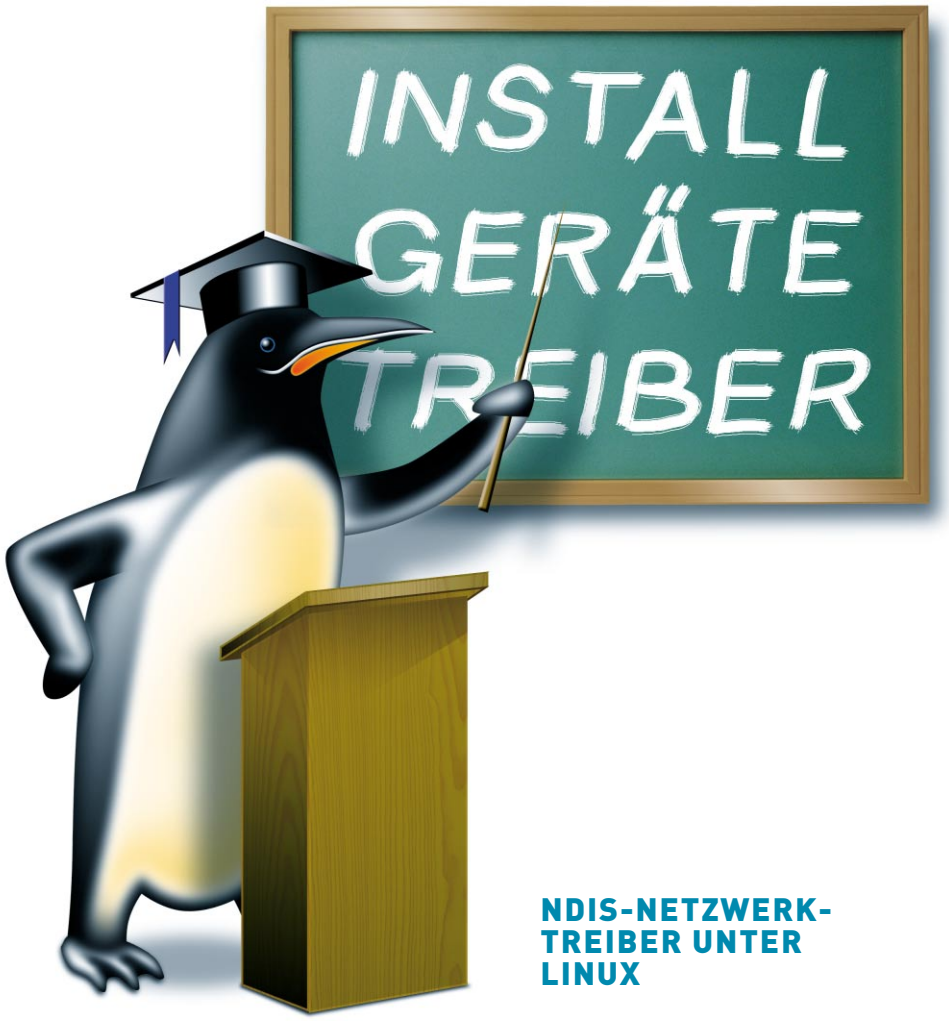
PRAXIS



NDIS-NETZWERK-TREIBER UNTER LINUX

Kooperationsbereit

Einfach Windows-Treiber unter Linux benutzen: Bei WLAN-Karten funktioniert das schon gut. Wir erklären die Funktionsweise des Ndiswrappers und zeigen, wie Sie die möglichen Klippen der Konfiguration umschiffen.

VON **MATTIAS SCHLENKER**

Ähnlich wie der NTFS-Dateisystem-Treiber *Captive* funktioniert *Ndiswrapper* bei WLAN-Karten: Dem originalen Windows-Treiber wird vorgegaukelt, in einem NT-Kernel zu arbeiten, während sich der NDIS-Treiber zum Linux-Kernel hin wie ein ganz normaler Linux-Treiber verhält. Im Gegensatz zum ebenfalls in dieser Ausgabe vorgestellten *Captive* läuft *Ndiswrapper* als gewöhnliches Kernel-Modul und benötigt keinen Userspace-Daemon. In der Theorie sollte dies dem *Ndiswrapper* eine ähnliche Performance wie ein normaler nativer Treiber bescheren, aber das System auch anfällig für Interrupt-Stürme und Abstürze durch Verwendung noch nicht implementierter Funktionen machen. Wie dies in der Praxis aussieht, wollen wir später herausfinden. Zunächst stehen Installation und Konfiguration an.

Gute Vorarbeit der Distributoren

Praktisch allen aktuellen Distributionen liegen verhältnismäßig neue Versionen des *Ndiswrappers* bei. Eine Stichprobe gegen Redaktionsschluss ergab, dass bis auf Ubuntu Linux 2005-10 und Debian 3.1 alle Distributionen mit Ndiswrapper in Version 1.2 oder höher ausgeliefert wurden. Spätestens mit der bei vielen Distributionen kurz nach Veröffentlichung dieses Heftes abgeschlossenen „Erneuerungsrunde“ dürfte die Unterstützung für aktuelle WLAN-Karten über die nächsten Monate garantiert sein. Für Nutzer älterer Distributionen haben wir im Kasten *An der Quelle* die Installation von *ndiswrapper* aus Quellcodes am Beispiel einer SuSE 9.2 durchgespielt.

Windows-Netzwerk-Treiber bestehen im Wesentlichen aus einer *inf*-Datei und einer *sys*-Datei. Die *inf*-Datei dient dabei als Konfigurationsdatei, während die *sys*-Datei den eigentlichen Treiber enthält. Beide Dateien liegen heutzutage meist ohne weitere Umverpackung auf der mit der Netzwerkkarte ausgelieferten CD oder im heruntergeladenen *zip*-File. In der Regel benötigen Sie die Treiber von Windows 2000 oder XP. Sehr selten ist die Treiber-Installation als *exe*-Programm

gestaltet. In diesem Fall hilft es nur, die Installation unter Windows zu starten, den ersten Dialog geöffnet zu lassen, bis das temporäre Verzeichnis mit den Treiberdateien identifiziert ist und anschließend die Installationsroutine wieder abzubrechen.

Smarte Konfiguration

Sind die beiden Treiber-Dateien identifiziert, sorgt das Kommandozeilenwerkzeug *ndiswrapper* für die Registrierung am System. Wechseln Sie in das Verzeichnis mit SYS- und INF-Datei und geben Sie den folgenden Befehl ein:

```
ndiswrapper -i dateiname.inf
```

. *ndiswrapper* liest nun die INF-Datei ein, erzeugt daraus seine Konfiguration und legt diese unter */etc/ndiswrapper/*treibername ab. Installierte Treiber zeigt der Befehl

```
ndiswrapper -l
```

an. Die Ausgabe *Driver present, Hardware present* sagt noch nicht viel über die Funktionsfähigkeit des verwendeten Windows-Treibers unter Linux aus – sie meldet lediglich anhand einer Chipsatz-ID erkannte und am PCI-, Cardbus- oder USB-Subsystem angemeldete Geräte.

Wirklich spannend ist das Laden des eigentlichen Treibers, das wie üblich mit dem Modul-Verwaltungswerkzeug *modprobe* erfolgt:

```
modprobe ndiswrapper
```

Friert jetzt der Rechner ein, sind *ndiswrapper* und Windows-Treiber nicht kompatibel. Probieren Sie in diesem Fall zunächst einen anderen Treiber aus. Gibt *modprobe* keine Fehlermeldung aus, zeigt der Befehl

```
dmesg
```

schließlich, ob der Treiber tatsächlich korrekt geladen wurde oder ob möglicherweise ein Interrupt-Sturm oder ähnliches die Nutzung verhindert. Zudem zeigen die mit *wlan0:* anfangenden Zeilen, welche Verschlüsselungsmethoden der Treiber beherrscht.

Waren Sie bis hier erfolgreich, sollten Sie mit dem Befehl

```
ndiswrapper -m
```

den Aliasnamen *wlan0* für das Modul *ndiswrapper* nachtragen. Das erspart das manuelle Nachladen nach einem Reboot.

Spannend: Suche nach dem Netz

Für Konfigurationsaufgaben an drahtlosen Netzwerkkarten, die mit dem Werkzeug *ifconfig* nicht bewältigt werden können, bringt Linux die *Wireless Tools* mit. Die zwei wichtigsten sind *iwlist* zur Suche nach Netzwerken und *iwconfig* für die übrigen Einstellungen der Karte wie die Kanalwahl oder das Setzen der WEP-Passphrase. Ohne weitere Parameter aufgerufen zeigt *iwconfig* den Status aller verfügbaren drahtlosen Netzwerkkarten an. Anschließend suchen Sie mit dem Befehl

```
iwlist wlan0 scan
```

nach den verfügbaren Netzen. Beachten Sie, dass nur Netze gefunden werden können, welche ihre ESSID per Broadcast übermitteln. Sie sollten also zumindest für die Tests den ESSID-Broadcast aktivieren. Ein netter Nebeneffekt ist, dass Sie die Signalstärke angezeigt bekommen, was bei der Ausrichtung der Antenne hilft. Im Prinzip könnten Sie nach der erfolgreichen Netzsuche die Arbeit auf der Kommandozeile abbrechen und die weitere Einrichtung der Netzwerkkarte im Konfigurationswerkzeug Ihrer Distribution vornehmen.

Wir empfehlen dennoch, alle Schritte zur Anmeldung am Netz einmal auf der Kommandozeile durchzuspielen. Dies hilft bei möglicherweise auftretenden Problemen, den Fehler einzugrenzen.

Das folgende Beispiel bezieht sich auf WEP mit einem 128-Bit-Schlüssel. Auf WPA wollen wir an dieser Stelle nicht weiter eingehen: Funktionieren die Tests mit dem vorübergehend auf WEP eingestellten Netzwerk, genügt es in der Regel, das Administrationswerkzeug der Distribution zu verwenden, um den WPA-Schlüssel zu setzen.

Stellen Sie zunächst den Modus des Netzwerkes auf *Managed* um:

```
iwconfig wlan0 mode „Managed“
```

Es folgt die Eingabe des Schlüssels. Nicht alle Versionen der *Wireless Tools* akzeptieren Passphrasen, weshalb eine Umrechnung in den bei 26-stelligen Hexadezimalcode notwendig ist.

An der Quelle

Bei Suse 9.2 funktionierte die mitgelieferte Version des ndiswrapper nicht mit unserem neueren USB-Netzwerk-Stick *Netgear WG111v2*. Wir entschlossen uns daher zur Kompilierung aus der Quellen. Die aktuellste Version des *ndiswrappers* luden wir von http://ndiswrapper.sf.net herunter.

- **Installieren Sie die Kernel-Quellcodes oder -Header (*kernel-source* oder *kernel-header*) sowie die Tools *make* und *gcc*.**
- **Entpacken Sie das heruntergeladene Quellcode-Paket und wechseln Sie in das entstandene Verzeichnis:**

```
tar xvzf ndiswrapper-1.10.tar.gz
cd ndiswrapper-1.10
```

- **Zum Kompilieren des *ndiswrappers* genügt ein simples *make*, die Installation erfolgt mit *make install*.**
- **Yast2 erkannte die Karte nicht automatisch. Wir mussten diese deshalb als USB-Netzwerkkarte unter expliziter Angabe des Moduls ndiswrapper eintragen. Die Einstellungen für den Zugang zum Netz nahmen wir ebenfalls in Yast2 vor. Mit dem von Suse mitgelieferten Kernel 2.6.8 schlug die Kompilierung fehl. Wir verwendeten deshalb den für den Captive-Artikel bereits alternativ installierten Kernel 2.6.15.3. Generell sollten mit Kerneln ab 2.6.11 keine Probleme mehr bei der Kompilierung auftreten.**

Versuchskaninchen: Der Netgear WG111v2 (ohne „T“) wird für ca. 30 Euro angeboten.

```
mattias@:~
athlontest2:/tmp/ndiswrapper-1.10 # make
make -C driver
make[1]: Entering directory `/tmp/ndiswrapper-1.10/driver'
make -C /lib/modules/2.6.15.3/build SUBDIRS=/tmp/ndiswrapper-1.10/driver \
        DRIVER_VERSION=1.10
make[2]: Entering directory `/usr/src/linux-2.6.15.3'
  LD      /tmp/ndiswrapper-1.10/driver/built-in.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/hal.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/iw_ndis.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/loader.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/misc_funcs.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/ndis.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/ntoskernel.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/ntoskernel_io.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/pe_linker.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/pnp.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/proc.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/wrapndis.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/wrapper.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/usb.o
  CC [M]  /tmp/ndiswrapper-1.10/driver/divdi3.o
  LD [M]  /tmp/ndiswrapper-1.10/driver/ndiswrapper.o
  Building modules, stage 2.
  MODPOST
  CC      /tmp/ndiswrapper-1.10/driver/ndiswrapper.mod.o
  LD [M]  /tmp/ndiswrapper-1.10/driver/ndiswrapper.ko
make[2]: Leaving directory `/usr/src/linux-2.6.15.3'
make[1]: Leaving directory `/tmp/ndiswrapper-1.10/driver'
make -C utils
make[1]: Entering directory `/tmp/ndiswrapper-1.10/utils'
gcc -g -Wall -DUTILS_VERSION=\"1.7\"  -o loadndisdriver loadndisdriver.c
make[1]: Leaving directory `/tmp/ndiswrapper-1.10/utils'
athlontest2:/tmp/ndiswrapper-1.10 #
```

Schnell durchkompiliert: Der *make*-Lauf dauert bei *ndiswrapper* kaum eine Minute.

```
iwconfig wlan0 key 1234-5678-9012-
➥ 3456-7890-1234-56
```

Mit gesetztem Schlüssel dürfen Sie „das Netz betreten“, sich also am Accesspoint anmelden:

```
iwconfig wlan0 essid netwerkname
```

Beim anschließend ausgeführten *iwconfig wlan0* wird die Hardware-Adresse des assoziierten Accesspoints angezeigt. Der Bezug der IP-Adresse läuft wie im drahtgebundenen Netz. Dynamisch mit

```
dhclient wlan0
```

oder

```
dhcpcd wlan0
```

Alternativ können Sie IP-Adresse und Netzmaske von Hand setzen:

```
ifconfig wlan0 inet 192.168.1.123
➥ netmask 255.255.255.0
```

War die manuelle Konfiguration erfolgreich, kann das dauerhafte Setzen der Schlüssel für das Netzwerk und gegebenenfalls der IP-Adresse im Konfigurationswerkzeug der Distribution erfolgen. Bei ausschließlicher Verwendung von WEP-Schlüsseln spielt es nicht einmal eine Rolle, ob *Ndiswrapper* selbst aus Quellcodes nachinstalliert wurde, da mit *Ndiswrapper* geladene Treiber den Wireless Tools die gleichen Schnittstellen zur Verfügung stellen wie native Treiber. Anders verhält es sich mit dem für WPA-Verschlüsselung verwendeten WPA-Supplicant: Hier ist in vielen Fällen die Nachinstallation einer aktuellen Version nötig.

Und die Performance?

Als einfachen Benchmark für die Übertragungsleistung gaben wir eine auf einem NFS-Share liegende Datei mit dem Universalwerkzeug *cat* nach */dev/null* aus. So konnten wir sicherstellen, dass keine Prozessorleistung fressende RSA-Verschlüsselung das Ergebnis verfälschte. Befanden sich Accesspoint und WLAN-Stick im gleichen Raum, zeigte *iwconfig* 54MBit/s Übertragungsrate. Netto lag diese jedoch im Bereich von etwa 11MBit/s, was in der gleichen Konstellation auch beim Einsatz unter Windows nur wenig übertroffen wurde.

Weit wichtiger ist jedoch die vom *ndiswrapper* erzeugte Systemlast, welche auf dem 2000er Athlon selten über 3% lag. Folglich sind auch bei einem Rechner unter 1000MHz und einer voll ausgenutzten 54MBit/s Verbindung keine Auslastungszustände zu erwarten, die das Arbeiten unerträglich machen - ganz abgesehen davon, dass beim normalen Surfen kein kontinuierlicher Datenstrom ansteht.

Fazit:

In der Summe seiner Eigenschaften ist *ndiswrapper* eine nützliche und gut durchdachte Software, die viele günstige WLAN-Karten endlich unter Linux verfügbar macht. Gut gefiel uns, dass die Distributoren das Potential erkannt haben und *ndiswrapper* ihren Paketen beilegen. Die von einigen Entwicklern geäußerten Befürchtungen, dass eine Software wie *ndiswrapper* Hardware-Hersteller davon abhalten würde, Linux-Treiber zu entwickeln, hat sich nicht bewahrheitet. Das Gegenteil ist der Fall: Mehr Anwender können auf Linux umsteigen und fragen beim Hersteller native Treiber nach. In der Folge haben gerade Firmen wie Intel ihre Unterstützung bei der Entwicklung „echter“ Linux-Treiber verstärkt. : jkn

Woher Treiber nehmen?

Probieren Sie zunächst den Treiber für Windows XP. Funktioniert dieser nicht zufriedenstellend, deinstallieren Sie ihn mit

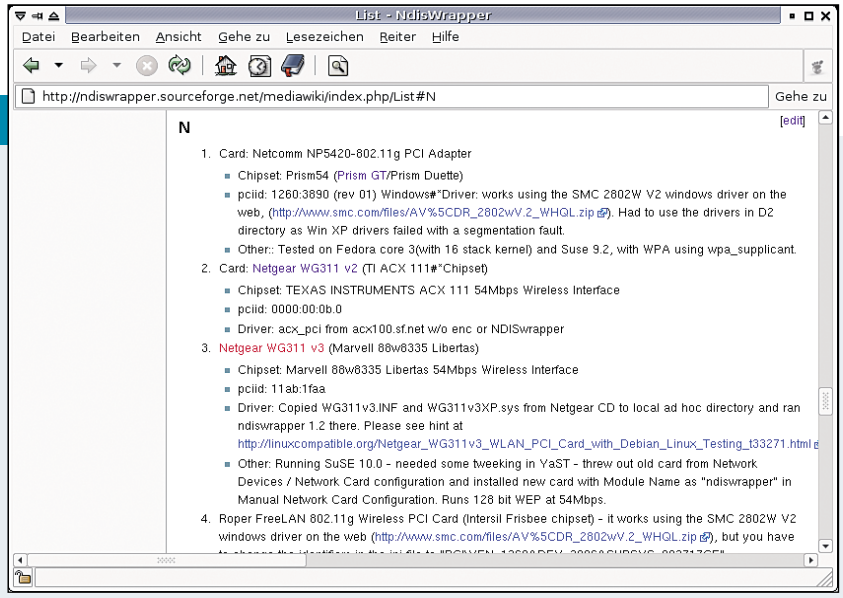
```
ndiswrapper -e treibername
```

Der *treibername* muss hierbei dem Kurznamen der Ausgabe von

```
ndiswrapper -l
```

entsprechen. Probieren Sie alternativ die Windows 2000 Version. Nur in seltenen Fällen wird es erforderlich sein, auch den Treiber von Windows 98 auszuprobieren.

Nicht immer ist der Treiber vom Hersteller Ihrer Karte am besten geeignet. Gerade No-Name-Anbieter packen gerne alte Treiber bei und bieten keine Downloads an. Die Suche in Foren und auf dem Wiki von *ndiswrapper* fördert oft baugleiche Karten oder generische Treiber für den verwendeten Chipsatz zutage. Bei der Ermittlung von Details helfen die Befehle *lsusb -v* oder *lspci -v*. Auch die „FCC-ID“ kann helfen: Karten mit gleicher FCC-ID verwenden garantiert den gleichen Chipsatz. So konnte schon manch eine zickige Karte zur Zusammenarbeit bewegt werden.



Informativ: Das Wiki auf ndiswrapper.sf.net hält Listen kompatibler Karten bereit. Tragen Sie Ihre Karte mit PCI- oder USB-ID, Treiber- und ndiswrapper-Version nach, wenn Sie eine noch nicht gelistete Karte erfolgreich eingerichtet haben.

Dass *ndiswrapper* den Linux-Distributionen beiliegt, bedeutet nicht, dass diese *ndiswrapper*-Version mit Ihrer Karte kooperiert.

Sollten die ersten beiden Tipps keinen Erfolg beschert haben, kompilieren Sie *ndiswrapper* deshalb frisch aus Quellen — am besten nach Deinstallation des Pakets vom Distributor. Gerade bei Chipsatz-Revisionen, die neuer als die Distributionsversion des *ndiswrappers* waren, hatten wir mit dieser Taktik Erfolg.