

NUT: Network UTility - ein Netzwerkmanager für Linux

Daniel Bahrtdt Stefan Bühler Oliver Groß
Betreuer: Dr. Boris Koldehofe

Abteilung Verteilte Systeme
Institut für Parallele und Verteilte Systeme
Fakultät 5: Informatik, Elektrotechnik und Informationstechnik
Universität Stuttgart



Motivation

- ▶ Konfiguration für ein Laptop in verschiedenen Arbeitsbereichen unpraktisch.
- ▶ Oft durch selbst zusammengehackte Skripte erleichtert, die meist aber mit “sudo” ausgeführt werden müssen.
- ▶ Umständliches Starten aller notwendigen Teile (WLAN Konfigurieren, IP zuweisen, VPN aufbauen)
- ▶ Unflexibel: Je nach Umgebung andere Konfiguration nötig

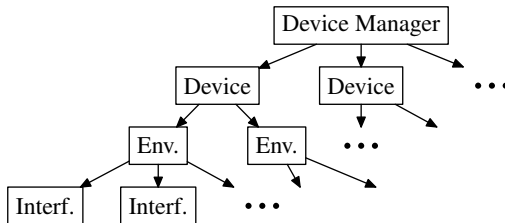
Motivation-Beispielskript

```
#!/bin/bash
echo "* loading modules...";
echo "  -> acer_acpi";
modprobe -r acer_acpi;
modprobe acer_acpi;
if [ "$1" = "bcm" ]
then
    echo "  -> bcm43xx";
    modprobe bcm43xx;
else
    echo "  -> ndiswrapper";
    modprobe ndiswrapper;
fi
echo "* enabling wlan hardware...";
echo "enabled: 1" > /proc/acpi/acer/wireless;
sleep 0.1;
if [ "$1" = "bcm" ]
then
    iwconfig wlan0 rate 11M;
fi
echo "* bringing up wlan interface...";
if [ "$2" = "connect" -o "$1" = "connect" ]
then
    ifup wlan0;
elif [ "$2" = "connect2" -o "$1" = "connect2" ]
then
    echo "* starting wpa supplicant...";
    wpa_supplicant -B -i wlan0 -D wext -c /etc/wpa_supplicant/wpa_supplicant.conf;
    sleep 1;
    echo "* starting dhcp client"
    dhclient wlan0
fi
```

Überblick NUT

- ▶ nuts: Daemon(Server) zur Verwaltung der Hardware
- ▶ libnutclient: Abstraktion der Kommunikation mit dem Server
- ▶ libnutwireless: Abstraktion der Kommunikation mit dem wpa_supplicant
- ▶ QNut: Grafische Benutzeroberfläche in Qt
- ▶ Genutzte Frameworks/Libraries: Qt4, dbus, libnl, iwlib

Überblick der Strukturen



- ▶ Devices: Entsprechen den Hardwaregeräten
 - ▶ Erkennt Zustandsänderungen wie Kabel ein-/ausstecken oder WLAN Verbindung.
 - ▶ Kann für WLAN Karten den wpa_supplicant automatisch starten und beenden.
- ▶ Environments: Entsprechen Umgebungen; z.Bsp. Arbeitsplatz, Zuhause, ...
- ▶ Interfaces: Entsprechen je einer IP

Environment

- ▶ Environments werden je nach Konfiguration vom Server automatisch ausgewählt, Kriterien für die Auswahl sind:
 - ▶ Der WLAN Name, in dem sich das Device befindet (ESSID)
 - ▶ Das Vorhandenseins eines Rechners mit einer bestimmten IP (und evtl. passender MAC Adresse)
 - ▶ Wunsch des Benutzers.

IP Zuweisung

- ▶ 3 verschiedene Methoden:
 - ▶ Statisch konfigurierte (oder vom Benutzer eingegebene) IP
 - ▶ Per DHCP (benötigt einen DHCP-Server)
 - ▶ Zeroconf (aka “IPv4 Link-Local Adresses”, RFC 3927); es wird eine lokal freie IP aus dem Bereich 169.254/16 gesucht. Die IP ist nur im lokalen Netzwerk gültig.
- ▶ Zur IP gehören weitere Werte, die auch pro Interface konfiguriert werden: Netmask, Gateway, DNS-Server.

Weitere Features

- ▶ Eventgesteuerte Scriptausführung: das ermöglicht z.Bsp. den Aufbau eines VPN nach dem Aufbau der darunterliegenden Verbindung (die Skripte haben Rootrechte)
- ▶ Unterstützt Plug'n'Play von Netzwerkgeräten, z.Bsp. PCMCIA Karten oder Laden und Entladen von Treibern wegen Inkompatibilität mit Suspend.

Konfigurationsbeispiel 1

```
device "eth0" {
    no-auto-start; //Nicht beim starten von nuts aktivieren
    dhcp; //Default environment dhcp (optional)
    environment "zeroconf" zeroconf; //Zeroconf environment
    //Environment mit benutzerdefinierbarem Interface
    environment "userdefineable" static user;
    // Environment wird abhängig von einer IP-Adresse ausgewählt
    // MAC-Adresse ist optional
    environment "home" select arp 192.168.0.1 00:07:40:EC:D0:BE;
};

device "eth1" {
    wpa-supPLICANT config "/etc/wpa_supplicant/wpa_supplicant.conf" driver "wext";
    dhcp;
    environment "infeap" {
        select essid "infeap";
    };
    environment "static/dynamic" {
        dhcp;
        static {
            ip 192.168.0.61;
            netmask 255.255.255.0;
            gateway 192.168.0.1;
            dns-server 192.168.0.1;
        };
    };
};
```

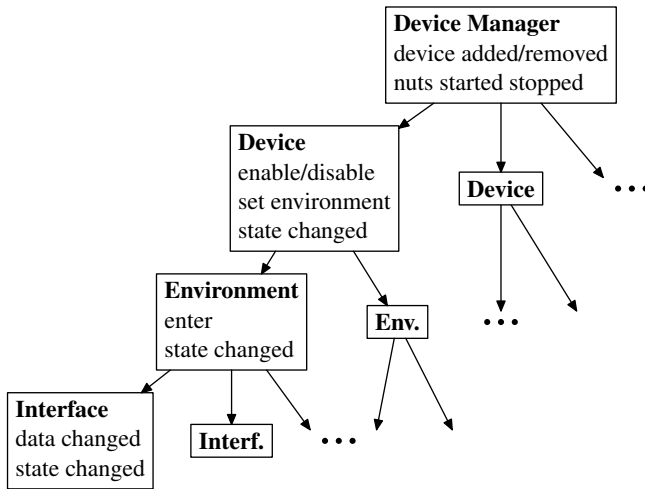
Konfigurationsbeispiel 2

```
device "eth1" {
    wpa-supPLICant config "/etc/wpa_supplicant/wpa_supplicant.conf" driver "wext";
    dhcp;
    environment "infeap" {
        select essid "infeap";
    };
    environment "home" {
        dhcp;
        select or {
            and {
                essid "buftop";
                arp 192.168.0.1 00:07:40:EC:D0:BE;
            };
            and {
                essid "unten";
                arp 192.168.178.1 00:15:0C:46:2D:C7;
            };
        };
    };
};
environment "static/dynamic" {
    dhcp;
    static {
        ip 192.168.0.61;
        netmask 255.255.255.0;
        gateway 192.168.0.1;
        dns-server 192.168.0.1;
    };
};
environment "zeroconf" {
    zeroconf;
};
};
```

nut client und wireless client

- ▶ libnutclient : Qt-Library für den Client-Teil der DBus Verbindung zum Server.
- ▶ Verwaltungsstruktur wie im Server
- ▶ libnutwireless : Qt-Library für die wpa_supplicant Verwaltung und Serverkommunikation.

nut client und wireless client



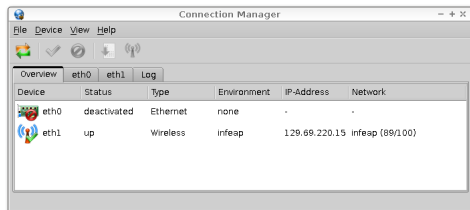
Überblick

- ▶ Abstraktion der Kommunikation mit dem wpa_supplicant und der WirelessExtension
- ▶ Überblick über den Hauptteil der Bibliothek:
 - ▶ Bereitstellung von Informationen zur Signalqualität
 - ▶ Es kann nach Netzwerken gescannt werden
 - ▶ Hinzufügen/Entfernen von Netzwerken (auch aus dem Scan)
 - ▶ Konfiguration eines bereits vorhandenen Netzwerks
 - ▶ Speichern der Konfiguration (sofern erlaubt)
- ▶ Wenn möglich, automatisches Setzen von benötigten Parametern

QNut

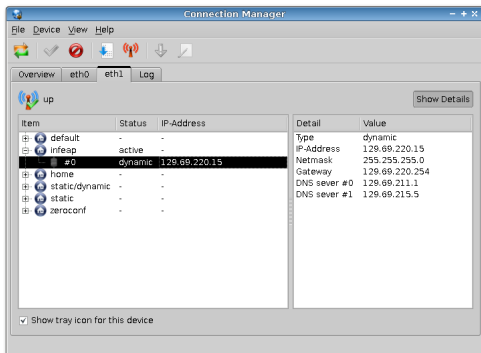
- ▶ nahezu vollständige Steuerung des Servers über die Library
- ▶ Steuerung des wpa_supplicant ebenfalls über die Library
- ▶ benutzerspezifische Skripte möglich
- ▶ Feedback für den Benutzer

Hauptfenster



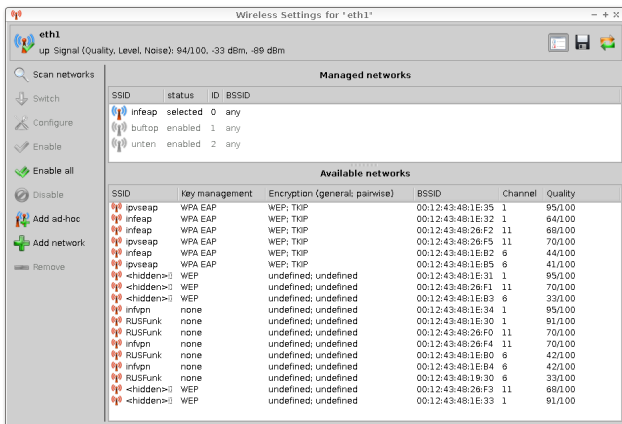
- Übersichten für Devices, Environments und Drahtlosnetzen
- Aktueller Status

Steuerung und Skripte



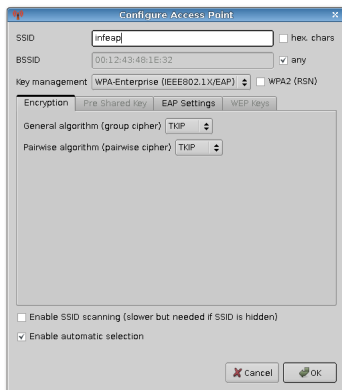
- ▶ Detailansichten für Environments und Interfaces
- ▶ Auswahl eines aktiven Environments
- ▶ Konfiguration eines Interfaces

Steuerung und Skripte



- ▶ Anpassung der verwalteten Drahtlosnetze
- ▶ Scannen nach vorhandenen Drahtlosnetzen

Steuerung und Skripte



- ▶ Hinzufügen und Entfernen von Drahtlosnetzen
- ▶ Konfiguration vorhandener Netzwerke

Sonstiges



- ▶ Tray-Icon zum schnelleren Zugang
- ▶ Statusänderung der Devices als Tool-Tip
- ▶ Ausführung von Skripten je Devicestatus

Andere Netzwerkmanager

		kwifi	gtkwifi	wifi radar	NetworkManager	NUt
Gerät	Environments	X	X	X	X	✓
	Ethernet	X	X	X	✓	✓
	Drahtlos	✓	✓	✓	✓	✓
	PPP	X	X	X	✓	X ⁴
IP-Konf.	static	✓	✓	✓	✓	✓
	dhcp	✓	✓	✓	✓	✓
	zeroconf	X	X	X	✓	✓
WLAN	unverschlüsselt	✓	✓	✓	✓	✓ ³
	WEP	✓	✓	✓	✓	✓ ³
	WPA	✓	✓	✓	✓	✓ ³
	Konfig. speichern	✓ ⁵	✓ ⁵	✓ ⁵	✓ ⁵	✓ ³

- 1 built-in
- 2 external
- 3 über wpa_supplicant
- 4 mit Skripten möglich aber nicht als Konfigurationsoption verfügbar
- 5 eigene

Motivation-NetworkManager<=>NUT

	NetworkManager	NUT
Environments	X	✓
Server-Skripte	X	✓
Nutzer-Skripte	X	✓
QT-GUI	✓(KDE3)	✓(Qt4)
GTK-GUI	✓(Gnome)	X
DBus-Interface	✓	✓
Client-Library	X	✓(Qt4)
Autom. Netzwechsel	✓	nur pro Gerät
Plugin-Struktur	✓	X

Mögliche zukünftige Änderungen

- ▶ Bugfixes (sofern weitere nötig)
- ▶ Server
 - ▶ Fallback-Konfiguration für DHCP
 - ▶ Signal vom Server falls ein neues Netzwerk gefunden wurde
 - ▶ PPP als Konfigurationsoption (momentan nur über Skripte)
- ▶ libnut/qnut
 - ▶ Etwas mehr Unabhängigkeit vom wpa_supplicant
 - ▶ Mehr WLAN-Informationen (z.B. Übertragungsgeschwindigkeit)
- ▶ Sehr zukünftiges
 - ▶ DBus Interface generischer machen
 - ▶ Konsolencient
 - ▶ IPv6

Warum brauche ich NUT?

- ▶ Es ist kostenlos (GPL)
- ▶ Es ist schnell
- ▶ Es verbraucht wenig Speicher und keine CPU-Zeit (im Idle)
- ▶ Es steuert alle gängigen vom wpa_supplicant unterstützten
- ▶ Es steuert alle Netzwerkadapter an
- ▶ Es ist durch die Skripte enorm flexibel

Ende

Homepage, Debuild Skripte und Ebuild:
<http://repo.or.cz/w/nut.git>