Networking in C

Protokolle

TCP	UDP
connection-oriented	connection-less
stream (byte oriented)	datagram (message oriented)
server/client	p2p

Erste Aufgabe

Client, welcher sich zu einem Server verbindet und eine ASCII-kodierte, NUL-terminierte Zeichenkette sendet (maximal 2^{10} Byte). Der Server sendet eine gleich große, NUL-terminierte Zeichenkette zurück, die der Client ausgeben soll.

Name resolution

Umsetzung von host name/service name auf IP-Adresse(n). `getaddrinfo()` liefert eine verlinkte Liste von IP-Adressen und Portnummern in einer `struct addrinfo`.

Server hören auf allen IPs auf eingehende Verbindungen. Clients versuchen jede IP, bis eine Verbindung zu Stande kommt.

`freeaddrinfo()` gibt die verlinkte Liste wieder frei.

getaddrinfo()

Sockets müssen an eine Adresse gebunden werden. Mit den Daten aus `struct addrinfo` kann man einen Socket erzeugen:

```
struct addrinfo *ais;
struct addrinfo hints = {…};
int err = getaddrinfo(host, port, &hints, &ais);
for(struct addrinfo * ai = ais; ai; ai = ai->ai_next) {
  int s = socket(ai->ai_family, ai->ai_socktype,
                 ai->ai_protocol);
  /* do something with socket */
}
freeaddrinfo(ais);
```

Server

Der Server bindet den Socket an eine Adresse und hört auf eingehende Verbindungen:

```
/* do something with socket */
bind(s, ai->ai_addr, ai->ai_addrlen)
listen(sock_fd, SO_MAXCONN);
```

Server run loop

```
while(1) {
    /* blocks until new connection */
    int client_socket = accept(sock_fd, …);
    read/write(client_socket, …);
    close(client_socket);
}
close(sock_fd);
```

Client

Der Client verbindet den Socket mit `connect()`:

```
/* do something with socket */
if (connect(s, ai->ai_addr, ai->ai_addrlen)) {
  /* error, try next struct addrinfo */
  close(s);
  s = -1;
  continue;
} else {
  /* success - keep socket and abort loop */
  break;
}
```

Client run loop

```
while(1) {
    /* handle user i/o */
    read/write(s, …);
    /* handle user i/o */
}
close(s);
```

Recap

- getaddrinfo()
- socket()
- connect()
- write()
- read()
- printf()

```
struct addrinfo hints = {0};
hints.ai_flags = AI_ADDRCONFIG;
hints.ai_family = PF_UNSPEC;
hints.ai_socktype = SOCK_STREAM;
hints.ai_protocol = IPPROTO_TCP;
```

TCP ist byte-orientiert, „Nachrichten“ können also auch teilweise gelesen oder geschrieben werden.