TCP Socket Programming

Christian Grothoff

Berner Fachhochschule

November 9, 2017

Today: TCP Socket Programming

Learning objectives:

- More socket APIs: `connect()`, `listen()`, `accept()`, `recv()`, `send()`
- Practice TCP

Programming objective: implement a *file transfer* application.

Creating a TCP socket

```
#include <sys/types.h>
#include <sys/socket.h>

int socket (int domain, int type, int protocol);
```

Use `AF_INET` or `AF_INET6` for *domain*.
Today, we will discuss the *type* being `SOCK_STREAM`.
We need to set *protocol* to `IPPROTO_TCP` or 0.

Bind *server* socket to a port

```
struct sockaddr_in local;
local.sin_family      = AF_INET;
local.sin_port        = htons (LOCALPORT);
local.sin_addr.s_addr = INADDR_ANY;

bind (lsock,
      (struct sockaddr *) &local,
      sizeof(local));
```

Listen for incoming connections

```
listen (lsock, 5);
```

Connect *client* socket to server

```
struct sockaddr_in local;
local.sin_family      = AF_INET;
local.sin_port        = htons (LOCALPORT);
local.sin_addr.s_addr = INADDR_ANY;

bind (cli,
      (struct sockaddr *) &local,
      sizeof(local));
```

Accept incoming connection

```
struct sockaddr_storage addr;
socklen_t alen = sizeof (addr);
incoming = accept (lsock, &addr, &alen);
```

Sending and receiving data

```
ssize_t ret;

ret = send (sock, data, len, 0);
ret = recv (sock, data, len, 0);
```

TCP file transfer

- Write two programs, `tcp-receive` and `tcp-transmit`
- Server sends file, client receives file
- Pass respective file name as command-line argument
- Support multiple concurrent clients
- Migrate from `select()` to `epoll()`
- Migrate from `send()` to `sendfile()`
- Transmit checksum back to server, log discovered problems by client IP address (hint: `shutdown()`).