

Contents

- [1 Drahtlose Schnipsel](#)
 - ◆ [1.1 wl](#)
 - ◇ [1.1.1 wl scan](#)
 - ◆ [1.2 Welche IP-Adressen haben die Wireless-Clients?](#)
 - ◇ [1.2.1 Erweitertes Skript mit Hostnamen](#)
- [2 Traffic-Schnipsel](#)
 - ◆ [2.1 ifconfig Traffic-Monitor](#)
 - ◆ [2.2 Anzeige der maximalen Verbindungen](#)
- [3 Idle Time umgehen](#)
- [4 Kleines Sendmail-Programm \(ohne Fehlerabhandlung\)](#)
- [5 Syslog per E-Mail](#)

Drahtlose Schnipsel

wl

Die meisten Wireless-Optionen können über das Programm "wl" auch über die Konsole eingestellt und überwacht werden

Eine Übersicht über alle wl-Komandos erhält man mit der Eingabe von:

```
~ # wl cmds
ver          cmds          up            down
out          clk           restart      reboot
ucflags     radio        dump         srddump
nvddump     nvset        nvget        revinfo
msglevel    PM           wake         promisc
monitor     frag         rts          cwmin
cwmax       srl          lrl          rate
mrate       infra        ap           bssid
channel     tssi        txpwr        txpwrl
txpathpwr  txpwrlimit  powerindex  atten
phyreg      radioreg     shmem        macreg
antdiv      txant        plcphdr      phytype
scbdump     ratedump    rateparam    wepstatus
primary_key addwep       rmwep        wep
tkip        aes          keys         tsc
wsec_test   tkip_countermeasures wsec_restrict eap
authorize   deauthorize  deauthenticate wsec
wpa_auth    set_pmk      scan         passive
regulatory  spect        scanresults  assoc
status      disassoc     chanlist     channels
channels_in_country curpower     scansuppress  evm
rateset     roam_trigger roam_delta    roam_scan_period
```

Skript-Schnipsel

```
suprates          scan_channel_time scan_unassoc_time scan_home_time
scan_passive_time scan_nprobes      prb_resp_timeout  channel_qa
channel_qa_start  country          locale            join
ssid              mac             macmode           wds
lazywds           noise            fqacurcy          crsuprs
int               lbt             band              bands
phylist           shortslot        shortslot_override shortslot_restrict
ignore_bcns       pktcnt           upgrade            gmode
gmode_protection  gmode_protection_control gmode_protection_cts gmode_protection_override
legacy_erp        scb_timeout      assoclist          rssi
isup              fasttimer        slowtimer          glacialtimer
radar             rssidump         interference       aciargs
frameburst        pwr_percent      wet                dtim
wds_remote_mac    wds_wpa_role_old wds_wpa_role       authe_sta_list
autho_sta_list    measure_req      quiet              csa
constraint         rm_req           rm_rep             wme
sta_info          cap
~ #
```

Dieses Programm beherrscht weit mehr Optionen als WRT überhaupt verarbeiten kann. Einige Befehle können nur im Client-Modus und andere nur im AP-Modus verwendet werden.

Hier folgen ein paar kleine Client-Beispiele:

wl scan

Was machts? Das drahtlose Netzwerk wird im über alle Kanäle gescannt, nach eine Pause von 5 Sekunden wird der Bildschirm gelöscht, und das Ergebnis des Scanns ausgegeben. Mit der Tastenkombination Strg-C kann das Skript beendet werden.

```
#!/bin/sh
while [ 1 ]; do
  wl scan
  sleep 5
  clear
  wl scanresults
done
```

Ausgabe:

```
Mode: Managed  RSSI: -76 dBm  noise: -93 dBm  Channel: 10
BSSID: 00:FB:DD:11:22:33      Capability: ESS
Supported Rates: [ 1(b) 2(b) 5.5 11 ]
```

```
SSID: "\x00G\x11ARBOP\x00\x01\xCD\x0A\x13\x90\x00\x02-PE\xFE"
Mode: Managed  RSSI: -85 dBm  noise: -91 dBm  Channel: 4
BSSID: 11:22:33:50:45:FE      Capability: ESS
Supported Rates: [ 1(b) 2(b) 5.5(b) 11(b) ]
```

Um auch die versteckten Netzwerke angezeigt zu bekommen, kann man den Scanner in den Monitor- und passiven Modus versetzen. Die zweite SSID ist übrigens eine Turbo-Cell

```
~ # wl monitor 1
~ # wl passive 1
```

Skript-Schnipsel

Jetzt kann es natürlich unübersichtlich werden ... Das Programm "grep" hilft bei der Auswahl von Zeilen, die den gewünschten Informationsgehalt besitzen

Hier wird nach *SSID*: und *Mode*: gesucht

```
while [ 1 ]; do
  wl scan
  sleep 5
  clear
  wl scanresults | grep 'SSID: "\|Mode:'
done
```

Das Ergebnis sieht dann so aus:

```
SSID: "www.fbn-dd.de (HSS)"
Mode: Managed   RSSI: -76 dBm   noise: -88 dBm   Channel: 10
SSID: "WLAN"
Mode: Managed   RSSI: -94 dBm   noise: -88 dBm   Channel: 11
SSID: "WLAN"
Mode: Managed   RSSI: -96 dBm   noise: -88 dBm   Channel: 11
SSID: "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
Mode: Managed   RSSI: -75 dBm   noise: -88 dBm   Channel: 1
SSID: "\x00G\x11ARBOP\x00\x01\xCD\x0A\x13\x90\x00\x02-PE\xFE"
Mode: Managed   RSSI: -87 dBm   noise: -88 dBm   Channel: 4
```

Jetzt hat man also einen Überblick, was denn alles so auf den Kanälen gefunkt wird.

In meinem Falle bleibt mir nichts anderes übrig, als mich mit *www.fbn-dd.de (HSS)* zu verbinden.

Mich interessiert also nur dieser.

```
while [ 1 ]; do
  CHANNEL=$(wl channel | grep target | cut -c16- )
  MAC=$(wl assoclist|cut -d' ' -f2 )
  RATE=$(wl rate|cut -d' ' -f3 )
  RSSI=$(wl rssi|cut -d' ' -f3 )
  NOISE=$(wl noise|cut -d' ' -f3 )
  let SNR="$RSSI- $NOISE"
  clear
  echo "MAC=$MAC Kanal $CHANNEL: Rate=$RATE RSSI=$RSSI Noise=$NOISE SNR=$SNR"; sleep 5
done
#Ergibt:
MAC=00:FB:DD:11:22:33 Kanal: Rate=11 RSSI=-77 Noise=-91 SNR=14
```

Die Idee zum letzten Schnipsel stammt von [hier](#).

Welche IP-Adressen haben die Wireless-Clients?

Was machts? Der Befehl **wl assoclist** gibt die MAC-Adressen der verbundenen Funk-Clients zurück. **cat /proc/net/arp** liefert alle Clients mit MAC- und IP-Adresse, die in letzter Zeit aktiv waren. Sucht man in der ARP-Tabelle nach den MAC-Adressen der assoclist, dann kann man den drahtlosen (?Sinn?) IP-Adressen zuordnen.

```
wl assoclist | awk '{print$2}' > /tmp/assocLIST
while read assocLINE
do
  awk '/'"$assocLINE"'/ {print$1}' /proc/net/arp
done < /tmp/assocLIST
```

Skript-Schnipsel

Erweitertes Skript mit Hostnamen

Um eine bessere Übersicht zu bekommen (mit Hostnamen, MAC und IP), empfiehlt sich dieses Skript.

Dieses funktioniert allerdings nur, wenn der Wireless-Client eine IP-Adresse vom DHCP-Server bekommen hat (daher dumpleases).

```
# mkdir -p /tmp/www
while [ 1 ]; do
  wl assoclist | awk '{print$2}' > /tmp/assocLIST
  # echo "<meta http-equiv='refresh' content='10'><b>Host names and IP addresses of WLAN clients</b>"
  while read assocLINE; do
    dumpleases | awk 'BEGIN IGNORECASE=1; /"$assocLINE"/ {print "Host name: " $1, "MAC: " $2, "IP: " $3}'
    # echo "<br>";
  done < /tmp/assocLIST      # >> /tmp/www/wlan.html
  sleep 10
done
```

Output:

```
Hostname: tp MAC: 01:81:18:3d:49:5e IP: 192.168.2.101
```

\$1, \$2, \$3 lässt sich dabei in der Reihenfolge vertauschen oder teilweise weglassen:

```
....awk '{print $1,$3}'
```

Output:

```
tp 192.168.2.101
```

Wenn man es im Browser ausgeben möchte, lässt man die # im Skript weg und ruft die Seite <http://Router/user/wlan.html> auf.

Traffic-Schnipsel

ifconfig Traffic-Monitor

Was machts? ifconfig gibt die Konfiguration der Interfaces zurück.

```
~ # ifconfig
br0      Link encap:Ethernet  HWaddr 00:12:34:56:78:99
         inet addr:10.25.134.1  Bcast:10.25.134.255  Mask:255.255.255.0
         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
         RX packets:86779  errors:0  dropped:0  overruns:0  frame:0
         TX packets:112449  errors:0  dropped:0  overruns:0  carrier:0
         collisions:0  txqueuelen:0
         RX bytes:7236365 (6.9 MiB)  TX bytes:111529135 (106.3 MiB)

usw. ...
```

... bastelt man sich ein wenig Code um diese Ausgabe, dann kann man sich damit einen sehr einfachen Traffic-Monitor bauen.

Skript-Schnipsel

```
#!/bin/sh
while [ 1 ]; do
  curr_input_bytes=$(ifconfig $(route | awk '/default/ {print$8}') | awk -F' +|:' '/bytes/ {print$2})
  curr_output_bytes=$(ifconfig $(route | awk '/default/ {print$8}') | awk -F' +|:' '/bytes/ {print$3})
  if [ "$old_input_bytes" -eq '' ]; then
    old_input_bytes=0;
  fi
  if [ "$old_output_bytes" -eq '' ]; then
    old_output_bytes=0;
  fi
  let "delta_input_bytes = $curr_input_bytes - $old_input_bytes"
  let "delta_output_bytes = $curr_output_bytes - $old_output_bytes"
  let "input_speed = $delta_input_bytes / 60"
  let "output_speed = $delta_output_bytes / 60"
  old_input_bytes=$curr_input_bytes
  old_output_bytes=$curr_output_bytes
  clear
  echo -e "Input:\t\t\t$curr_input_bytes\tBytes "
  echo -e "Output:\t\t\t$curr_output_bytes\tBytes"
  echo -e "Input speed:\t\t$input_speed\tBytes/second"
  echo -e "Output speed:\t\t$output_speed\tBytes/second"
  sleep 60
done
```

Anstatt sich das Ganze auf der Konsole anzusehen, kann man es natürlich auch in eine HTML-Seite schicken.<pr> Man muss nur die Ausgabe umleiten und das Verzeichnis /tmp/www anlegen.

```
#!/bin/sh
mkdir /tmp/www
while [ 1 ]; do
  curr_input_bytes=$(ifconfig $(route | awk '/default/ {print$8}') | awk -F' +|:' '/bytes/ {print$2})
  curr_output_bytes=$(ifconfig $(route | awk '/default/ {print$8}') | awk -F' +|:' '/bytes/ {print$3})
  if [ "$old_input_bytes" -eq '' ]; then
    old_input_bytes=0;
  fi
  if [ "$old_output_bytes" -eq '' ]; then
    old_output_bytes=0;
  fi
  let "delta_input_bytes = $curr_input_bytes - $old_input_bytes"
  let "delta_output_bytes = $curr_output_bytes - $old_output_bytes"
  let "input_speed = $delta_input_bytes / 60"
  let "output_speed = $delta_output_bytes / 60"
  old_input_bytes=$curr_input_bytes
  old_output_bytes=$curr_output_bytes
  echo -e '
<html><body>
<head>
<META HTTP-EQUIV='Refresh' CONTENT='60'>
</head>
<table>
  <tr bgcolor=#c1c1c1>
    <td></td>
    <td>Gesamt<br>[byte]</td>
    <td>Geschwindigkeit<br>[byte/sek]</td>
  </tr>
  <tr>
    <td bgcolor=#c1c1c1 >Eingehend</td>
    <td>'$curr_input_bytes'</td>
    <td>'$input_speed'</td>
  </tr>
</table>
'
```

Skript-Schnipsel

```
<td bgcolor=#c1c1c1>Ausgehend</td>
<td>'$curr_output_bytes'</td>
<td>'$output_speed'</td>
<tr>
</table>
<body><html>' > /tmp/www/traffic.htm
sleep 60
done
```

Das Ganze findet man dann unter <http://routerIP/user/traffic.htm>

Wie kann ich dieses Dokument schützen? Ich möchte, dass man es nur anschauen kann, wenn man im Web interface eingeloggt ist. Kann das hier jemand schreiben?

Wenn man dieses Schnipsel nicht traffic.htm sondern traffic.asp nennt, dann sollte der Schutz auch funktionieren.

Anzeige der maximalen Verbindungen

Mit folgendem Schnipsel kann man sich die Anzahl der aufgebauten Verbindungen anzeigen lassen: `wc -l < /proc/net/ip_conntrack` wenn man sich nur für einen Port oder einen Portrange interessiert, geht das wie folgt: Einzelner Port: `grep 4661 /proc/net/ip_conntrack | wc -l`

Portrange: `grep 46[61-72] /proc/net/ip_conntrack | wc -l`

Idle Time umgehen

Falls man vom Provider getrennt wird, wenn man keinen Traffic erzeugt:

```
#!/bin/sh
while [ 1 ]; do
  ping -c 5 www.example.com > /dev/null # ping 5 packets
  sleep 300 # wait 5 minutes
done
```

Falls dieses nicht funktioniert (der Provider ignoriert ICMP Pakete) dann benutze:

```
#!/bin/sh
while [ 1 ]; do
  wget http://www.example.com/ -O /tmp/index -q # download index file
  sleep 300 # wait 5 minutes
done
```

Kleines Sendmail-Programm (ohne Fehlerabhandlung)

Falls man keine Lust hat, ein Sendmail-Programm zu installieren.

```
#!/bin/sh
while read a; do
```

Anzeige der maximalen Verbindungen

Skript-Schnipsel

```
    echo $a >> body.$$
done
while getopts s:r:m:h: name; do
    case $name in
        s) server="$OPTARG";;
        r) rec="$OPTARG";;
        m) mailer="$OPTARG";;
        h) subject="$OPTARG";;
    esac
done
#echo $server
#echo $rec
#echo $mailer

echo "EHLO mailer" >> mail.$$
echo "MAIL FROM: $mailer" >> mail.$$
echo $rec | tr ' ' '\n' | while read x; do echo "RCPT TO: $x" >> mail.$$ ; done
echo "DATA" >> mail.$$
echo "Subject: $subject " >> mail.$$
cat body.$$ >> mail.$$
echo "." >> mail.$$
echo "QUIT" >> mail.$$

cat mail.$$ | nc $server 25

rm mail.$$
rm body.##
```

Syslog per E-Mail

Das Ganze ist in Zusammenarbeit mit Victor F. erfolgreich getestet worden der passende Thread findet sich [hier](#).

Die folgenden Skripte entweder als Custom Script speichern oder im jffs ablegen und zum Beispiel per cron ausführen lassen.

Bei SMTP over telnet können keine Anhänge versendet werden.

Syslog muss eingeschaltet sein.

```
#!/bin/sh

EMAIL_TO="Empfängeradresse"
EMAIL_SUBJECT="Betreff"

# Was soll in der Mail stehen?
EMAIL_content='cat /var/log/messages' #z. B. das Syslog

# HELO oder EHLO benutzen je nach Mailserver.
(echo HELO <Begrüßung am Mail-Server> #z. B. t-online.de
sleep 3
echo MAIL FROM: Absenderadresse
sleep 1
echo RCPT TO: $EMAIL_TO
sleep 1
echo "DATA
From: Absenderadresse
To: $EMAIL_TO
```

Kleines Sendmail-Programm (ohne Fehlerabhandlung)

Skript-Schnipsel

```
Subject: $EMAIL_SUBJECT
$EMAIL_content"
sleep 1
echo '.'
sleep 1
echo QUIT) | nc smtp.server 25      # hier den SMTP-Server und Port eintragen
```

Alternativ mit Angabe von Datum und anschließendem Reboot des Routers.

```
#!/bin/sh
# Skript verschickt Logfile per E-Mail
# script sends a given logfile to a specified mail
# account via SMTP and restarts router afterwards
# posted 2009 in DD-WRT forum
# thanks for help to pepe of DD-WRT forum
#
# Changelog:
# 01.05.09 init
# 09.05.09 inject logfile into DATA
# 10.05.09 abstracted over logfile and stuff
#
# Bugs:
# Won't work with web.de?
# Won't work with gmx.de?

# Allgemeine Angaben
LOGFILE=/var/log/messages
MAILSERVER="smtp.mailserver.org"
RESET_ROUTER=/usr/sbin/reboot
DATE="`date +%d`. `date +%B` `date +%Y` `date +%R`"
MARKER="### Report from $DATE ends here ###"

# Absenderdaten, je nach Mailserver fiktiv
EMAIL_FROM="router@dd-wrt.com"

# Empfaengerdaten
EMAIL_TO="user@mailserver.org"
EMAIL_SUBJECT="Syslog Report $DATE"

# ----- Ab hier nichts mehr ändern -----"
# ----- nothing to change from here -----"

# Setze Marker in Log-Datei
echo $MARKER >> $LOGFILE

# Logfile lesen
EMAIL_content=`cat $LOGFILE`

# Kontaktaufnahme mit Mailserver
(echo HELO $MAILSERVER
sleep 6
echo MAIL FROM: $EMAIL_FROM
sleep 1
echo RCPT TO: $EMAIL_TO
sleep 1
echo "DATA"
sleep 1
echo "From: $EMAIL_FROM
To: $EMAIL_TO
Subject: $EMAIL_SUBJECT

$EMAIL_content"
```


Skript-Schnipsel

```
sleep 1  
echo '.'  
sleep 1  
echo QUIT) | nc $MAILSERVER 25  
$RESET_ROUTER
```

Anstelle von

```
| nc $MAILSERVER 25
```

kann auch

```
| telnet $MAILSERVER 25
```

verwendet werden

Sämtliche E-Mail-Daten wie Empfängeradresse, SMTP-Server, Begrüßung und Absenderadresse sind durch reale Daten zu ersetzen. Ebenso ist es ratsam, das Ganze vorher manuell zu testen, da sich nicht alle Mail-Server gleich verhalten.

Hier mal eine Liste von Servern die zu funktionieren zu scheinen.

- mail.mymail.ch 25
- mail.oleco.de 25
- mail.arcor.de 25
- mgate.chello.at 25
- mx.freenet.de 25