Interfaces-Magie (/etc/network /interfaces auf Debian)

(2009-09-22)

Einer der Punkte, die eine Linux-Distribution wirklich auszeichnet, ist die Methode der Konfiguration. Insbesondere interessant ist zurzeit die Konfiguration der Netzwerk- und WLAN-Karten, besonders wenn es sich um ein Notebook handelt.

Bei Debian gibt es hierfür die Datei `/etc/network/interfaces`, die mehr kann, als man auf den ersten Blick vermuten würde. In diesem Artikel möchte ich kurz beschreiben, was man damit alles erreichen kann und warum das sogar noch toller ist, als die Netzwerkumgebungen bei Mac OS X ;-).

Zunächst einmal sollte man folgende Pakete nachinstallieren:

- `resolvconf` für die Verwaltung der Datei `/etc/resolv.conf`, welche die Nameserver beinhaltet (ansonsten kann man die Nameserver nicht über `interfaces` konfigurieren).
- `ifplugd` damit automatisch die Netzwerkkarte konfiguriert wird, sobald man ein Kabel einsteckt
- `guessnet` damit man automatisch eine Konfiguration aufgrund verschiedener Annahmen aktivieren lassen kann (zum Beispiel das Netz zuhause).

Bei der Konfiguration von `ifplugd` (die man mit `dpkg-reconfigure ifplugd` erneut aufrufen kann) habe ich folgende Optionen angegeben: `-q -f -u0 -d2 -w -I -b`

Der Unterschied zu den Standardoptionen ist nur, dass er schneller (2 statt 10 Sekunden) nach Entfernen des Netzwerkkabels die Konfiguration deaktiviert und dass er nicht mehr piepst (`-b`).

WLAN von wpa_supplicant konfigurieren lassen

Wer schonmal WPA-Netze mit Linux benutzt hat, ist sicherlich mit wpa_supplicant in Berührung gekommen. wpa_supplicant kann aber nicht nur WPA-Netze konfigurieren, sondern auch WEP- und offene Netze.

Konfiguriert wird wpa_supplicant über die Datei `/etc/wpa_supplicant /wpa_supplicant.conf`. Bei Debian sind die Standardeinstellungen sinnvoll, sodass man sich darauf beschränken kann, die Netze an sich einzurichten. Folgendes Beispiel dient dazu, sich mit einem offenen Netz zu verbinden:

```
network={
        ssid="FirmenNetz"
        key_mgmt=NONE
}
```

Wenn man in alle Netze möchte, die offen sind:

```
network={
        key_mgmt=NONE
}
```

Für ein WEP-Netz sieht das ganze ähnlich aus:

```
network={
        ssid="FirmenNetz"
        wep_key0="passwort"
        key_mgmt=NONE
}
```

Bei WPA-Netzen hat man nun zwei Möglichkeiten: Entweder, man gibt das Passwort (ich gehe hier nur auf WPA-PSK, also mit Passwort ein) im Klartext an oder man benutzt wpa_passphrase, um einen Hash zu erzeugen, den man dann ebenso verwenden kann. Der Vorteil am Hash ist, dass man die Datei öffnen kann (um beispielsweise ein neues Netz zu konfigurieren), ohne dass direkt alle Passwörter auf dem Bildschirm stehen. Im folgenden also zwei Beispiele, einmal mit Klartext-Passwort, einmal mit Hash:

```
network={
        ssid="FirmenNetz"
        psk="GeheimesPasswort"
}

network={
        ssid="FirmenNetz"
        psk=ae2c2eda8f85d336542ad773504d6290a63153e0d4bd139f18fe32c8f7802855
}
```

wpa_supplicant einbinden

Damit wpa_supplicant automatisch gestartet wird, sobald das WLAN-Interface da ist, trägt man folgende Konfiguration in /etc/network/interfaces ein (wlan0 muss natürlich durch den Namen des WLAN-Interfaces ersetzt werden):

```
iface wlan0 inet manual
        wpa-driver wext
        wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
```

IP-Konfiguration bei WLAN festlegen

Der Vorteil ist nun, dass man bei wpa_supplicant pro Netz angeben kann, wie es heißen soll (id_str) und man für jedes Netz in der /etc/network/interfaces dann separate Einstellungen machen kann. Falls man keinen Namen angegeben hat, wird „default" benutzt. Folgende Änderungen machen wir also an der /etc/wpa_supplicant /wpa_supplicant.conf:

```
network={
        ssid="FirmenNetz"
        psk=ae2c2eda8f85d336542ad773504d6290a63153e0d4bd139f18fe32c8f7802855
        id_str="firma"
}
```

Und die /etc/network/interfaces passen wir folgenderweise an:

```
iface firma inet static
        address 192.168.1.42
        netmask 255.255.255.0
        gateway 192.168.1.1
        dns-search firma.lan
        dns-nameservers 192.168.1.1
```

Sobald sich wpa_supplicant also nun mit dem WLAN „FirmenNetz" verbindet, werden automatisch die oben genannten Einstellungen vorgenommen.

IP-Konfiguration für LANs

Um guessnet zu aktivieren fügen wir folgende Zeilen in die /etc/network/interfaces ein:

```
mapping eth0
        script guessnet-ifupdown
        map default: default
```

Anschließend kann man zum Beispiel (für weitere Optionen siehe Manpage von guessnet) folgende Konfiguration benutzen:

```
iface home inet static
        address 192.168.1.23
        netmask 255.255.255.0
        gateway 192.168.1.1
        dns-nameservers 192.168.1.1
        test peer address 192.168.1.1 mac 00:a3:23:51:2c:42

iface default inet dhcp
```

Bevor nun eine IP-Konfiguration vorgenommen wird, läuft guessnet an und prüft, ob

der Rechner mit der IP-Adresse die angegebene MAC-Adresse hat. Falls ja, werden die Einstellungen von diesem Konfigurationsblock benutzt („home" in diesem Fall).

Eigene Scripts (VPN-Client starten)

Für manche Netze (Uni) braucht man einen VPN-Client oder ähnliches. Um beim Aktivieren/Deaktivieren eines Interfaces beliebige Aktionen zu starten, gibt es die up/down-Direktive. Folgendes Beispiel sollte es klarmachen, wie das funktioniert.

```
iface uni inet dhcp
        dns-nameservers 129.206.100.126 129.206.210.127
        dns-search urz.uni-heidelberg.de
        # Wir starten vpnc in einer Schleife, weil das Ding so oft abschmiert
        up /home/michael/Uni/VPN/wrapper.sh &
        down /home/michael/Uni/VPN/killwrapper.sh
```

Das Wrapperscript sieht so aus:

```
#!/bin/sh

while [ 1 ]; do
        # As long as the vpn server is reachable...
        ping -c 1 -W 5 vpn.uni-heidelberg.de
        if [ $? -eq 0 ]; then
                # ...check if we need the VPN
                ping -c 1 -W 5 www.google.de
                [ $? -eq 0 ] && { exit; }

                # or run the VPN
                /usr/local/sbin/vpnc --no-detach /etc/vpnc/unihd.conf
        fi
        sleep 1
done
```

Und das Killscript so:

```
#!/bin/sh
killall wrapper.sh
/usr/local/sbin/vpnc-disconnect || exit 0
```

Automatische Erkennung überschreiben

Sollte die Erkennung mal versagen, oder man ist zu faul um alles korrekt aufzusetzen, kann man zum Beispiel die Benutzung der IP-Einstellungen für das Uni-Interface folgendermaßen enforcieren:

```
# ifup eth0=uni
```

Vollständiges Beispiel

Die Datei `/etc/network/interfaces`:

```
auto lo
iface lo inet loopback

allow-hotplug wlan2 eth0

# Bei LAN bestimmt guessnet den Namen des Interfaces
mapping eth0
        script guessnet-ifupdown
        map default: default

# Bei WLAN bestimmt wpa_supplicant den Namen des Interfaces
iface wlan2 inet manual
        wpa-driver wext
        wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf

# Standardinterface, einfach DHCP
iface default inet dhcp

# Kein Kabel? Dann keine Konfiguration
iface eth0 inet manual
        test missing-cable
        pre-up echo No link present.
        pre-up false

# Zuhause
iface home inet static
        address 192.168.1.42
        netmask 255.255.255.0
        network 192.168.1.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
        dns-nameservers 192.168.1.1
        test peer address 192.168.1.1 mac 00:a3:23:51:2c:42

iface uni inet dhcp
        dns-nameservers 129.206.100.126 129.206.210.127
        dns-search urz.uni-heidelberg.de
        # Wir starten vpnc in einer Schleife, weil das Ding so oft abschmiert
        up /home/michael/Uni/VPN/wrapper.sh &
        down /home/michael/Uni/VPN/killwrapper.sh
```

Die Datei `/etc/wpa_supplicant/wpa_supplicant.conf`:

```
network={
        ssid="UNI-HEIDELBERG"
        key_mgmt=NONE
        id_str="uni"
}

network={
        ssid="foo"
        psk="bar"
        id_str="home"
}
```