for accessing the device

Password

Confirmation

### SSH Access

Dropbear offers SSH network shell access and an integrated SCP server

#### Dropbear Instance

Interface  lan:  wan:  unspecified

Listen only on the given interface or, if unspecified, on all

Port  Specifies the listening port of this Dropbear instance

Password authentication  Allow SSH password authentication

Allow root logins with password  Allow the root user to login with password

Gateway ports  Allow remote hosts to connect to local SSH forwarded ports

#### SSH-Keys

Here you can paste public SSH-Keys (one per line) for SSH public-key authentication.

# Schnittstellen

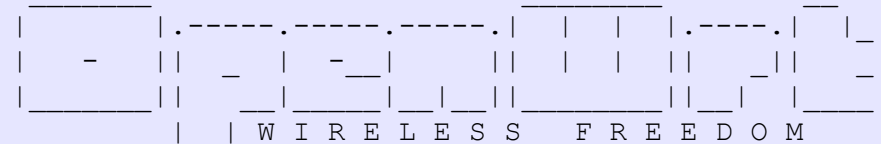
- Web (LuCI)
  - Zunächst nur HTTP
  - SSL nachinstallieren sobald Router am WAN

- SSH

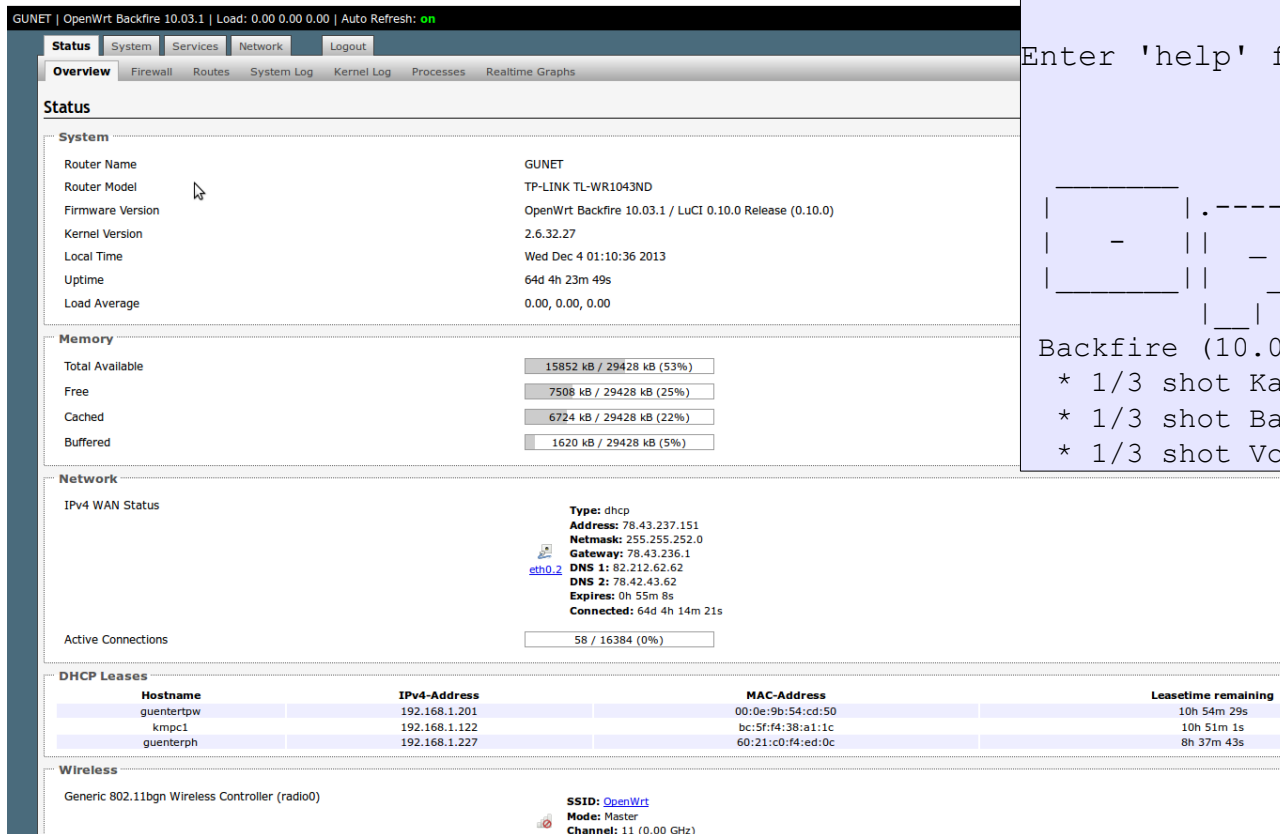
```
guenter@T42p:~$ ssh root@192.168.1.1
root@192.168.1.1's password:
```

```
BusyBox v1.15.3 (2011-11-24 00:44:20
CET) built-in shell (ash)
```

```
Enter 'help' for a list of built-in commands.
```



```
Backfire (10.03.1, r29592) -----
* 1/3 shot Kahlua   In a shot glass, layer Kahlua
* 1/3 shot Bailey's on the bottom, then Bailey's,
* 1/3 shot Vodka   then Vodka.
```



The screenshot shows the LuCI web interface for an OpenWrt router. The top navigation bar includes 'Status', 'System', 'Services', 'Network', and 'Logout'. The 'Overview' tab is active, showing various system metrics.

**Status**

- System**
  - Router Name: GUNET
  - Router Model: TP-LINK TL-WR1043ND
  - Firmware Version: OpenWrt Backfire 10.03.1 / LuCI 0.10.0 Release (0.10.0)
  - Kernel Version: 2.6.32.27
  - Local Time: Wed Dec 4 01:10:36 2013
  - Uptime: 64d 4h 23m 49s
  - Load Average: 0.00, 0.00, 0.00
- Memory**
  - Total Available: 15852 kB / 29428 kB (53%)
  - Free: 7508 kB / 29428 kB (25%)
  - Cached: 6724 kB / 29428 kB (22%)
  - Buffered: 1620 kB / 29428 kB (5%)
- Network**
  - IPv4 WAN Status:
    - Type: dhcp
    - Address: 78.43.237.151
    - Netmask: 255.255.252.0
    - Gateway: 78.43.236.1
    - DNS 1: 82.212.62.62
    - DNS 2: 78.42.43.62
    - Expires: 0h 55m 8s
    - Connected: 64d 4h 14m 21s
  - Active Connections: 58 / 16384 (0%)
- DHCP Leases**

Hostname	IPv4-Address	MAC-Address	Leasetime remaining
guentertpw	192.168.1.201	00:0e:9b:54:cd:50	10h 54m 29s
kmpr1	192.168.1.122	bc:5f:f4:38:a1:1c	10h 51m 1s
guenterph	192.168.1.227	60:21:c0:f4:ed:0c	8h 37m 43s
- Wireless**
  - Generic 802.11bgn Wireless Controller (radio0)
    - SSID: OpenWrt
    - Mode: Master
    - Channel: 11 (0.00 GHz)

# Konfiguration

- OpenWRT ist ein Linux, aber ohne GUI, d.h. Textkonsole (per SSH) und Web Interface.
- Viele Pakete heißen anders als gewohnt, z.B. Paketmanager `opkg`. Damit er überhaupt irgendwelche Pakete findet, muß man zunächst `opkg update` durchführen.

```
root@OpenWrt:~# opkg update
Downloading http://downloads.openwrt.org/backfire/10.03.1/ar71xx/packages/Packages.gz.
Inflating http://downloads.openwrt.org/backfire/10.03.1/ar71xx/packages/Packages.gz.
Updated list of available packages in /var/opkg-lists/packages.
root@OpenWrt:~#
```

- Pakete installieren mit `opkg install` (hier: HTTPS). Abhängigkeiten werden aufgelöst. Konfigurationsdateien meist in `/etc/config`

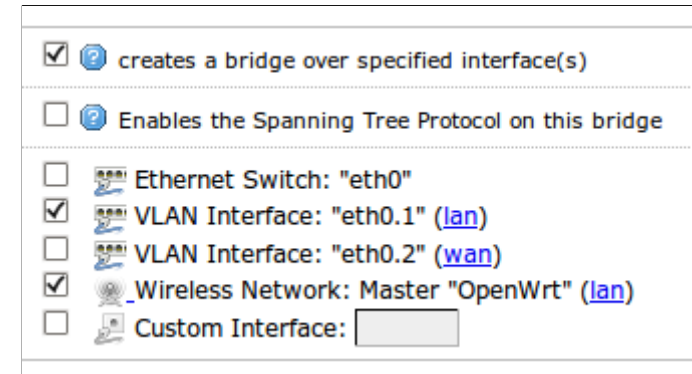
```
root@OpenWrt:~# opkg install uhttpd-mod-tls
Installing uhttpd-mod-tls (28) to root...
Downloading http://downloads.openwrt.org/backfire/10.03.1/ar71xx/packages/uhttpd-mod-tls_28_ar71xx.ipk
Installing libcyassl (1.4.0-2) to root...
Downloading http://downloads.openwrt.org/backfire/10.03.1/ar71xx/packages/libcyassl_1.4.0-2_ar71xx.ipk
Configuring libcyassl.
Configuring uhttpd-mod-tls.
root@OpenWrt:~# opkg install luci-ssl
...
Configuring px5g.
Configuring luci-ssl.
root@OpenWrt:~#
```



# Netzwerk-Übersicht, Firewall

- Netzwerk Interfaces

- 1 WLAN, 4 LAN (1-4), 1 WAN(0)
- VLANs (virtuelle LANs): lan, wan
- br-lan (LAN+WLAN), eth0 (LAN+WAN), WLAN

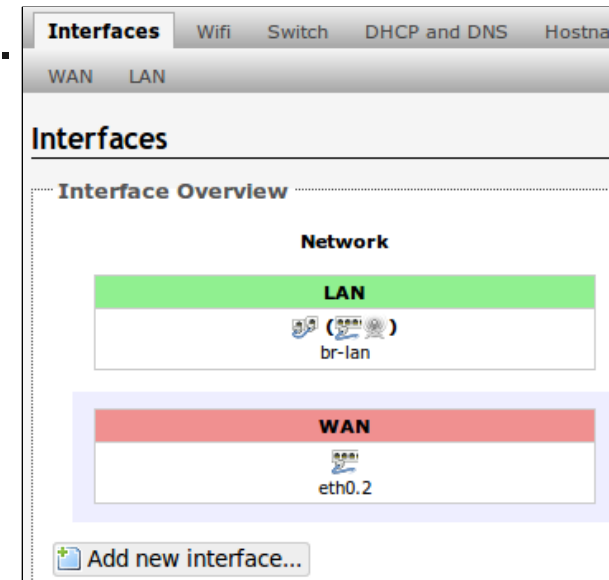


- Firewall:

- per Default aktiv und hat die gängigen Regeln, d.h. von außen nach innen ist (fast) nichts erlaubt, von innen nach außen alles.
- Zonen: LAN, WAN

Zones		Zone ⇒ Forwardings	Input	Output	Forward	Masquerading	MSS clamping	
lan:	lan:	⇒ wan	accept	accept	reject	<input type="checkbox"/>	<input type="checkbox"/>	
wan:	wan:	⇒ REJECT	reject	accept	reject	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Add



## Zugriffe von außen

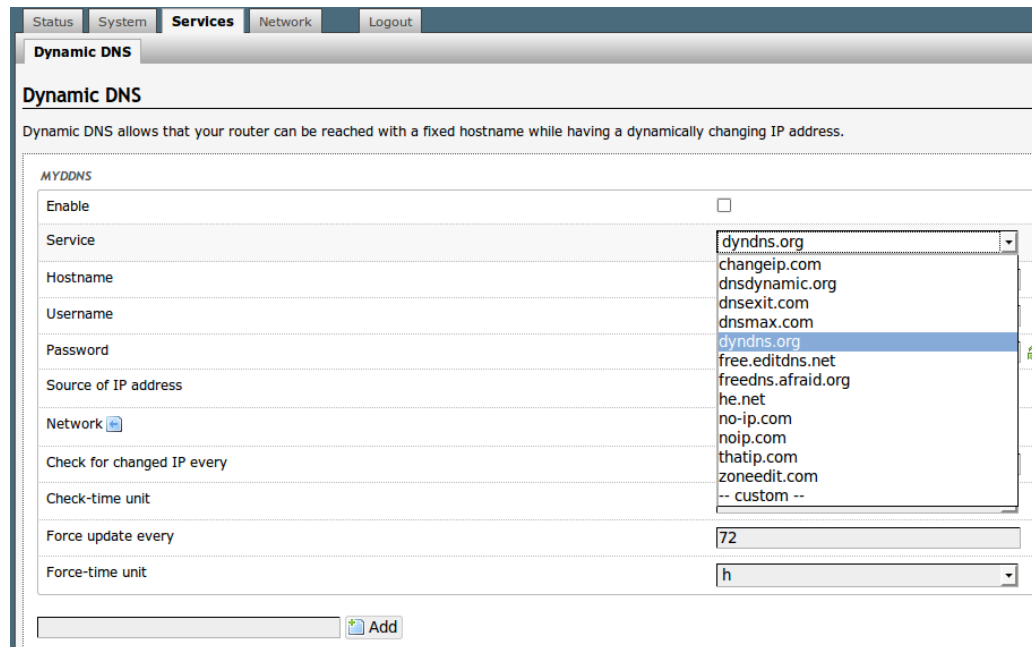
- Dynamisches DNS einrichten
- 3 Sicherungsebenen
  - Port Knocking: Verhindert eine Reaktion auf einen Scan durch Hacker und andere (passiert ständig im Internet)
  - Ändern SSH-Port: Erschwert Angriffe durch Probieren (auch von innen, durch Gäste)
  - SSH nur mit Zertifikat, also nicht mit Userid/Passwort: die eigentliche Sicherheitsmaßnahme (vorherige Registrierung erforderlich, kein Passwort geht übers Netz)

## Dynamisches DNS (IP4)

- Typischer Internetanschluß hat keine feste IP-Adresse.
- Es wird ein fester DNS-Name gebraucht, der immer auf die jeweils richtige IP zeigt.
- Dafür braucht man
  - Service (DynDNS nicht mehr kostenlos)
    - Entscheidung für **no-ip**
  - Unterstützung im Router (Client)
    - Paket `luci-app-ddns` nachinstallieren (auch mit LuCI möglich)

# Dynamisches DNS (Forts.)

- Der Reiter Dynamic DNS in LuCI entsteht automatisch nach Installation von `luci-app-ddns`.



The screenshot shows the 'Dynamic DNS' configuration page in LuCI. The 'Service' dropdown menu is open, displaying a list of providers including `dyndns.org`, `changeip.com`, `dnsexit.com`, `dnsmax.com`, `free.editdns.net`, `freedns.afraid.org`, `he.net`, `no-ip.com`, `noip.com`, `thatip.com`, `zoneedit.com`, and `-- custom --`. The `dyndns.org` option is currently selected.

Free DDNS

Use our Free Dynamic DNS to map a dynamic IP address, or long URL to an easy to remember hostname.

Limited Domain Choices

Up To 3 Hostnames

Dynamic DNS Updates

Hostnames Expire Every 30 Days

URL & Port 80 Redirects

Email Support

Free

Sign Up

# Port Knocking

- Beschreibung nur bei DD-WRT gefunden
  - Dies ist die einfachste Knocking-Variante (statisch)
- Lange Fehlersuche (mit iptables-Kenntnissen gelöst)
- Hier gezeigte Knocking-Sequenzen und SSH-Port sind Default → Ändern
- Resultat: Freigabe **eines** Ports für **eine** IP-Adresse

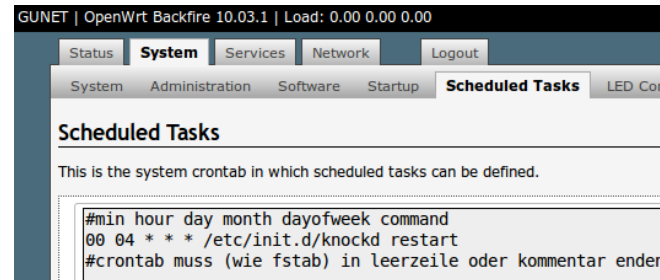
```
[options]
    logfile = /var/log/knockd.log

[openSSH]
    sequence      = 7000,8000,9000
    seq_timeout  = 10
    tcpflags     = syn
command      = /usr/sbin/iptables -A INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
    command      = /usr/sbin/iptables -I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT

[closeSSH]
    sequence      = 9000,8000,7000
    seq_timeout  = 10
    tcpflags     = syn
    command      = /usr/sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
```

# Port Knocking

- Installieren Paket knockd (Server)
- Anpassen Datei /etc/knockd.conf
  - Knocking-Sequenz ändern
  - iptables Befehl anpassen (iptables ist die Firewall)
  - Autostart via init-Skript

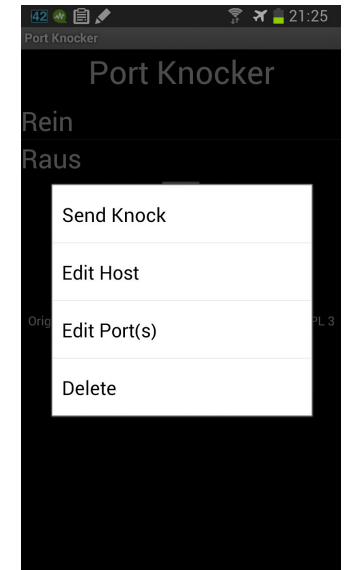


- Sporadische Probleme → Jede Nacht Restart
- Bereinigung von iptables („Leichen“)

```
/usr/sbin/iptables -D INPUT
-s 195.212.29.187
-p tcp --dport 22 -j ACCEPT
```

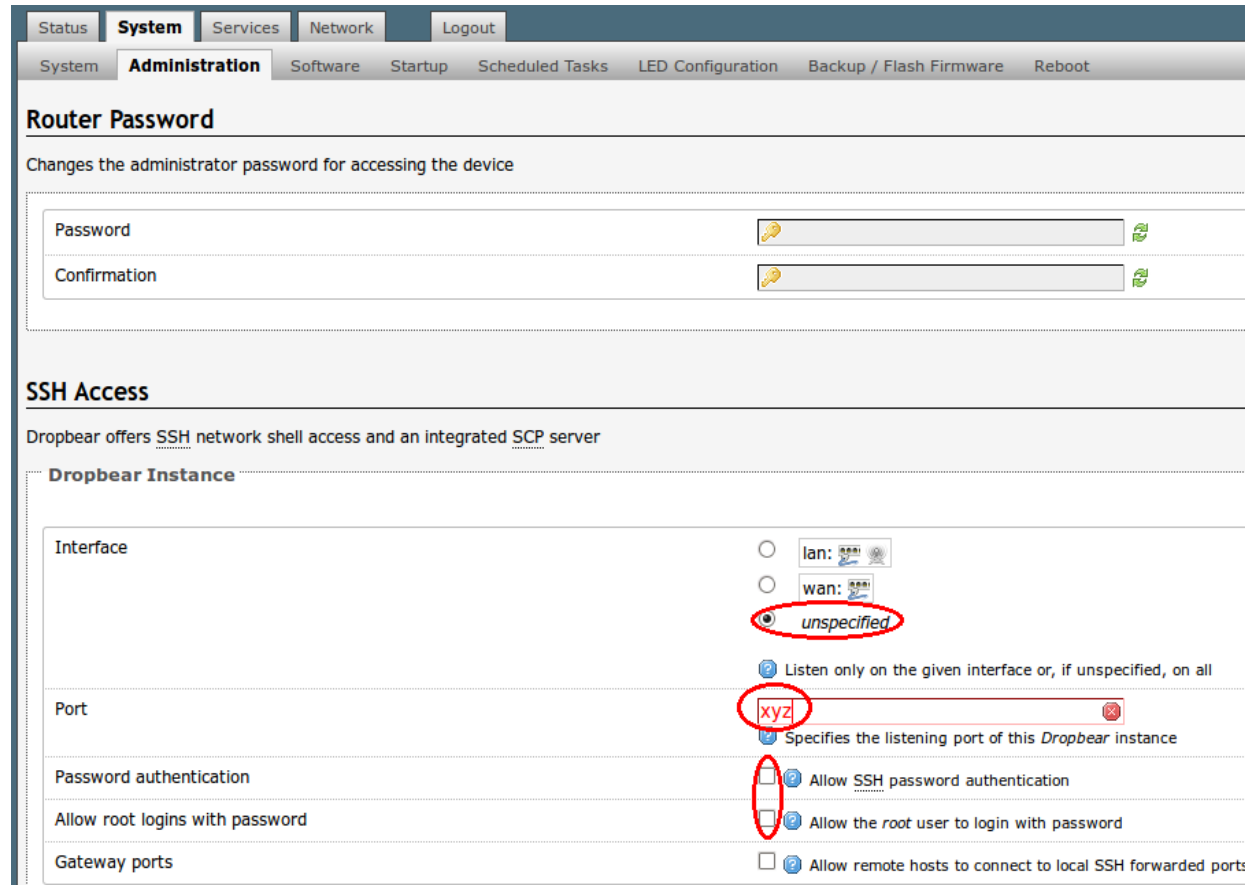
- Knocking Client installieren/einrichten

- Linux: Hackertool netcat `for x in 111 222 333; do nc -w 1 -z 192.168.1.1 $x; done`
- Windows: dito <http://linhost.info/2011/05/netcat-for-windows/>
- Android: Port Kocker (Google Play Store)



# Ändern SSH-Port

- Der SSH Server heißt Dropbear.
- An dieser Stelle noch „password authentication“ behalten
- Kann auch separat für WAN und LAN eingestellt werden



The screenshot shows the OpenWRT web interface with the following configuration for SSH Access:

- Router Password:** Fields for Password and Confirmation.
- SSH Access:** Dropbear offers SSH network shell access and an integrated SCP server.
- Dropbear Instance:**
  - Interface:** Radio buttons for lan, wan, and unspecified (selected and circled in red).
  - Port:** Text input field containing 'xyz' (circled in red).
  - Password authentication:** Checked checkbox (circled in red).
  - Allow root logins with password:** Checked checkbox (circled in red).
  - Gateway ports:** Unchecked checkbox.

# SSH mit Zertifikat

- Schritte pro Client:
  - Erzeugen Public/Private Key Pair
    - Dabei Passphrase für spätere Zugriffe festlegen
  - Übertragen des Public Key an den Router
  - Auf dem Router den Key als berechtigten Client-Key eintragen
  - Testen – beim Login wird (grafisch) nach der vorher festgelegten Passphrase gefragt, es geht kein Passwort über die Leitung.
- Android Client: App „VX ConnectBot“

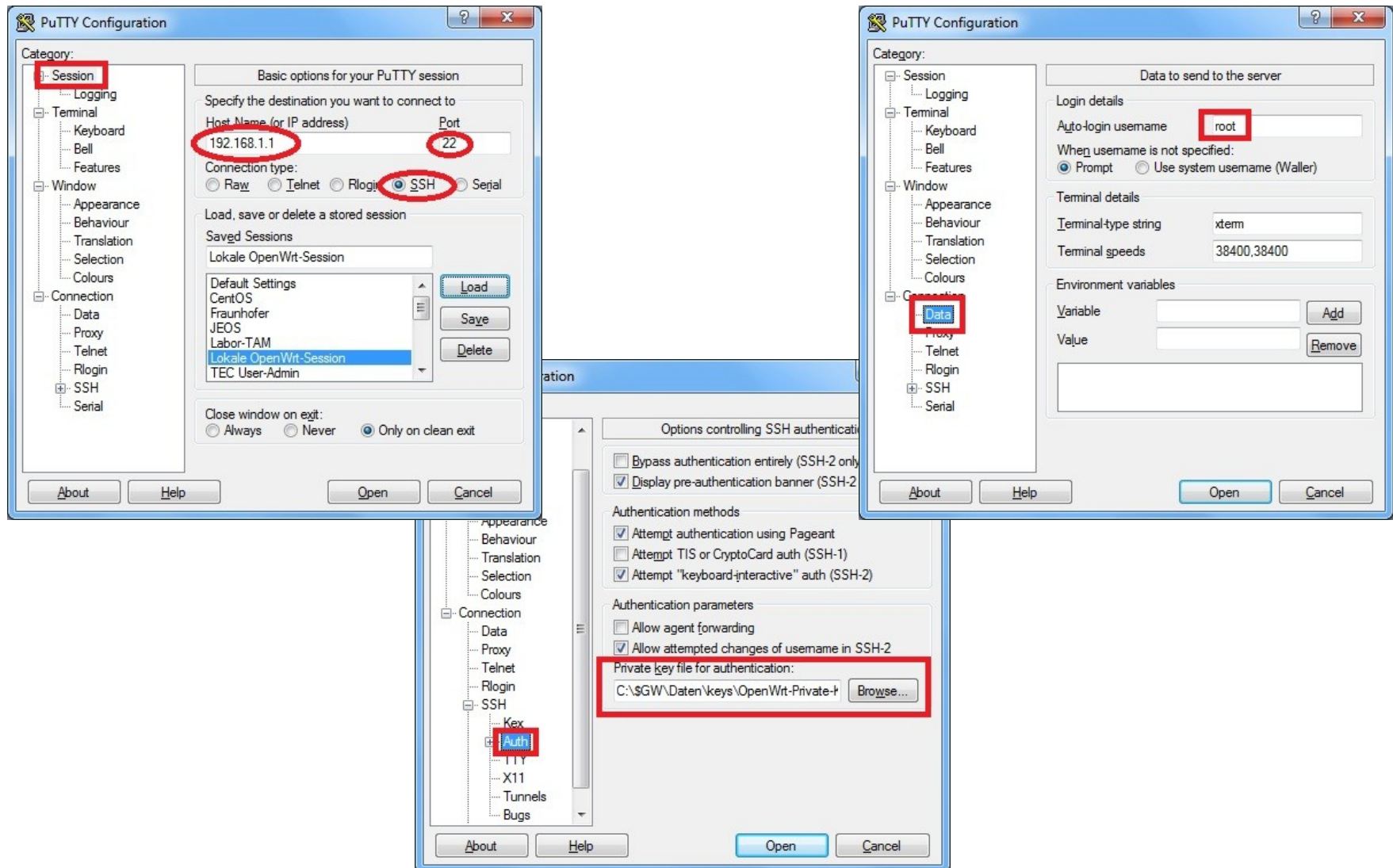






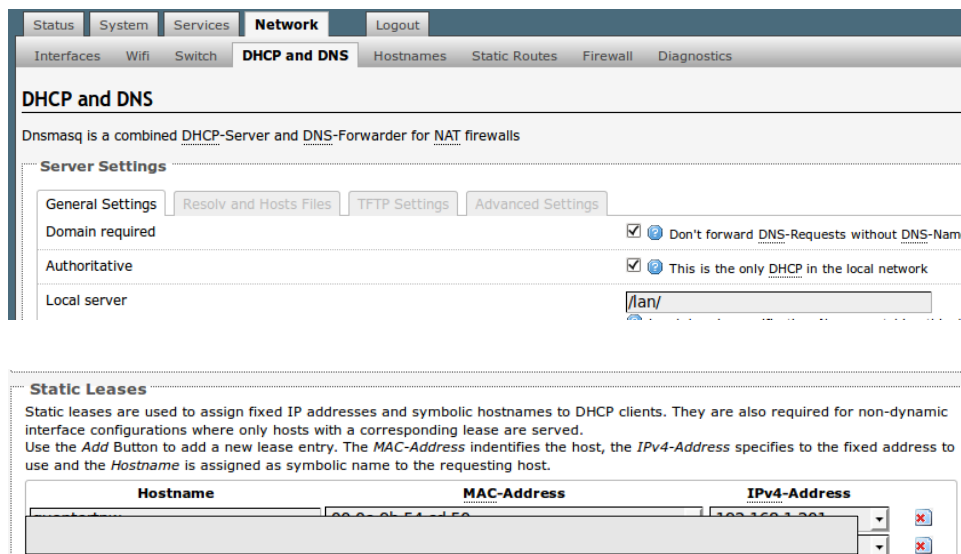
# SSH mit Zertifikat (Windows Client - Putty)

- Konfigurieren einer Verbindung mit Putty



# DHCP, DNS

- Ziel: Jeder Client im Haus soll immer die gleiche IP erhalten. Trotzdem soll DHCP verwendet werden, damit die Geräte auch anderswo funktionieren. Lösung: **Statisches Leasing auf Basis der MAC-Adresse**. Merkbare Namen im DNS-Server.
  - Nummernplan, getrennt nach WLAN und Ethernet
  - Aktuell 27 Einträge (MAC-Adressen)



The screenshot shows the 'DHCP and DNS' configuration page in OpenWRT. It includes sections for 'Server Settings' and 'Static Leases'.

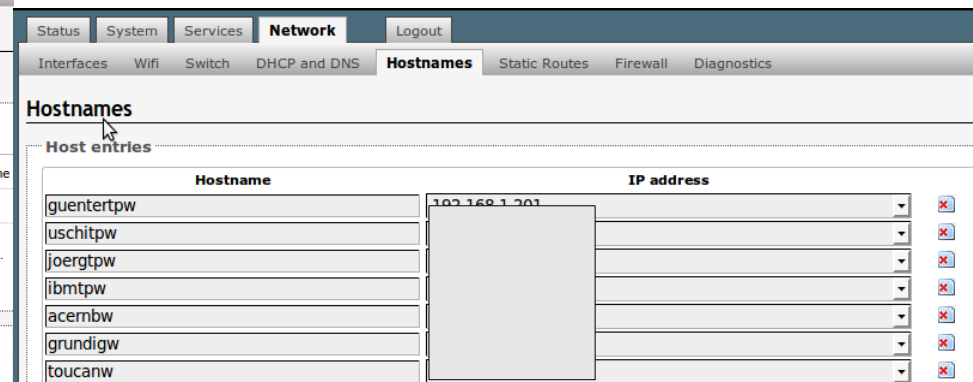
**Server Settings:**

- General Settings: Resolv and Hosts Files, TFTP Settings, Advanced Settings
- Domain required:  Don't forward DNS-Requests without DNS-Name
- Authoritative:  This is the only DHCP in the local network
- Local server: /lan/

**Static Leases:**

Static leases are used to assign fixed IP addresses and symbolic hostnames to DHCP clients. They are also required for non-dynamic interface configurations where only hosts with a corresponding lease are served. Use the Add Button to add a new lease entry. The MAC-Address identifies the host, the IPv4-Address specifies to the fixed address to use and the Hostname is assigned as symbolic name to the requesting host.

Hostname	MAC-Address	IPv4-Address
	00:02:54:00:15:00	192.168.1.201



The screenshot shows the 'Hostnames' configuration page in OpenWRT, displaying a list of host entries.

Hostname	IP address
guentertpw	192.168.1.201
uschitpw	
joertgpw	
ibmtpw	
acernbw	
grundigw	
toucanw	

# Wake on LAN

- Starten eines Rechners über die Netzwerkkarte durch Zusenden eines „Magic Packet“ an die MAC-Adresse
  - Lokal:
    - Linux 

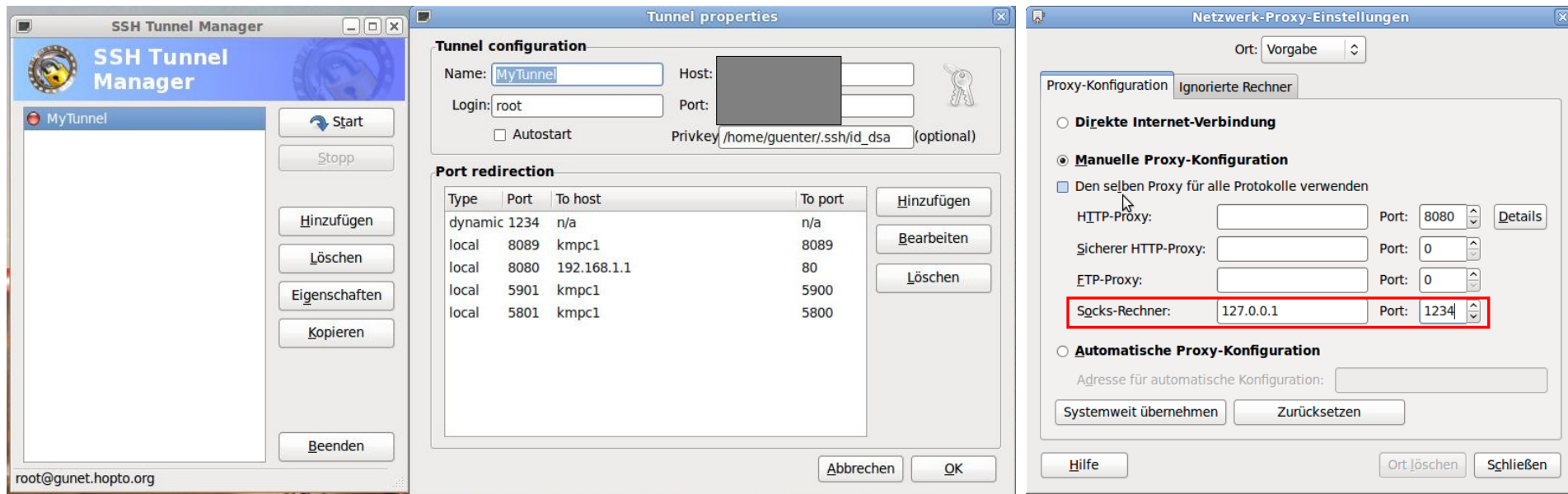
```
#wakeonlan aa:bb:cc:dd:ee:ff
```
    - Windows: [http://www.chip.de/downloads/WakeOnLan-WOL\\_51880639.html](http://www.chip.de/downloads/WakeOnLan-WOL_51880639.html)
    - Android: App WOL im Playstore
  - Remote via SSH auf dem Router:
    - Paket `wol` nachinstallieren
    - ```
wol -h <Subnetz> <MAC-Adresse Zielrechner>  
root@GUNET:~#wol -h 192.168.1.255 aa:bb:cc:dd:ee:ff
```

# Port Forwarding („VPN für Arme“)

- Es soll über die SSH-Verbindung ein Port auf einem Rechner im LAN erreicht werden. Wir machen auf der SSH-Verbindung einen local port redirect.
- Beispiel (Linux): 

```
guenter@T42p:~$ ssh -L 0.0.0.0:12345:192.168.1.184:8080 -p 22 root@192.168.1.1
```

  - Ohne 0.0.0.0 geht der Zugriff nur über localhost:12345 oder 127.0.0.1:12345. Dies ist eine Eigenheit des SSH Daemon dropbear.
  - Der lokale Port 12345 wird umgeleitet auf 192.168.1.184:8080 (auch DNS-Name)
  - Die SSH Verbindung zum Router geht wie gewohnt über root (auch via no-ip).
  - Der SSH-Port ist bei mir geändert
- Komfortables Tool: gSTM (Gnome SSH Tunnel Manager):
  - Mehrfache Redirects, dynamic macht Router zum Socks-Proxy



The image shows three screenshots of software interfaces related to SSH tunneling and proxy configuration.

**SSH Tunnel Manager:** A window titled "SSH Tunnel Manager" showing a list of tunnels. One tunnel named "MyTunnel" is visible. Buttons for "Start", "Stopp", "Hinzufügen", "Löschen", "Eigenschaften", "Kopieren", and "Beenden" are present.

**Tunnel properties:** A window titled "Tunnel properties" showing configuration for "MyTunnel". Fields include Name, Host, Login (root), Port, and Privkey (/home/guenter/.ssh/id\_dsa). A "Port redirection" table is shown below:

| Type    | Port | To host     | To port |
|---------|------|-------------|---------|
| dynamic | 1234 | n/a         | n/a     |
| local   | 8089 | kmpc1       | 8089    |
| local   | 8080 | 192.168.1.1 | 80      |
| local   | 5901 | kmpc1       | 5900    |
| local   | 5801 | kmpc1       | 5800    |

**Netzwerk-Proxy-Einstellungen:** A window titled "Netzwerk-Proxy-Einstellungen" showing proxy configuration. The "Manuelle Proxy-Konfiguration" section is selected. The "Socks-Rechner" field is highlighted with a red box, showing "127.0.0.1" and "Port: 1234".

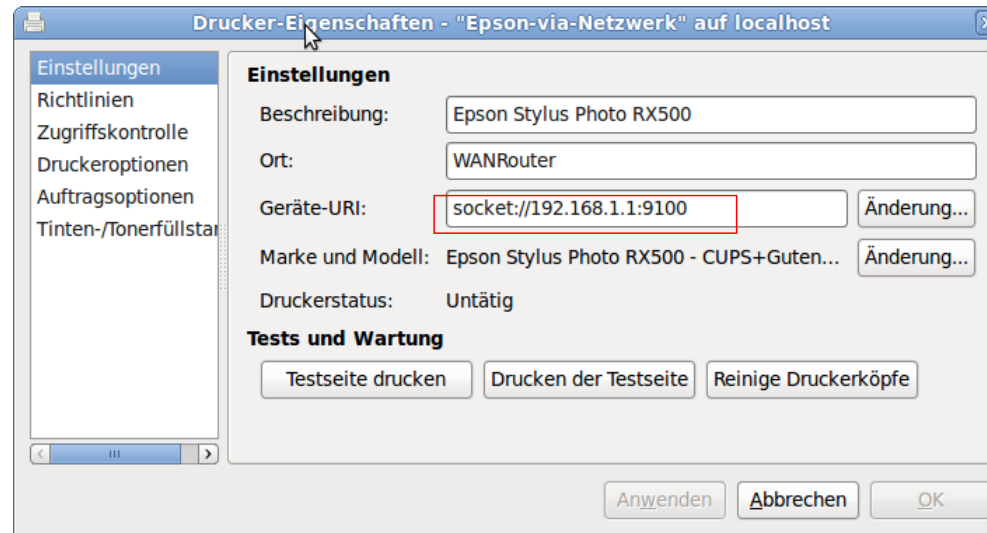
# Netzwerkdrucker am USB-Port

- Der TP-Link-Router hat einen USB-Port. Man kann hier entweder externe Speichermedien oder einen Drucker anschließen. 3 Ansätze:
  - Samba – absoluter Overkill, denn hier geht es um Freigabe von Speicher und Drucker.
  - CUPS – reine Druckerfreigabe, aber es wird zuerst der komplette Druckjob gespooled und dann geht das Drucken los. Wegen Ressourcenverbrauch bei großen Druckjobs zu riskant.
  - p910nd – bei dieser Variante wird der Drucker mit dem druckenden Client „kurzgeschlossen“, d.h. der Server/Router wird wenig belastet, reicht nur Daten durch.
    - Pakete: USB-Basissupport (`kmod-usb-core`, `kmod-usb-ohci`, `kmod-usb2`), `usbutils`, USB-Drucker-Support (`kmod-usb-printer`), Druckerserver (`p910nd`)
    - In Konfigurationsdatei `/etc/config/p910nd` den Schalter `enabled` auf „1“ setzen
    - Firewallregel für Port 9100 in `/etc/config/firewall` einfügen.
    - Das durch die Installation entstandene Init-Skript aktivieren:

```
#/etc/init.d/p910nd enable
```

# Netzwerkdrucker: Clients

- Linux



- Windows

- Unterschied je nach Version

- Nachteile

- Keine Statusinformationen (Füllgrad Tinte)
- Kein Scannersupport (→ Kabel Umstecken)





# Netzwerkdrucker – Windows Client

- Windows 7:  
**Devices and Printers**  
**Add a printer**  
**Add a local printer** → Wizard  
**Create a new port**  
Standard TCP/IP port | **Next**  
Hostname or IP address:  
**<Router-Adresse ohne Port>**  
Port name: **<selbst vergeben>**  
Query the printer and automatically select the driver to use **<deselektieren>** | **Next**  
<Fehlermeldung: Additional port information required>  
**Device type Custom | Settings...**  
Protocol **Raw**  
Raw Settings Port Number **9100**  
**Richtigen Printer Driver wählen**  
**Wizard beenden**

- Windows XP (Vorsicht, Hölle!)  
Nicht wie in der Beschreibung von OpenWRT/p910nd vorgehen und in einem bestehenden Drucker einen weiteren (IP) Port hinzufügen und diesem Netzanbindung beibringen.  
**Füge einen neuen Drucker hinzu.**  
control panel/printer settings → **printer properties**  
Ports (Tab) → **Add Port**  
Standard TCP/IP Port → **New Port...**  
<Dem Wizard folgen>  
Printer Name or IP Address: **<Router-Adresse ohne Port>**  
<evtl. Fehlermeldung ignorieren>  
**Device type Custom | Settings...**  
Protocol **Raw**  
Raw Settings Port Number **9100**  
usw.

# Zeitsteuerung

- Es gibt die ganz normale crontab Umgebung. Voraussetzung: crond ist vorhanden und wird automatisch gestartet. Sonst nachinstallieren.
- Konfiguration über `/etc/crontabs/root` oder LuCI.
- Der crond Daemon muß nach einer Veränderung der crontab durchgestartet werden. Befehle dazu: `killall crond; /etc/init.d/cron start`

- Die crontab kann mit LuCI im Browser editiert werden.
- Aktivieren per Submit Button.

The screenshot shows the LuCI web interface for configuring scheduled tasks. The main content area is titled 'Scheduled Tasks' and contains the following text:

```
#min hour day month dayofweek command
00 04 * * * /etc/init.d/knockd restart
00 10 07 12 * /home/guenter/wecken.sh
05 20 10 12 * /home/guenter/wecken.sh
02 13 12 12 * /home/guenter/wecken.sh
#crontab muss (wie fstab) in leerzeile oder kommentar enden
```

At the bottom of the interface, there are two buttons: 'Reset' (with a red 'x' icon) and 'Submit' (with a green checkmark icon).

## Zeitgesteuertes Wecken

- So kann man die Features zu neuen Lösungen kombinieren:
- Aufnahmen mit TV-Karte am PC.
  - Problem: Aufwecken durch die TV-SW führt zu häufigem unerwünschten Hochfahren.
  - Lösung: Aufwecken von außen durch den Router. Seither kein unerwünschtes Hochfahren mehr beobachtet.
    - Das Skript `wecken.sh` ist ein Einzeiler und besteht lediglich aus dem bereits gesehenen Befehl `wol`.