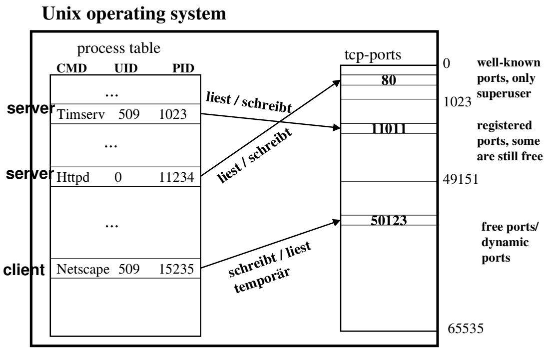
TCP noch mal ... anders!

(keiner verläßt den Raum, ohne
„socket“
rückwärts sagen zu können!)

Skript S. 143

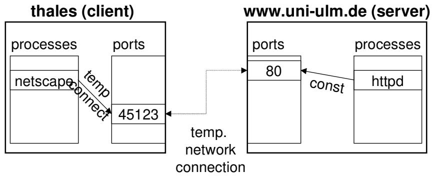


- Wie erfahre ich, **welches** Programm unter Unix aktuell einen bestimmten Port belegt?
  Kommando *lsof* = list of open files (and ports)
  (z.B.: `lsof | grep TCP | grep portno`)

Skript S. 130,138

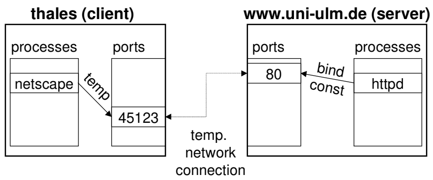
netscape auf thales liest webseite http://www.uni-ulm.de

```
netscape: gethostbyname(„www.uni-ulm.de“) // get IP-Addr. server
fd = socket(AF_INET, SOCKET_STREAM, 0) // TCP-IP socket
sin_port = htons(80);                  // well-known port
sin_family = AF_INET                   // Internetdomain
connect(fd, [to host www.uni-ulm.de; port 80]);
write(fd, „GET / HTTP/1.0\n\n“, 16); read(fd, buf, sizeof(buf));
```



httpd (Webserverprog) auf www.uni-ulm.de erhält Anfrage von Netscape auf thales.mathematik.uni-ulm.de

```
httpd: sin_port = 80;                  // Dienst an Port 80
sin_family = AF_INET                   // Internetdomain
fd = socket(AF_INET, SOCKET_STREAM, 0) // TCP-IP socket
sin_port = htons(80);                  // well-known port
sin_family = AF_INET                   // Internetdomain
bind(fd, [Internetdomain; port 80]);   // socket „anbinden“
listen(fd, SOMAXCON);                  // max. Anfragen Queue
nfd = accept(fd, ....);     // Anfrage von thales via neuem fd annehmen
read(nfd, buf, sizeof(buf)); write(nfd, „<html><body>...“, xyz);
```



Skript S. 135

struct sockaddr

- **zentraler Informationsträger für bind, accept und connect ist die Struktur sockaddr**
  - bei Protokollart AF_INET: struct sockaddr_in
  - bei Protokollart AF_UNIX: struct sockaddr_un
- **wichtigster Inhalt bei (sin_family=)AF_INET:**
  - sin_port = Portnummer
  - sin_addr = IP-Adresse Server bzw. Client (bei accept)
- **wer braucht was?**
  - bind: braucht Infos **für** Server (Protokoll+Port)
  - accept: schreibt Infos **über** Client (Protokoll+IP+Port)
  - connect: braucht Infos **über** Server (Protokoll+IP+Port)

Skript S. 137

Wie erfahre ich, **welcher Rechner** von **welchem Port** aus sich an meinen Server angedockt hat?

accept(fd, (struct sockaddr)&**c_addr**, ...)*

accept schreibt Infos über den Client in die sockaddr-Struktur c_addr

*printf(„port: %d\n“, **ntohs**(c_addr.sin_port));*

ntohs = converts network byte order to host byte order

*printf(„rechner: %s\n“, **inet_ntoa**(c_addr.sin_addr));*

inet_ntoa = konvertiert IP-Adresse in String (doted-decimal form z.B.134.60.66.21)

Skript S. 148

- bisher: *iterativer server* arbeitet alles brav nacheinander ab
- die Anfragen werden in eine Queue gestellt, deren Länge per *listen()* bestimmt wurde
- kommen mehr Anfragen, werden diese abgelehnt
- besseres schnelleres Konzept: *concurrent server* = paralleles abarbeiten

Skript S. 148