Einrichtung und WLAN unter CentOS 5

Bei der Einrichtung einer WLAN-Karte gibt es je nach verwendeter Hardware unterschiedliche Ansätze. Nachfolgende Doku beschreibt die Nutzung des originalen VirDOS Treiber mittels **ndiswrapper** bei der ZyXEl AG120 PCMCIA-Karte und eines native Linuxtreibers von Ralink bei einer **N1 Wireless Notebook Card** (*Part-No. **F5D8011***) von **BELKIN**.

WLAN-Karte einrichten mit ndiswrapper

Da die WLAN-Karte AG120 von ZyXEL nicht adhoc vom Kernel unterstützt wird, bedienen wir uns der Programmbibliothek **ndiswrapper**.

ndiswrapper installieren

Diese installieren wir einfach via **yum** nach:

```
# yum install dkms-ndiswrapper
```

Hersteller-Treiber installieren

Zu erst einmal installieren wir den für VirDOS vorgesehenen Treiber via:

```
# ndiswrapper -i /etc/AG-120/AG120.inf
```

Hiermit wird der Treiber unter **/etc** folgende Verzeichnisstruktur angelegt:

```
# pwd
/etc/ndiswrapper/ag120
# ll
insgesamt 372
-rw-r--r-- 1 root root    726 22. Sep 16:31 167B:2116:0308:3409.5.conf
lrwxrwxrwx 1 root root     26 22. Sep 16:31 167B:2116.5.conf ->
167B:2116:0308:3409.5.conf
-rw-r--r-- 1 root root  17124 22. Sep 16:31 ag120.inf
-rw-r--r-- 1 root root 332800 22. Sep 16:31 ag120.sys
```

Modul laden

Anschließen lädt man das *ndiswrapper*-Modul mit dem Befehl:

```
# modprobe ndiswrapper
```

Mit diesem Befehl wird der Treiber für die Karte geladen, mit dem Erfolg, dass damit nun die Karte

selbst wie gewünscht funktioniert.

Konfiguration

Abschließend tragen wir in der **/etc/modprobe.conf** noch folgende Zeile ein:

```
alias wlan0 ndiswrapper
```

Alternativ können wir auch mit der Option **-m** von *ndiswrapper* erledigen lassen.

```
# ndiswrapper -m
```

Damit legen wir fest, mit welchem Device, hier **wlan0**, später die Verbindung aufgebaut werden soll.

RT2860 Linux-Treiber WLAN-Karte einrichten

Da nach dem Laden des originalen Treibers durch **ndiswrapper** bei der BELKIN-Karte der Rechner einfriert, nutzen wir bei dieser Karte einen anderen Weg.

Kernel-Entwicklungspaket installieren

Damit wir unser WLAN-Kernelmodul bauen können,benötigen wir noch das Kernel-Entwicklungspaket ***kernel-devel***, welches wir - wenn noch nicht bereits passiert - einfach nachinstallieren.

```
# yum install kernel-devel
```

Download

Zuerst laden wir uns von der Seite das passende Archiv (2010_01_29_RT2860_Linux_STA_v2.3.0.0.tar.bz2): **RT2860PCI/mPCI/CB/PCIe(RT2760/RT2790/RT2860/RT2890)**.

```
 # wget
http://www.ralinktech.com/download.php?t=U0wyRnpjMlYwY3k4eU1ERXdMekF4THpJNUw
yUnZkMjVzYjJGa05ERTJNVEV5T1RFd05pNWllakk5UFQweU1ERXdYekF4WHpJNVgxSlVNamcyTUY
5TWFXNTFlRjlUVkVGZmRqSXVNeTR3TGpBdWRHRnlD -O
2010_01_29_RT2860_Linux_STA_v2.3.0.0.tar.bz2
```

Installation

Das heruntergeladene Archiv kopieren wir dann in unser Installationsverzeichnis

```
# mv 2010_01_29_RT2860_Linux_STA_v2.3.0.0.tar.bz2 /usr/local/src/
```

und wechseln in das Zielverzeichnis.

```
# cd /usr/local/src/
```

Hier entpacken wir das Archiv:

```
# tar -xjvf 2010_01_29_RT2860_Linux_STA_v2.3.0.0.tar.bz2
```

Und setzen abschließend einen symbolischen link auf unser entpacktes Verzeichnis:

```
# ln -s 2010_01_29_RT2860_Linux_STA_v2.3.0.0 RT2860
```

Vor dem Übersetzen des Modules passen wir noch eine Datei an, damit wir später den Treiber durch NetworkManager nutzen können.

```
# vim /usr/local/src/RT2860/os/linux/config.mk

...
# Support Wpa_Supplicant
HAS_WPA_SUPPLICANT=y

# Support Native WpaSupplicant for Network Maganger
HAS_NATIVE_WPA_SUPPLICANT_SUPPORT=y
```

Anschließend übersetzen wir den Treiber Source Code.

```
# make
```

Nach der erfolgreichen Kompilierung lassen wir die kompilierten Treiber und alles was dazu gehört installieren.

```
# make install
```

Kernel-Modul laden

Nachfolgender Befehl lädt das **Kernel-Modul** abschließend und **aktiviert** damit die Karte, welche dann durch den z.B. **NetworkManager** verwaltet werden kann:

```
# modprobe rt2860sta
```

Das erfolgreiche laden des **Kernel-Modules** kann mit nachfolgendem Befehl überprüft werden, welche eine Ausgabe in etwa wie nachfolgend dargestellt erzeugen sollte:

```
# lsmod | grep rt2860sta
rt2860sta             617072  1
```

Die Karte sollte nun Einsatzbereit sein !

Network Manager

Der einfachste und komfortabelste Weg bei der Nutzung der WLAN-Verbindungen ist wohl die Verwendung des **NetworkManagers**.

Installation

Diesen intsallieren wir am einfachsten via **yum**.

```
# yum install NetworkManager
```

Erster Programmstart

Den Dienst starten wir einfach durch:

```
# service NetworkManager
```

automatisches Starten

Damit nun unser NetworkManager-Daemon beim Booten automatisch gestartet wird, nehmen wir noch folgende Konfigurationsschritte vor.

```
# chkconfig NetworkManager on
```

Anschließend überprüfen wir noch unsere Änderung:

```
# chkconfig --list | grep NetworkManager
NetworkManager  0:Aus   1:Aus   2:Ein   3:Ein   4:Ein   5:Ein   6:Aus
```

PasswortManager

Der NetworkManager speichert den *WPA-Schlüssel* im **gnome-keyring manager**. Damit der NetworkManager auf diesen Keyring zugreifen kann, wird bei der Anmeldung das Passwort des Schlüsselbundes abgefragt. Zuweilen kann diese abfrage als ser störend empfunden werden.
An statt das Passwort des Gnome-schlüsselbundes ganz anzuschalten (Sichereitsrisiko) bedienen wir uns des **pam_keyring**. Grundidee dabei ist, dass wir das gleiche Passwort beim Schlüsselbund wie beim Anmelden verwenden.

Installation

Als erstes installieren wir uns das vorgenannte Paket **pam_keyring** via **yum**.

```
# yum install pam_keyring gnome-keyring-manager
```

Konfiguration

Anschließend passen wir die Konfigurationsdatei ***/etc/pam.d/gdm***

```
# vim /etc/pam.d/gdm
```

```
#%PAM-1.0
auth       required    pam_env.so
# Django 15.03.2010 eingefügt zum einfachen Anmelden
auth       optional    pam_keyring.so try_first_pass
#
auth       include     system-auth
account    required    pam_nologin.so
account    include     system-auth
password   include     system-auth
session    optional    pam_keyinit.so force revoke
session    include     system-auth
session    required    pam_loginuid.so
session    optional    pam_console.so
# Django 15.03.2010 eingefügt zum einfachen Anmelden
session    optional    pam_keyring.so
```

Passwort synchronisieren

Sofern wir (*noch*) unterschiedliche Passworte bei der Anmeldung und beim Gnome-schlüsselbund verwenden, synchronisieren wir die beiden Passworte.

```
[django@daxy ~]$ /usr/libexec/pam-keyring-tool -c
Old password:
New password:
Verify password:
```

Verschlüsselungskonfiguration des Access-Points ermitteln

Um sicher zu stellen, wie die Konfigurationsdatei für den WPA-Bittsteller **wpa_supplicant** gefüttert werden muss, ermitteln wir erst einmal die Möglichkeiten des Access-Points. Hierzu benutzen wir das Programm **iwlist**. Der Befehl

```
iwlist scanning
```

zeigt alle WLANs in Reichweite inklusive ihrer Verschlüsselungskonfiguration an.

```
[root@compaq-evo ag120]# iwlist wlan0 scanning
wlan0     Scan completed :
```

```
          Cell 01 - Address: 00:19:CB:32:FA:5C
                    ESSID:"Hagbard_Celine"
                    Protocol:IEEE 802.11g
                    Mode:Managed
                    Frequency:2.417 GHz (Channel 2)
                    Quality:37/100  Signal level:-72 dBm  Noise level:-96
dBm
                    Encryption key:on
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                              9 Mb/s; 12 Mb/s; 48 Mb/s; 18 Mb/s; 24 Mb/s
                              36 Mb/s; 54 Mb/s
                    Extra:bcn_int=100
                    Extra:atim=0
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
```

WPA_SUPPLICANT Einrichten

Damit wir eine Verbindung zum WLAN-Sender aufbauen können, benötigen wir noch ***wpa_supplicant***.

Installation vom WPA-Bittsteller

Zuerst insallieren wir uns dir benötigten Programmteile via yum.

```
yum install wpa_supplicant
yum install wpa_supplicant-gui
```

Konfiguration von WPA-Supplicant

Da wir zum Ansprechen der WLAN-Karte den ndiswrapper benutzen und dort neben dem Kartenalias ***wlan0*** den Treiber ***wext*** verwenden wollen, tragen wir die nötigen Zeilen in die Konfigurationsdatei ein.

```
vi /etc/sysconfig/wpa_supplicant
INTERFACES="-iwlan0"
DRIVERS="-Dwext"
```

Schlüssel für das WLAN erstellen

Damit wir den Schlüssel in der Konfigdatei richtig eintragen können, erstellen wir uns erst selbigen mit dem Prgramm ***wpa_passphrase***:

```
wpa_passphrase Hagbard_Celine Basti-is-a-geek >>
/etc/wpa_supplicant/wpa_supplicant.conf
```

Konfigurationsdatei für WPA-Supplicant vervollständigen

Passend zu den Möglichkeiten des Access-Points tragen wir nun in die Konfigurationsparameter in der ***/etc/wpa_supplicant/wpa_supplicant.conf*** nach:

```
ctrl_interface=/var/run/wpa_supplicant
eapol_version=1
ap_scan=1
network={
        ssid="Hagbard_Celine"
        scan_ssid=1
        proto=RSN
        key_mgmt=WPA-PSK
        pairwise=CCMP
        group=CCMP
        #psk="Basti-is-a-geek"
        psk=46fb2a059b712ce5ed497c555759b931234373c7f082ac064980e283489274f0
}
```

Service waproamd stoppen und entfernen

Da sich **waproamd** mit **wpa_supplicant** nicht vertragen und **waproamd** laut deren Web-Seite nicht mehr weitergepflegt wird, stoppen wir den Dienst mit:

```
service waproamd stop
```

und entfernen das Programmpaket vom System via

```
yum erase waproamd
```

Service wpa_supplicant starten

Bevor wir den wpa_supplicant Service starten, testen wir, ob die Verbindung zum Access-Point hergestellt werden kann.

```
wpa_supplicant -iwlan0 -Dwext -c/etc/wpa_supplicant/wpa_supplicant.conf -d
```

Nach entsprechender positiver Rückmeldung starten wir den Service neu:

```
/sbin/chkconfig --level 2345 wpa_supplicant on
```