TCP/IP-Administration unter UNIX

Falko Dreßler,

Regionales Rechenzentrum

falko.dressler@rrze.uni-erlangen.de

Überlick

- UNIX und TCP/IP
- Sockets
- Konfiguration der Netzwerkschnittstelle
  - netstat
  - ifconfig
  - ndd
- inetd
- Routing
  - route
  - routed
  - gated
- Nameservicekonfiguration
  - nslookup
  - named
- Sonstige Tools
  - ping
  - traceroute
  - arp
- Fehlersuche

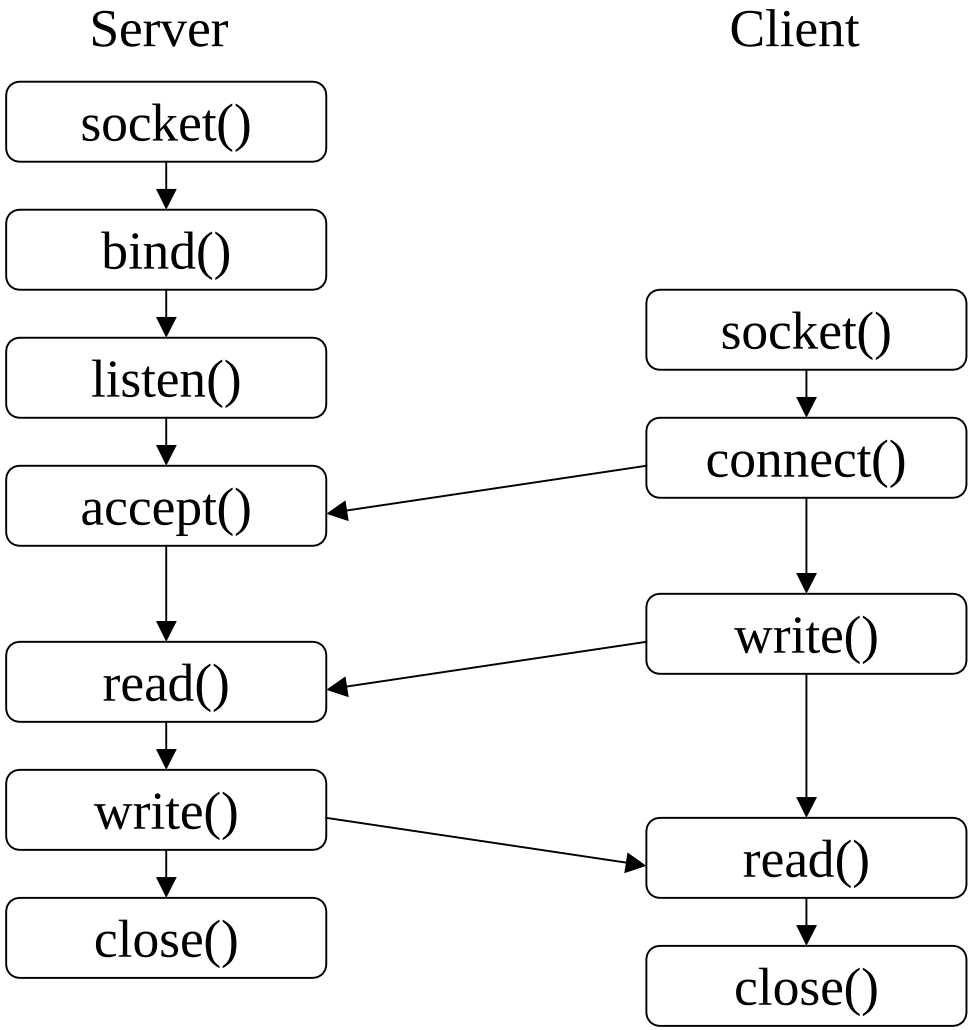
UNIX und TCP/IP

- Erste TCP/IP-Implementation unter 4.2BSD 1982
- Die Verbreitung von UNIX und TCPP/IP ging Hand in Hand
- Implementierung über Socket-Schnittstelle (BSD) oder Transport-Layer- (Streams-) Interface (SystemV)
- Die meisten Netzwerkanwendungen basieren auf dem Client-Server-Modell: Ein Server erbringt einen Dienst für einen Client. Solche Anwendungen sind unsymmetrisch, d.h.der Server wartet auf eine Anfrage von einem Client, erbringt den Dienst und wartet anschließend auf die nächste Anfrage. Server-Prozesse werden in der Regel durch einen sogenannten Daemon-Prozeß realisiert.
- Kommandos sind sehr vom verwendeten Betriebssystem abhängig. Sofern nicht anders gesagt, beziehen sich alle Beispiele auf Solaris. Im Zweifelsfall lese man die entsprechende man-Page.

Sockets

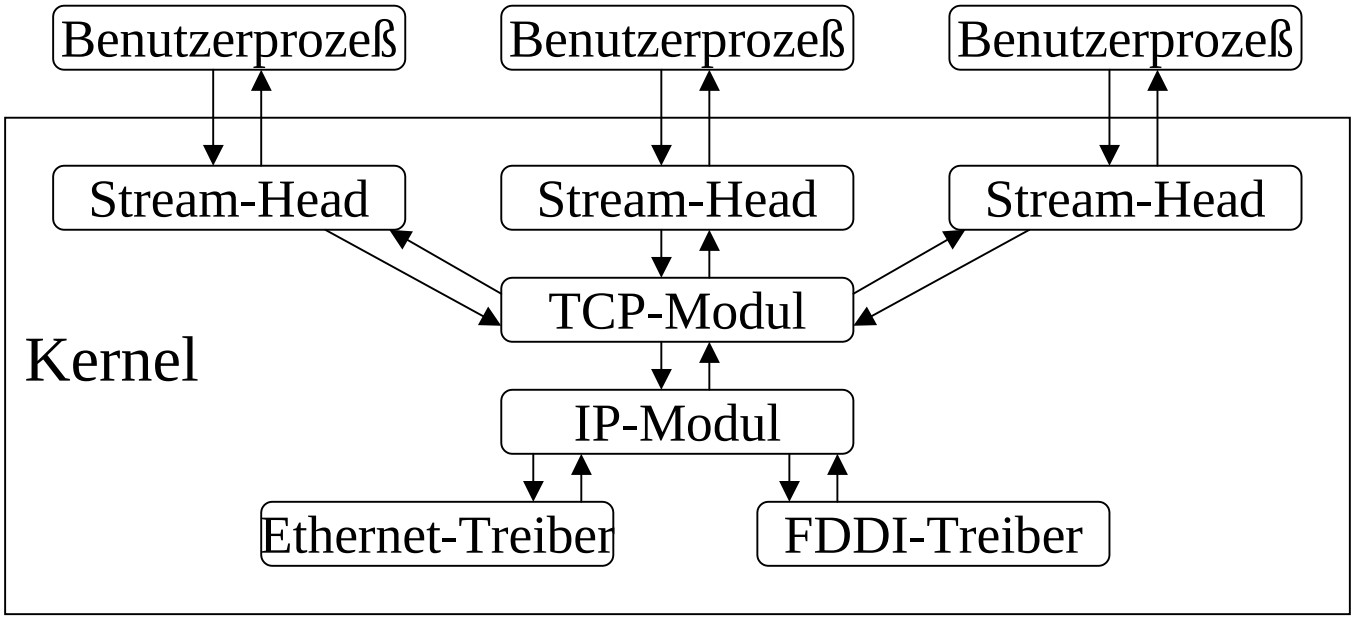
- In BSD-UNIX wird TCP/IP durch Socket-Schnittstelle realisiert
- Ein Socket ist ein Kommunikationsendpunkt auf Transportschichtebene. Wird bestimmt durch: Transportschichtprotokoll, IP-Adresse und Port. Eine Kommunikationsverbindung besteht immer aus 2 Sockets. Man nennt dies auch eine association. Bsp: {tcp, 131.188.3.150, 1022, 131.188.3.2, 22}
- Eine Socket-Schnittstelle stellt dir folgenden Operationen bereit:
  - socket(): erzeugt einen Socket. Dabei wird bereits das Transportschichtprotokoll festgelegt.
  - bind(): legt die lokale IP-Adresse fest.
  - listen(): wartet auf ankommende Verbindungen (passive open).
  - accept(): nimmt Verbindung an.
  - connect(): baut eine Verbindung auf (active open).
  - Auf den Socket kann mit den üblichen Fileoperationen (z.B. read() / write()) zugegriffen werden.

Socket-Schnittstelle: Ein Beispiel



SystemV Transport Layer Interface

- Streams-Konzept



- Zwischen Gerätetreiber (Hardwareschnittstelle) und Stream-Head (Benutzerschnittstelle) können weitere Module eingeschoben werden.
- Funktionalität der Programmierschnittstelle ähnlich wie bei Sockets.

netstat

netstat - Statusinformationen des Netzwerkes

Synopis:

```
netstat [ -anv ]
netstat [ -s | -g | -m | -p | -f address_family ] [-P protocol ] [ -n ]
netstat -i | -I interface [ interval ]
netstat -r [ -anv ]
netstat -M [ -ns ]
```

Beschreibung:

netstat zeigt die Inhalte von mehreren Netzwerk-Datenbankstrukturen in verschiedenen Formaten: Die erste Form zeigt für jedes Protokoll eine Liste der aktiven Sockets. Die zweite selektiert bestimmte Datenbankstrukturen, abhängig von den Optionen. Die dritte Form liefert laufende Verkehrsstatistiken der konfigurierten Interfaces. Die vierte liefert Routinginformationen und die letzte Form zeigt Multicastinformationen.

Optionen:

-a	zeigt den Status aller Sockets. Ohne diese Option werden nur aktive Sockets angezeit.
-v	‚verbose' Informationen.
-n	zeigt Netzwerkadressen als Nummern (keine Adreßauflösung).
-s	zeigt Protokoll-Statistiken.
-g	zeigt die Zugehörigkeit zu einer Multicastgruppe.
-m	Statistiken, die von Managementroutinen über den Puffer-Pool aufgezeichnet werden.
-p	zeigt die ARP-Tabelle.
-f address_family	zeigt Reports über die angegebene Adreßfamilie (inet, unix).
-P protocol	zeigt die Statistik nur für ein Protokoll (z.B. tcp).

netstat 2

-i	zeigt den Status aller konfigurierten Interfaces.
-I interface	zeigt Statistiken des angegebenen Interfaces.
-r	zeigt die Routingtabellen.
-M	zeigt die Multicast-Routingtabellen.

Ausgaben und Beispiele:

Zweite Form: Es wird für jeden aktiven Socket die lokale und die remote Adresse, die Sende- und Empfangswarteschlange (Bytes), das Protokoll und der interne Status des Protokolls angezeigt.

Das Format der Ausgabe für die Socketadresse ist „hostname.port“ bzw. „network.port“, falls eine Socketadresse ein Netzwerk bezeichnet. Nicht spezifizierte oder „Wildcard“-Adressen werden mit „*“ gekennzeichnet.

Dritte Form: Falls ein Intervall angegeben wurde, zeigt netstat eine Tabelle mit kummulierten Statistiken über die transferierten Pakete, die Fehler und die Kollisionen, die Netzwerkadressen des Interfaces und die maximale Paketgröße (MTU). Jede weitere Zeile zeigt inkrementelle Statistiken für das Intervall.

Vierte Form: Es wird die Routingtabelle ausgegeben, in der die verfügbaren Routen und deren Status verzeichnet sind. Jeder Eintrag besteht aus Zielhost oder Zielnetzwerk, dem Gateway, dem Status, der Anzahl der aktiven Verbindungen, der Anzahl der über diese Route übertragenen Pakete, sowie dem Interfacenamen, der für diese Route benutzt wird.

Die Statusinformationen bestehen aus folgenden Einträgen:

- U - Die Route ist „UP“
- G - Die Zieladresse ist ein Gateway
- D - Die Route wurde dynamisch durch ein Redirect erzeugt
- H - Die Zieladresse ist ein Host

Zusammen mit der Option „-v“ erhält man auch die Maske für die Routen.

netstat -a -P tcp	netstat -rn	netstat -s -P tcp
netstat -i	netstat -s	netstat -in 10

ifconfig

ifconfig - Konfiguration der Parameter eines Netzwerkinterfaces

Synopis:

```
ifconfig interface [[ address_family ] address ] [ parameters ]
```

Beschreibung:

ifconfig wir benutzt, um eine Adresse einem Netzwerkinterface zuzuweisen bzw. Netzwerkparameter eines Interfaces zu verstellen. ifconfig wird während der Boot-.Phase des Rechners benutzt, um Netzverbindungen herstellen zu können. ifconfig ohne Parameter aufgerufen, liefert die aktuellen Einstellungen des Interfaces. Beim Aufruf als root, wird zusätzlich die MAC-Adresse ausgegeben.

interface	Interfacename (z.B. le0, le1 - Ethernetinterfaces bei Sun)
address_family	Adreßfamilie, z.Zt. nur inet sinnvoll
address	IP-Adresse des Interfaces (als Nummer oder, wenn Namensauflösung läuft, als Name)

Parameter:

up/down	markiert ein Interface als up bzw. down.
[-]arp	Ein- / Ausschalten des ARP-Protokolls.
metric n	setzt die Routing-Metrik des Interfaces auf n.
[-]debug	Ein- / Ausschalten des Debugging-Modus.
netmask mask	setzt die Netzmaske auf mask.
dest_address	setzt die Adresse der Gegenstelle bei Punkt-zu.-Punkt-Verbindungen.
broadcast address	setzt die Broadcastadresse auf address.

ifconfig 2

Ausgaben und Beispiele:

```
% ifconfig hme0
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255
# ifconfig hme0
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255
        ether 8:0:20:92:e9:4

% ifconfig -a
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
        inet 127.0.0.1 netmask ff000000
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255
lane0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 192.44.88.11 netmask ffffff00 broadcast 192.44.88.255

# ifconfig hme0 inet 131.188.3.150 netmask 255.255.255.0 broadcast 131.188.3.255

# ifconfig hme0 down
# ifconfig hme0 up
```

ifconfig - logical Interfaces

- Ab Solaris 2.5 kann man einer Netzwerkkarte mehrere IP-Adressen zuweisen. Dies ist u.a. dann sinnvoll, wenn einem wichtigen Dienst (z.B. WWW-Server) eine feste IP-Adresse zugewiesen werden soll, man aber dynamisch zwischen mehreren Rechnern umschalten möchte (Ausfallsicherheit).

```
# ifconfig -a | grep hme0
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255

# ifconfig hme0:1 131.188.3.254 netmask 255.255.255.0 broadcast 131.188.3.255

# ifconfig -a | grep hme0
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255
hme0:1: flags=842<BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.254 netmask ffffff00 broadcast 131.188.3.255

# ifconfig hme0:1 up

# ifconfig -a | grep hme0
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.150 netmask ffffff00 broadcast 131.188.3.255
hme0:1: flags=843<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 131.188.3.254 netmask ffffff00 broadcast 131.188.3.255
```

ndd

ndd - Lesen und Setzen von Treiberparametern

Synopis:

```
ndd [ -set ] driver parameter [ value ]
```

Beschreibung:

ndd holt bzw. setzt Konfigurationsparameter von Kerneltreibern. Bisher werden nur die Treiber des TCP/IP-Stacks unterstützt.

driver	der zu konfigurierende Gerätetreiber, z.B. /dev/tcp, /dev/hme
parameter	der abzufragende bzw. zu ändernde Wert. ‚?' liefert eine Liste aller möglichen Werte.

Optionen:

-set	setzt den Wert für parameter auf value.

Beispiele:

```
# ndd /dev/ip \?
ip_forwarding                  (read and write)
ip_respond_to_address_mask_broadcast(read and write)
...
# ndd /dev/ip ip_forwarding
0
# ndd -set /dev/ip ip_forwarding 1
```

inetd

- Um nicht für jeden Dienst ständig einen Server-Prozeß laufen zu lassen, wurde der inetd eingeführt.
- Der inetd wartet an einer Reihe von Sockets und startet bei Bedarf den richtigen Daemon.
- Konfiguration über `/etc/inetd.conf`:

```
ftp     stream      tcp         nowait      root        /usr/sbin/in.ftpd       in.ftpd
telnet  stream      tcp         nowait      root        /usr/sbin/in.telnetd    in.telnetd
echo    stream      tcp         nowait      root        internal
echo    dgram       udp         wait        root        internal
```

- Bedeutung der Felder:
  - ftp: bezeichnet den Dienst. Muß in `/etc/services` oder der entsprechenden NIS-Tabelle stehen.
  - stream/dgram: gibt an, ob es sich um einen Stream- oder Datagram-Dienst handelt.
  - tcp/udp: gibt das verwendete Protokoll an.
  - wait/nowait: bei wait erlaubt der inetd erst wieder einen Zugriff auf den Dienst, wenn die Bearbeitung der letzten Anfrage abgeschossen wurde.
  - root: bezeichnet die UID, unter der Daemonprozeß gestartet wird.
  - /usr/sbin/in.ftpd: ist der Server, der gestartet werden soll.
  - Der Rest sind Argumente für den Server.
- Nach einer Änderung der inetd.conf-Datei muß der inetd neu initialisiert werden. Dies geschieht durch ein gesendetes SIGUP.

route

route - manuelles Modifizieren der Routingtabelle

Synopis:

```
route [ -fn ] add | delete [ host | net ] destination [ gateway [ metric ]]
```

Beschreibung:

Mit route können Routingeinträge von Hand erstellt oder gelöscht werden. Dieses Kommando ist privilegiert und darf nur vom Superuser ausgeführt werden.

Optionen:

-f alle Einträge der Routingtabelle werden gelöscht (flush).

-n es wird keine Nameserviceauflösung vorgenommen.

Beispiele:

```
# route add 131.188.74.14 131.188.3.1 1
# route add net 131.188.74.0 131.188.3.1 1
# route delete 131.188.74.14 131.188.3.1
# route delete net 131.188.74.0 131.188.3.1
# route -fn
# route add default 131.188.3.1 1
```

routed, in.routed

routed - Netzwerk-Routingdaemon

Synopis:

```
in.routed [ -gqstv ] logfile
```

Beschreibung:

Normalerweise wird der routed beim Hochfahren des Rechners gestartet, um die Netzwerk-Routingtabellen zu verwalten. Hierbei wird das Routing Information Protocol (RIP) benutzt.

Der Routingdaemon überwacht den UDP-Port 520 auf ankommende RIP-Pakete. Diese Pakete beinhalten auch einen „hop count“ (Metrik), um so die Qualität der Routen bewerten zu können.

Optionen:

-g	es wird eine default-Route generiert.
-s	es werden Routinginformationen an andere Stationen abgegeben.
-q	es werden Routinginformationen nur angenommen, aber keine abgegeben.
-t	alle eingehenden oder abgehenden Pakete werden ausgegeben.
-v	es wird ein Logfile angelegt, in dem Änderungen der Routingtabelle mit Zeitstempel dokumentiert werden.

gated

gated - Gateway-Routingdaemon

Synopis:

```
gated [-c] [-C] [-n] [-N] [-t trace_options] [-f config_file] [trace_file]
```

Beschreibung:

gated ist ein Routingdaemon, der mehrere Routingprotokolle versteht. Er soll den routed und andere Routingdaemons ersetzten. Z.Zt. Kann der gated die Protokolle RIP, EGP, BGP, OSPF und HELLO behandeln. Die Konfiguration befindet sich standardmäßig in der Datei `/etc/gated.conf`.

Optionen:

-c	es wird nur die Konfigurationsdatei geparsed. Werden keine Fehler gefunden, wird ein dump-file angelegt.
-C	es wird nur die Konfigurationsdatei geparsed. Werden keine Fehler gefunden, ist der exit-code 0, sonst 1.
-n	die Routingtabellen des Kernels werden nicht verändert.
-N	der gated wird nicht im Daemonmodus gestartet.
-t	setzt die trace-Option.
-f	erlaubt die Verwendung einer anderen Konfigurationsdatei.

Konfiguration:

Die Konfiguration des gated ist recht umfangreicht. Eine Anleitung gibt die man-Page zu gated-config und das folgende einfacge Beispiel.

gated - Beispiel

- Konfigurationsdatei auf der cssun:

```
interfaces {
     interface all simplex;
};
rip on {
     interface 131.18.3.9  metricin 1 metricout 1;
     interface 131.1888.71.9            metricin 3 metricout 5;
     interface 131.188.3.58             noripin noripout;
};
static {
     host 131.188.3.58     gateway 127.0.0.1 noinstall;
     host 131.188.3.9      gateway 127.0.0.1 noinstall;
     host 131.188.71.9     gateway 127.0.0.1 noinstall;
}
export proto rip {
     proto default;
     proto static { all metric 1; };
     proto direct restrict;
     proto rip {
             all restrict;
             0.0.0.0       mask 255.255.255.0;
             131.188.5.0   mask 255.255.255.0;
     };
};
```

- Bedeutung:
  - Alle Schnittstellen werden als simplex behandelt, d.h. auch wenn keine Daten empfangen werden, gelten sie als up.
  - Über das 3.9er-Interface werden Routen mit der Metrik 1, über das 71er mit der Metrik 5 und über das 3.58er gar nicht verbreitet.
  - Die Routen zu den eigenen Interfaces werden als statische Routen implementiert, aber nicht im Kern eingetragen.
  - Über RIP werden der default-Route und alle statischen Routen exportiert, nicht aber die direkt angeschlossenen Netzwerke und von den per RIP gelernten Netze nur das 5er-Netz.

Nameservice-Konfiguration

- In der Datei `/etc/hosts` können Abbildungen IP-Adresse <-> Rechnername statisch konfiguriert werden:

```
127.0.0.1     localhost
131.188.3.150 lisa.rrze.uni-erlangen.de lisa
192.44.88.11  lisa-gigabit lisa-gigabit.rrze.uni-erlangen.de
```

- Über die Datei `/etc/nsswitch.conf` wird die Auswertereihenfolge der verschiedenen Informationsdienste festgelegt:

```
hosts:        files dns
networks:     nis [NOTFOUND=return] files
protocols:    files
netmasks:     nis [NOTFOUND=return] files
```

- In der Datei `/etc/resolv.conf` werden Nameserver und Suchreihenfolge verschiedener Sub-Domains festgelegt:

```
nameserver    131.188.3.2
nameserver    131.188.2.45
domain        rrze.uni-erlangen.de
search        rrze.uni-erlangen.de uni-erlangen.de
```

- Rechner können entweder als Nameservice-Clients (kein aktiver Namedaemon), Caching-Nameserver (aktiver Namedaemon, aber keine lokalen Zonendateien) oder Secondary-Nameserver (aktiver Namedaemon mit lokalen Zonendateien) konfiguriert werden.

nslookup

nslookup - aktive Abfragen eines Nameservers

Synopis:

```
nslookup [ -option ] host [ server ]
nslookup
```

Beschreibung:

nslookup sendet Anfragen an Nameserver, bereitet die Antwort auf und zeigt dann die gewünschten Informationen an. Das Programm kann sowohl interaktiv als auch nicht-interaktiv genutzt werden. Wird nslookup ohne Parameter aufgerufen, gelangt man in den interaktiven Modus.

Interessante Kommandos:

Die Kommandos von nslookup sind sehr vielfältig. Nähere Informationen findet man in der man-Page.

host	gibt den Eintrag für host aus.
host server	dito, aber server wird als Nameserver benutzt.
help	eine Hilfe.
server host	ab sofort wird der Server host für Abfragen benutzt.
set option	setzt Optionen:
domain=domainname	setzt den Default-Domainname auf domainname.
type=searchtype	setzt den Suchtyp (z.B. type=mx für Ausgabe der Mailexchanger).
ls domain	listet eine ganze Domain auf. Die Ausgabe kann man mit „>“ in eine Datei umlenken.

caching named

- Konfiguration eines caching named (`/etc/named.conf`):

```
options {
        directory "/var/named";
        named-xfer "/usr/sbin/named-xfer-8";
        forward only;
        forwarders {
                131.188.3.72;
                131.188.3.73;
        };
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "named.local";
};
```

- Die Zeile mit den forwarders gibt die verwendeten Nameserver an, von dem die nicht im Cache befindlichen Informationen geholt werden.

secondary named

- Konfiguration eines secondary named (`/etc/named.conf`):

```
options {
        check-names master fail;
        check-names slave fail;
        check-names response ignore;
        directory "/var/named";
        named-xfer "/usr/sbin/named-xfer-8";
        forwarders {
                131.188.3.2;
                131.188.3.45;
        };
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "named.local";
};
zone "uni-erlangen.de" {
        type slave;
        masters {
                131.188.3.2;
        };
        file "uni-erlangen.de.BCK";
};
zone "188.131.IN-ADDR.ARPA" {
        type slave;
        masters {
                131.188.3.2;
        };
        file "188.131.IN-ADDR.ARPA.BCK";
};
```

ping

ping - sendet ICMP ECHO_REQUEST Pakete an Netzwerk-Hosts

Synopis:

```
ping  host [timeout]
ping  [-s] [-dlLnrRv] [-i interface] [-I interval]
      [-t ttl] host [packetsize] [count]
```

Beschreibung:

ping benutzt das ICMP ECHO_REQUEST Datagram, um ein ICMP ECHO_RESPONSE Datagram des angegebenen Hosts zu generieren. In der ersten Form (ohne Optionen) wird, falls der Host antwortet, „host is alive“ ausgegeben. Falls nicht, wird nach einem optionalen Timeoout (Standard ist 20 Sek.) eine entsprechende Meldung erstellt. In der zweiten Form (mit „-s“) wird jede Sekunde (außer „-I interval“) ein Paket an den Host geschickt, wobei Paketgröße und Anzahl konfigurierbar sind. Wenn ping durch ein SIGINT abgebrochen wird oder die Anzahl der zu sendenden Paket erreicht wurde, wird eine Statistik über Antwortzeiten und Paketverluste erstellt.

Optionen:

-r keine Verwendung der Routingtabellen.

-R verwendet die „record route“-Option. Zusammen mit „-v“ wird der Weg des Paketes angezeigt.

-i interface benutze Interface interface (sinnvoll nur bei Multicast).

-n keine Nameserviceauflösung.

Beispiele:

```
# ping faui45.informatik.uni-erlangen.de
# ping -s faui45.informatik.uni-erlangen.de
# ping -svRn www.sun.com
```

traceroute

traceroute - gibt die Route aus, über die ein Netzwerk-Host erreichbar ist.

Synopis:

```
tarceroute [-m max_ttl] [-n] [-p port] [-q nqueries] [-r]
           [-s src_addr] [-g addr] [-w waittime] host [packetsize]
```

Beschreibung:

traceroute benutzt das „time to live“-Feld im IP-Header, um eine ICMP TIME_EXCEEDED-Antwort eines jeden Gateways auf der Route zu einem Host zu generieren. Hierzu werden drei Testpakete (oder „-q“) mit einer TTL von 1 verschickt. Das angesprochene Gateway antwortet mit einer ICMP TIME_EXCEEDED-Meldung. Somit erhält traceroute die IP-Adresse des ersten Gateways. Es wird jetzt solange die TTL um 1 erhöht, bis eine ICMP PORT_UNREACHABLE-Meldung ankommt (Host ist erreicht) oder die maximale TTL („-m“) erreicht ist. Die Ausgabe von !H bedeutet, daß der Host nicht erreichbar ist. Ein (ttl=255!), daß das TTL-Feld im Antwortpaket von der Erwartung abweicht.

Optionen:

-m max_ttl	setzt die maximale TTL.
-n	keine Nameserviceauflösung.
-p port	setzt die Portnummer.
-q nqueries	setzt die Anzahl der Pakete pro Durchgang.
-s src_addr	setzt die Quelladresse bei Hosts mit mehreren Interfaces.
-g addr	benutzen der „Loose Source Record Route“-Option.
-v	Ausgabe der ICMP Pakete, sofern nicht TIME_EXCEEDED oder UNREADABLE.
-w waittime	setzt die Wartezeit für eine Antwort.

arp

arp - Anzeigt und Verwaltung des ARP-Caches

Synopis:

```
arp hostname
arp -a [ unix [ kmem ]]
arp -d hostname
arp -s hostname ether-address [ temp ] [ pub ] [ trail ]
arp -f file
```

Beschreibung:

arp zeigt und modifiziert die ARP-Tabellen. Ohne Option zeigt das Programm den z.Zt. gültigen Eintrag für den angegebenen Host.

Optionen:

-a	alle ARP-Einträge werden aufgelistet.
-d	löscht den Eintrag für den angegebenen Host.
-s	setzt den Eintrag für den Host. Der Eintrag ist permanent, falls nicht temp angegeben wird. pub gibt an, daß der Eintrag bei Anfragen weitergegeben wird. trail zeigt an, daß „trailer encapsulations“ zum Host gesendet werden können.
-f file	die Datei file wird eingelesen und in die ARP-Tabelle eingetragen.

Hinweis:

Mit arp -a bekommt man dieselben Informationen wie mit netstat -p. Mit netstat -pn kann man aber die Adreßauflösung unterdrücken.

Fehlersuche

- Ist das Interface korrekt konfiguriert und UP?

```
kawa{root}[~]# ifconfig -a
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
     inet 127.0.0.1 netmask ff000000
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
     inet 131.188.3.154 netmask ffffff00 broadcast 131.188.3.255
     ether 8:0:20:96:8:bd
```

- Läßt sich das eigene Interface anpingen?

```
kawa{root}[~]# ping 131.188.3.154
131.188.3.154 is alive
```

- Können Rechner im eigenen Subnetz erreicht werden?

```
kawa{root}[~]# ping 131.188.3.150
131.188.3.150 is alive
```

- Gehen Daten über die Netzwerkschnittstelle?

```
kawa{root}[~]# netstat -i 2
    input   hme0      output           input  (Total)    output
packets errs  packets errs  colls  packets errs  packets errs  colls
42738774 0     32558669 0     0      45296582 0     35116477 0     0
12      0     2       0     0      16      0     6       0     0
9       0     1       0     0      13      0     5       0     0
```

Fehlersuche 2

- Bei Problemen im eigenen Subnetz: Wie schaut der ARP-Eintrag aus?

```
kawa{root}[~]# arp -a
Net to Media Table
Device   IP Address               Mask      Flags   Phys Addr
------ -------------------- -------------- ----- ---------------
hme0   southern.gate.uni-erlangen.de 255.255.255.255       00:50:0f:10:a6:17
hme0   herkules.rrze.uni-erlangen.de 255.255.255.255       08:00:20:96:03:0d
hme0   lisa.rrze.uni-erlangen.de 255.255.255.255       08:00:20:92:e9:04
hme0   kawa.rrze.uni-erlangen.de 255.255.255.255 SP    08:00:20:96:08:bd
hme0   BASE-ADDRESS.MCAST.NET 240.0.0.0       SM    01:00:5e:00:00:00
```

- Ist ein default-Route bekannt?

```
kawa{root}[~]# netstat -rn
Routing Table:
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
131.188.3.0          131.188.3.154         U        3  32241  hme0
224.0.0.0            131.188.3.154         U        3      0  hme0
default              131.188.3.1           UG       0  52187
127.0.0.1            127.0.0.1             UH       02464303  lo0
```

- Im Fehlerfall: Läuft ein Routingdaemon oder soll der default-Route statisch konfiguriert sein?
- Läßt sich der Router anpingen?

```
kawa{root}[~]# ping 131.188.3.1
131.188.3.1 is alive
```

Fehlersuche 3

- Kann ein Rechner außerhalb des eigenen Subnetzes erreicht werden?

```
kawa{root}[~]# raceroute -n 131.188.128.54
traceroute to 131.188.128.54 (131.188.128.54), 30 hops max, 40 byte packets
 1  131.188.3.1  2 ms  1 ms  1 ms
 2  131.188.10.1  2 ms  2 ms  3 ms
 3  131.188.7.128  2 ms  1 ms  1 ms
 4  131.188.128.54  3 ms  1 ms  1 ms
```

- Funktioniert der Nameservice?

```
kawa{root}[~]# nslookup www.sun.com
Server:  localhost
Address:  127.0.0.1
Name:    wwwwseast.usec.sun.com
Address:  192.9.49.30
Aliases:  www.sun.com
```

- Kann das Zielsystem auf dem gewünschten Port angesprochen werden?

```
kawa{root}[~]# telnet mailhub smtp
Trying 131.188.71.14...
Connected to mailhub.rrze.uni-erlangen.de.
Escape character is '^]'.
220 max5.rrze.uni-erlangen.de (Mail*Hub TurboSendmail) ESMTP Service ready
quit
221 Until later [131.188.3.154]
Connection closed by foreign host.
```