SEMINAR 13
TCP/IP SOCKETS

SOCKET IS A FILE DESCRIPTOR

AVAILABLE FOR

READING AND WRITING

CREATING A SOCKET ENDPOINT

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

MOST COMMON DOMAINS

- AF_UNIX - LOCAL SOCKET FOR IPC
- AF_INET - IPV4 INTERNET PROTOCOLS
- AF_INET6 - IPV6 INTERNET PROTOCOLS
- AF_PACKET - LOW-LEVEL INTERFACE

MOST COMMON TYPES

- SOCK_STREAM
- SOCK_DGRAM

(AVAILABILITY DEPENDS ON DOMAIN)

ADDITIONAL FLAGS:

- SOCK_NONBLOCK
- SOCK_CLOEXEC

PAIR OF UNIX SOCKETS

```
int socketpair(
   int domain,   // AF_UNIX or AF_TIPC
   int type,
   int protocol, // must be 0 for AF_UNIX
   int sv[2]
);
```

PART I

CLIENT CONNECTION

CONNECT SOCKET TO ADDRESS

```
int connect(
  int sockfd,
  const struct sockaddr *addr,
  socklen_t addrlen
);
```

IPV4 ADDRESS

```
struct sockaddr_in {
  sa_family_t    sin_family;
  in_port_t      sin_port;
  struct in_addr sin_addr;
};

struct in_addr {
  uint32_t       s_addr;
};
```

WARNING: PORT AND ADDR MUST BE IN NETWORK BYTE ORDER

ORDER CONVERSION

```
uint32_t htonl(uint32_t hostlong);
uint32_t ntohl(uint32_t netlong);
uint16_t htons(uint16_t hostshort);
uint16_t ntohs(uint16_t netshort);
```

AFTER THAT WE CAN WORK WITH SOCK_STREAM SOCKET USING READ AND WRITE

CLOSE CLOSES THE SOCKET

SHUTDOWN PERFORMS TCP GRACEFUL SHUTDOWN

PART II
SERVER

BIND SOCKET TO ADDRESS

```
int bind(
  int sockfd,
  const struct sockaddr *addr,
  socklen_t addrlen
);
```

MASK SOCKET AS PASSIVE AND CREATE QUEUE

```
int listen(
  int sockfd,
  int backlog, // <= SOMAXCONN (128 for Linux)
);
```

ACCEPT NEW CONNECTION

```
int accept(
  int sockfd,
  struct sockaddr *restrict addr, // May be NULL
  socklen_t *restrict addrlen     // May be NULL
);
```