Bye, bye Plastikrouter

Mein Router hört auf mein Kommando

Alexander Schreiber <als@thangorodrim.ch>

http://www.thangorodrim.ch/

Chemnitzer Linux-Tage 2016, 2016-03-20

Es sind nicht die Rechner, die den Admin machen. Es ist die Geisteshaltung.

– Werner Olschewski <w.olschewski@telesens.de> in d.a.s.r.

Inhalt

Über den Autor

- beschäftigt sich seit fast 20 Jahren mit Linux
- tätig als Systemingenieur bei YouTube in Zürich
- hat den einen oder anderen Computer im Haus

Um was geht es?

- Plastikrouter
  - Was ist ein Plastikrouter?
  - Warum ist das ein Problem?
- Die Alternativen
  - Hardware
  - Software

Was ist ein Plastikrouter?

The good ...

- kleine, kompakte Plastikkistchen
- geringer Stromverbrauch, leise (kein Lüfter)
- oft vom Provider gestellt == "kostenlos"
- meist ausreichend in Fähigkeiten
- vorkonfiguriert/minimale Konfiguration nötig
- minimales Wissen notwendig zum Einsatz
- Hinstellen, Kabel dran, Zugangsdaten einstellen, fertig
- $\Longrightarrow$ Alles prima, oder?

. . . the bad . . .

- Fähigkeiten auf Heimanwender ausgelegt
  - Basis: Web, Videostreams (passiv), Onlinespiele
  - POTS-Ersatz: VoIP-Telefonie, evtl. sogar analoge Endgeräte
  - Bonus: Speicher- & Mediafunktionen
- erweitere Fähigkeiten eher dünn gesät
  - VPN-Anbindung
  - remote-Login (ssh)
  - direkter Zugriff auf Paketfilterregeln

. . . and the ugly

- Sicherheitslücken im Grosspaket
- diverse Dienste in Betrieb $\longrightarrow$ Angriffspunkte
- alte Softwareversionen, keine Sicherheitsupdates
- Hintertüren mit hart codierten Zugangsdaten
- default Logindaten & remote Zugang aktiv

Kleine Horrorgalerie

- 13000 Asus RT-N66 gehackt $\longrightarrow$ 20 Jahre Codeaudits (FCC)
- FritzBox web interface code injection
- mehrere Router mit Sercomm modem: Hintertür, “Sicherheitsupdate” versteckt die Hintertür
- MikroTik RouterOS: default User “admin”, leeres Passwort
- Fortinet: ssh-Zugang mit festverdrahtetem Passwort
- Juniper: festverdrahtetes master Passwort for ScreenOS
- DLink: CSRF im Web-Interface
- Ubiquity Networks: ISPs liefern Geräte mit aktivem Remotezugang und default Passwort
- http://routersecurity.org/bugs.php

Alles kaputt?

- Ab Werk ja, leider.
- Aber: Rettung ist möglich.
  - Auswahl guter Hardware ...
  - ... mit Unterstützung durch OpenSource Plattformen
  - Ersetzen der Werksfirmware mit OpenSource
  - $\longrightarrow$ Gewinn an Sicherheit, Stabilität und Features.

Zieldefinition, oder: was wollen wir eigentlich?

- robusten, zuverlässigen Internet-Router
- robusten, zuverlässigen Wireless-AP
- aktuelle Systemsoftware & Kernel, mit security updates
- sichere Plattform, Schutz gegen drive-by Angriffe
- flexible, leistungsfähige Plattform
- spezielle Dienste: transproxy, web-filter, VPN, ...

Die Alternativen

- Alternative Lösungen auf Hardware- und Softwareseite
- Hardware:
  - Hardware an sich meist unproblematisch
  - $\longrightarrow$ Einsatz vorhandener Hardware (Router, AP) evtl. möglich
  - grosse Bandbreite an Hardwareplattformen verfügbar
- Software:
  - Werksfirmware problematisch $\longrightarrow$ ersetzen
  - Auswahl der Basis: Linux, (Free,Net,Open)BSD, Linux meist beste Hardwareunterstützung
  - falls Linux: normale Distribution oder spezialisierte Routerdistro?
- Kompatibilität Hardware & Software klären

Software - Anforderungen

- OpenSource (Linux, *BSD)
- Quellen wirklich verfügbar & verwendbar (selbst baubar)
- Sicherheitsupdates & Bugfixes erfolgen und sind schmerzarm
- idealerweise komplette Unix-Umgebung mit Router-Tools
- dazu Paketfilter, VPN, Shaping, ...

Software - Übersicht

- Linux Standarddistros: Debian, Slackware, Ubuntu, ...
- Linux spezielle Routerdistros: OpenWRT, DD-WRT, Tomato
- die BSDs: FreeBSD, NetBSD, OpenBSD
- **massive** Variation an Hardwareunterstützung & Features
- generell:
  - AccessPoint oder resourcenknapper Router: Linux Routerdistro
  - Router mit genug Kapazität: Linux Standarddistro
  - *BSD: kommt sehr auf den Einzelfall an
- Die gewünschte Softwareplattform diktiert Hardwareauswahl und umgekehrt!

Grundsätzliches

- Aufgabe: Internet-Router und Wireless-AP
- All-in-One oder Aufgabenteilung?
- All-in-One: deutlich weniger Plattformauswahl, eine Kiste
- Aufgabeteilung: sehr freie Plattformauswahl, aber 2 Kisten
- Muss Recycling sein oder ist Neubeschaffung ok?
- $\longrightarrow$ Aufgabenteilung mit Neubeschaffung

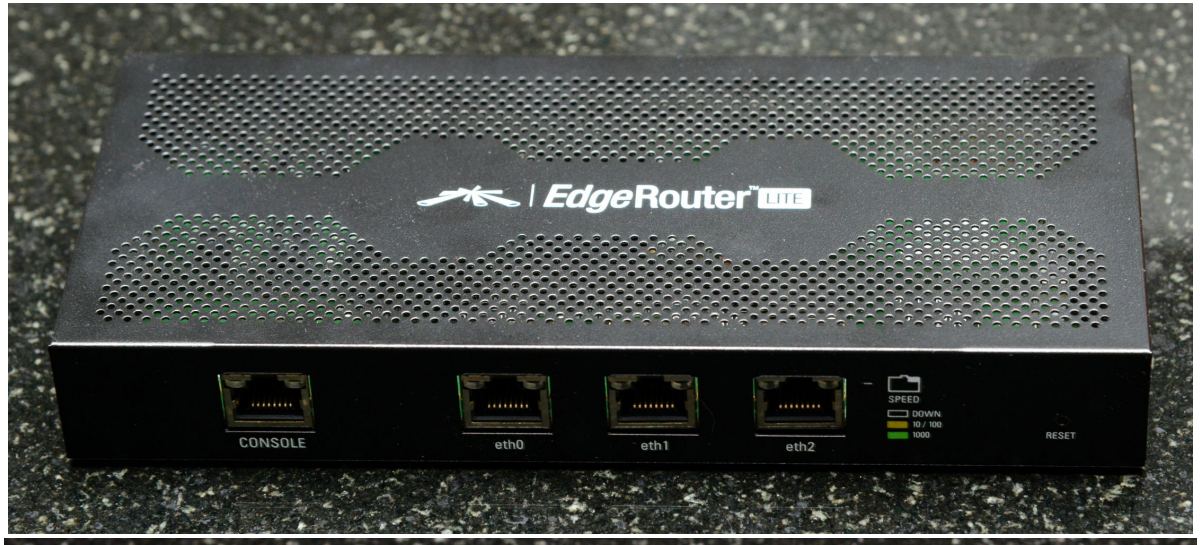
Hardware-Übersicht, Router

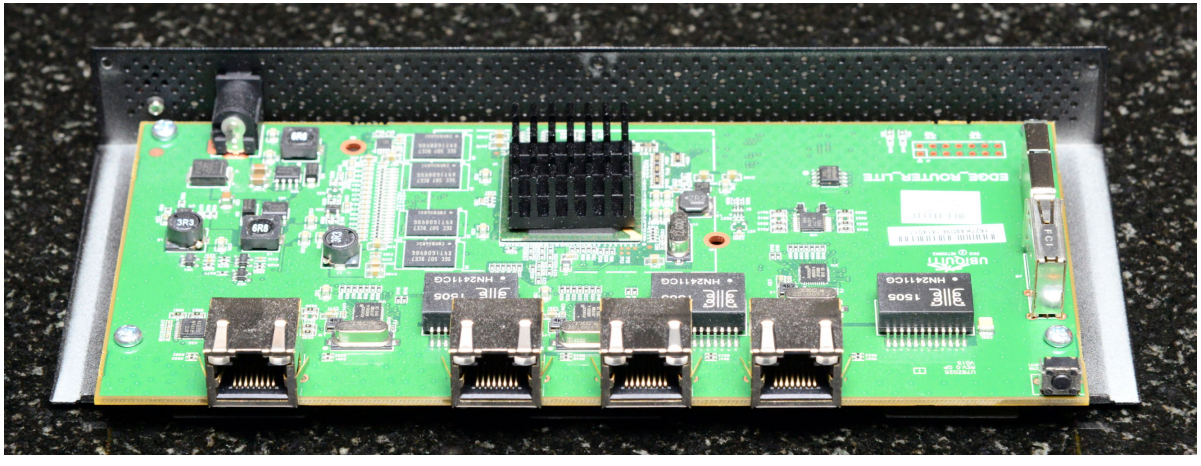
- Anforderungen:
  - min. 2, besser 3 NICs (separate, nicht via internem Switch)
  - RAM: min 256 MB, besser mehr
  - Storage: min 2 GB, besser mehr
  - CPU: hinreichend schnell für Uplink
- Grosse Auswahl:
  - vorhandener PC (plus NICs): einfach, stromhungrig, laut
  - dedizierte Router-Hardware:
    - x86: Alix (AMD Geode oder APU), Soekris (Intel), ...
    - !x86: Banana Pi BPI-R1, Edge Router Lite, ...
    - sowie zahlreiche fertige, befreibare Routerkistchen
  - nicht dedizierte Router-Hardware:
    - alles, was min. 2 NICs hat, besser 3 NICs

Hardware-Übersicht, Wireless-AP

- Anforderungen:
  - Frequenzbänder: 2.4 GHz, 5 GHz
  - WiFi-Standards: 801.11a/b/g/n/ac?
    - abhängig von erwarteten Clients
    - beeinflusst maximale Bandbreite
  - Reichweite
  - MIMO
- Preisklasse
- Stromversorgung (PoE)

Hardwareauswahl: Ubiquity EdgeRouter Lite 3





Übersicht Edgerouter Lite 3, ab Werk

- alias Erlite-3
- Hardware
  - Cavium Octeon chipset
  - dual core MIPS64 @500 MHz, 512 MB RAM, 3x GBit
  - Cisco style serial console @ 115200, h/w watchdog
  - Firmware läuft von internem USB-Stick
- Software
  - OS: Ubiquity EdgeOS, basiert auf Vyatta, basiert auf Debian
  - kernel: 2.6.32.13-UBNT
  - 2.6.32 EOL seit 2016-03-12, letzter 2.6.32.70
  - Debian-Version: squeeze, LTS EOL seit 2016-02-29

Was machen wir damit?

- soll Router mit Standarddistro werden, CPU/RAM ausreichend
- $\longrightarrow$ Debian als Wunschdistro
- Voraussetzungen prüfen:
  - Cavium Octeon von Debian MIPS offiziell unterstützt
  - Cavium Octeon von aktuellem Linuxkernel unterstützt
- Wie kommt Debian auf den Erlite-3?
- Da war doch was mit “interner USB-Stick”...

Bereit zum Entern

- Umbau via USB-Stick
- Problem: Erlite etwas wählerisch bei USB-Sticks
  - mitgelieferter OK
  - wenige andere akzeptiert (u.a. SanDisk Cruzer Fit 16GB)
  - viele andere nicht gefunden
  - Grund: uBoot initialisiert USB Hardware nicht komplett
  - nach NetBoot mit NetBSD/Linux jeder USB Stick erkannt
  - Kompatibilitätliste u.a. im Gentoo Wiki
- fangen mit mitgeliefertem Stick an (Verzeichnis /rootfs)
- Kernel 2.6.32 bootet jessie nicht, wheezy geht noch

Umbau, Teil 1/3

- `debootstrap --arch=mips --verbose --foreign wheezy rootfs http://ftp.us.debian.org/debian/`
- EdgeOS booten, login ubnt:ubnt, `sudo su -`
- `mkdir /mnt/work ; mount /dev/sda2 /mnt/work`
- `chroot /mnt/work`
- `distro=wheezy ; export LANG=C`
- `/debootstrap/debootstrap --second-stage`
- `passwd`
- `vi /etc/inittab`
- T0 getty @115200 aktivieren, andere gettys auskommentieren
- `echo amnesiac > /etc/hostname`
- chroot verlassen, runterfahren

Umbau, Teil 2/3

- Datenextraktion vom Ubiquity Stick:
  - UBNT Kernel (vmlinux.64) von 1. Partition
  - `rootfs` Verzeichnis von 2. Partition kopieren
  - squashfs.img von 2. Partition mounten
  - `lib/modules/2.6.32.13-UBNT` kopieren
- neuen Stick vorbereiten (Sandisk Cruzer Fit 16 GB):
  - Partition 1: 256 MB, VFAT, `/boot`
  - Partition 2: der Rest, ext2 (wir booten 2.6.32!), /
  - kernel auf 1. Partition
  - rootfs und Kernelmodule auf 2. Partition
- neuen Stick in Erlite-3

Umbau, Teil 3/3

- Booten:
  - Boot nach USB-Stick Erkennung mit ESC anhalten
  - `fatload usb 0 $loadaddr vmlinux.64`
  - `bootoctlinux $loadaddr root=/dev/sda2 rootdelay=15 rw`
- Basissetup
  - `modprobe octeon-ethernet`
  - `dhclient -v eth0`
  - `apt-get install ssh screen sudo`
  - `/etc/fstab`: sda2 → /, `/dev/sda1` → `/boot`
  - `mount -a`
  - root User anlegen, zur Gruppe sudo hinzufügen
  - `/etc/ssh/sshd_conf: PermitRootLogin without-password`
  - `/etc/network/interfaces`: eth0 mit auto & DHCP eintragen

Kernel bauen

- `apt-get install build-essential libncurses-dev bc`
- Quellen für Linux 4.4.x von http://www.kernel.org/
- Konfigurieren
- `make deb-pkg`
- linux-image .deb installieren
- `cp /boot/vmlinuz-* /boot/vmlinuz.64`
- `shutdown -r now`

Umbau, Abschlussarbeiten

- Upgrade auf jessie:
  - apt-source von wheezy auf jessie ändern
  - `apt-get update`
  - `apt-get install sysvinit-core systemd-shim`
  - `apt-get dist-upgrade`
- runterfahren, rootfs sichern
- rootfs mit ext4 formatieren
- rootfs zurückspielen, `/etc/fstab` anpassen
- rootfs mit `noatime` mounten
- $\longrightarrow$ Erlite-3 jetzt mit minimalem Debian jessie

Weiterführendes

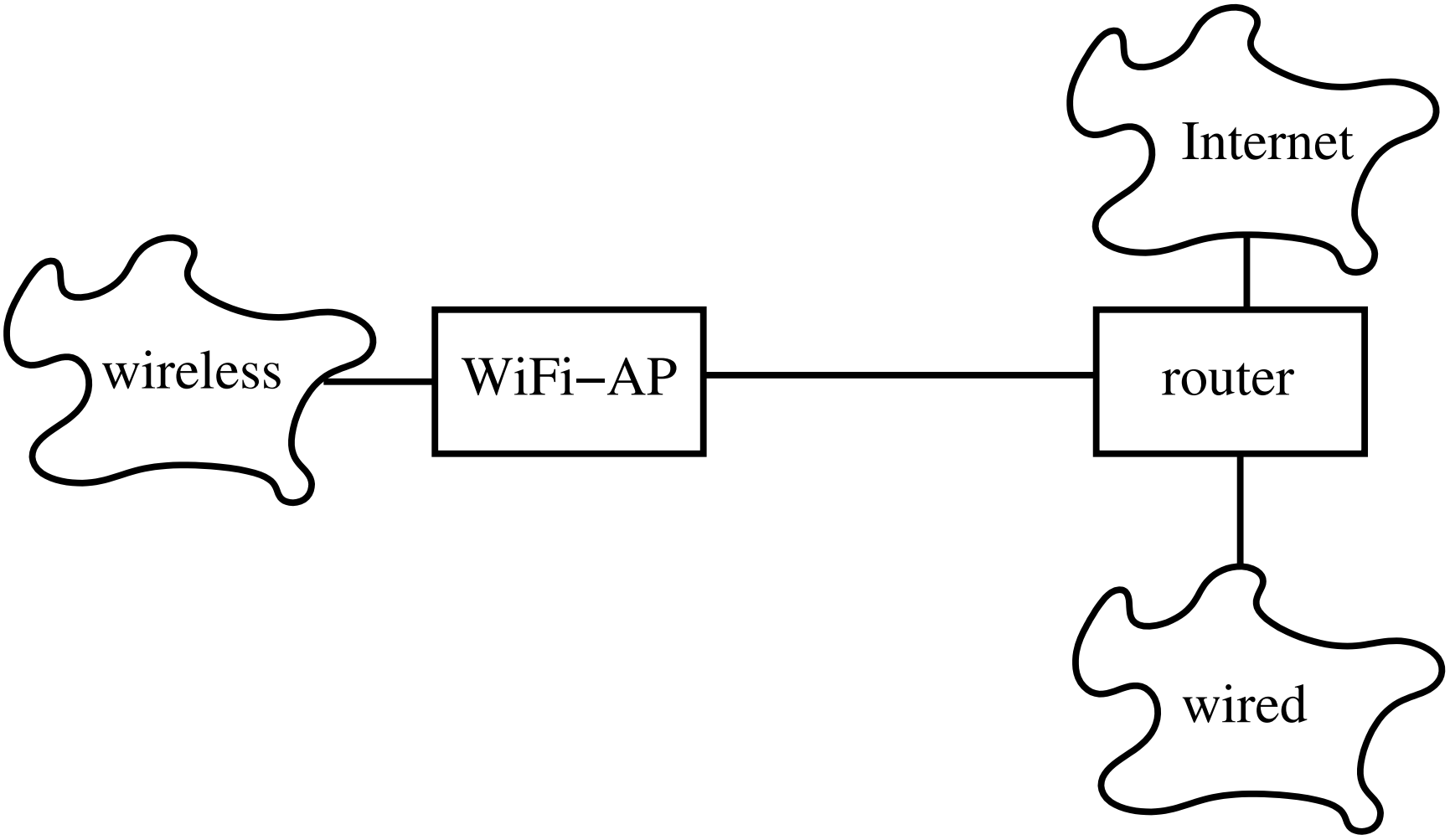
- logfiles auf tmpfs schreiben, logrotate schreibt auf Stick
- ssh auf Port $\neq 22 \rightarrow$ weniger Rauschen im Log
- DHCP-Server installieren (WiFi-AP an einem Port, Interface!)
- ggf. pppoe Komponenten installieren
- Paketfilter installieren und Filterregeln schreiben
- statische Adresse für internes Interface
- vernünftigen Hostnamen setzen
- Transproxy mit Filter?
- OpenVPN?
- ...

Hardwareauswahl: Ubiquity Unify UAP n300

WiFi-AP: Hardware und Setup

- Hardware: Ubiquity Networks Unify UAP n300
- Enterprise-AP
- 2.4 GHz, POE, LED-Ring, 2x2 MIMO, 100 MBit/s, max 4 W
- sehr gute Reichweite
- voll unterstützt von OpenWRT
- sehr einfach umzustellen:
  - http://www.openwrt.org/
  - Image download
  - Flashen via originale Web-UI des AP
  - reboot
- Konfigurieren als AP (via LuCI web ui), DHCP auf Router

Netzstruktur



Zusammenfassung

- Plastikrouter: ab Werk kaputt
- Abhilfe möglich:
  - Opensource hilft
  - Routerdistro oder Fullsize Distro möglich
  - grosse Bandbreite an Hardware
- $\longrightarrow$ sichere, stabile, aktuelle, offene Routerplattform
- Kernel, .config & Co. auf
  http://www.thangorodrim.ch/embedded/erlite-3/

Fragen?

Vielen Dank für Euer Interesse!