6.1 Server/Superserver

6.1.1 Vorbereitung

Zuerst sollen einfache Netzwerkdienste eingerichtet und getestet werden. Dazu installiert man das Paket `inetd`. Es wird konfiguriert in der Datei `/etc/inetd.conf`. In dieser Datei ist zunächst von allen Zeilen, die mit `echo`, `discard`, `daytime`, `time` und `chargen` beginnen, das Kommentarzeichen # am Beginn zu entfernen. Mit dem Werkzeug `netstat -ln` kann man nun sehen, welche Ports geöffnet wurden.

6.1.2 Client und Server, Service und Portnummer

TCP/IP ermöglicht grundsätzlich Verbindungen in beiden Richtungen (vollduplex).[1] Trotzdem gibt es stets eine Seite, die die Verbindung hervorruft (initiiert) und eine Seite, die auf den Verbindungswunsch beanwortet. Die erste Seite heißt *Client*, die andere *Server*.

Erst das Paar aus IP-Adresse und Portnummer ergibt zusammen eine eindeutigen Punkt im Netz. Die IP-Adresse und Portnummer des Servers müssen vor Beginn der Verbindung für den Client bekannt sein (*well-known*). Die Adresse und Portnummer des Clients dagegen erfährt der Server erst, wenn der Client die Verbindung initiiert.

Es gibt eine Konvention, an welcher Portnummer eines Systems standardmäßig welcher Dienst anzutreffen sein kann. Auf Linux und vielen anderen Systemen findet man eine Zuordnungstabelle in der Datei `/etc/services`. Oft empfiehlt es sich, Portnummern nach dieser Tabelle zu vergeben.[2]

6.1.3 Server testen mit `netcat`

`netcat` ist ein Client, mit dem man Server manuell testen kann:

Terminal
```
schueler@debian964:~$ netcat localhost daytime
Mon Jan 14 15:06:32 2013
```

Erster Parameter ist die IP-Adresse, zweiter Parameter die Portnummer oder der Dienstname des Ziels. Hier wurde der `daytime`-Service getestet, der durch den `inetd`-Superserver eingerichtet worden war.

Mit den folgenden Befehlen (Nutzereingabe in Fettdruck) erhält man die Titelseite des Webservers an Port 80 des Systems mit der IP 10.1.1.1:

Terminal
```
schueler@debian964:~$ netcat -C 10.1.1.1 80
GET / HTTP/1.0
 ↵  ↵
HTTP/1.1 200 OK
Content-Length: 6531
Content-Type: text/html
Content-Location: http://10.1.1.1/Default.htm
...
Connection closed by foreign host.
```

Bekannterweise kann man TCP auffassen als die Übertragung eines Buchstaben-Stroms (Char-Stream). `netcat` leitet diesen Strom als Client vom lokalen System ins Netz. So wie `cat` eine Durchverbindung ist und `tee` ein T-Stück, so kann man netcat als ein L-Stück ins Netz ansehen. Weitere Dokumentation zu `netcat` findet man unter:
`http://www.jfranken.de/homepages/johannes/vortraege/netcat_inhalt.de.html`

[1] Der Vorgänger von TCP erlaubte nur einseitige Verbindungen. Damals gab es zwei Ports für jeden Dienst. Daher gibt es heute meistens ungerade Portnummern bei grundlegenden Diensten.

[2] Es sei denn, man möchte einen Server verstecken

6.1.4 Server testen mit `telnet`

Der ursprüngliche Zweck des telnet-Dienstes war es, das Arbeiten auf der Konsole entfernter Rechner, z.B. für die Fernwartung zu ermöglichen. Zeichen für Zeichen wurde ge-echo-t, außerdem wurden auch die sonstigen Bildschirmausgaben zum Client geschickt. Da telnet das Passwort im Klartext überträgt, ist dieser Dienst längst durch ssh als sicherere Variante abgelöst worden. Der heutige Zweck des telnet-Dienstes liegt darin, dass auch mit dem telnet-Client nahezu 1:1-Zeichenübertragung möglich ist.

Der telnet-Client heißt `telnet`, der telnet-Server `telnetd`. Mit der Tastenkombination Strg - AltGr - 9 (bei deutscher Tastatur) kann man aus einer telnet-Verbindung aussteigen.

6.1.5 Eigene Server bauen mit inetd

Das Programm `inetd` ist ein so genannter Super-Server, der bestimmte Dienste bei Bedarf starten kann. Das lohnt sich vor allem für seltener genutzte Dienste, für die nicht dauernd ein eigener Prozess mitlaufen soll. Einige Serverdienste (`echo`, `discard`, `daytime`, `time` und `chargen`) sind bereits im `inetd` eingebaut. Andere Serverdienste kann man hinzunehmen. Der Vorteil von `inetd` liegt darin, dass die Netzwerkfunktionalität bereits eingebaut ist und die einzelnen aufgerufenen Serverprogramme nur über ihre Tastatureingabe und ihre Bildschirmausgabe kommunizieren müssen. Die folgenden Beispiele sollten am besten mit `netcat` getestet werden: Hier ein Beispiel für einen zusätzlichen Eintrag in `/etc/inetd.conf`:

```
1 # nut-Dienst an Port 3493:
2 #Port                    User    Programm   argv[0]    argv[1,2,3,...]
3 nut stream  tcp nowait  nobody /bin/echo /bin/echo So ein Quark.
```

Diese Beispiele sollten am besten mit `netcat` getestet werden:

Terminal

```
schueler@debian964:~$ su
Passwort:
root@debian964:~# apt-get install netcat
...
root@debian964:~# vi /etc/inetd.conf
... (Eintrag hinzufügen)
root@debian964:~# /etc/init.d/openbsd-inetd reload
Reloading internet superserver: inetd.
root@debian964:~# exit
schueler@debian964:~$ netcat localhost 3493
So ein Quark.
```

Hier sind weitere Beispiele:

```
1 # nut=3493, daap=3689, sieve=4190, tfido=60177, fido=60179
2 nut    stream tcp nowait nobody /bin/cat        /bin/cat /etc/motd
3 daap   stream tcp nowait nobody /bin/sh         /bin/sh
4 sieve  stream tcp nowait nobody /bin/sh         /bin/sh -c ls
5 tfido  stream tcp nowait nobody /home/schueler/a.out /home/schueler/a.out
6 fido   stream tcp nowait nobody /usr/bin/tee    /usr/bin/tee /tmp/aus.txt
```

Wenn weitere, eigene Beispiele nicht wie erhofft funktionieren, so kann das an Zwischenpuffern im benutzten Dienstprogramm liegen. Bei der Programmierung eigener Server sollte man diese Zwischenpuffer stets sofort leeren (in C: `fflush(stdout);`).