Embedded Operating Systems

Agenda

- Socket
  - INET vs UNIX sockets
  - Socket programming
- Shared Memory
  - Internals
  - Shared memory programming

IPC

Socket

- Socket is defined as communication endpoint.
- Sockets can be used for bi-directional communication.
- Using socket one process can communicate with another process on same machine (UNIX socket) or different machine (INET sockets) in the network.
- Sockets can also be used for communication over bluetooth, CAN, etc.
- terminal> man 7 socket
- Reading: Beginning Linux Programming

UNIX socket

- Bi-directional stream based communication mechanism to communicate between two processes running on the same computer.
- UNIX socket is a special file (s).
- terminal> man 7 unix

INET (Internet) socket

- Bi-directional communication mechanism to communicate between two processes running on the same computer or different computers in a network.

- terminal> man 7 ip
- INET socket do not have special file. It is identified by IP address and port number combination.
  - IP address uniquely identify each machine in the network.
    - IPv4 - 32-bit IP address e.g. 192.168.0.101
    - IPv6 - 128-bit IP address e.g. 2001:0db8:85a3:0000:0000:8a2e:0370:7334
  - PORT number is 16-bit logical number (0-65535) to identify socket uniquely on a system. _ 0-1023 - Used for well-known services like http (80), https (443), ftp (21), ssh (22), telnet (23), etc. _ 1024+ - Used for applications like mysql (3306), mqtt(1883), etc.
- INET sockets can be stream based or packet based depending on the protocol.
  - TCP -- Stream based protocol
    - Acknowledgement
    - Order of data transfer
    - Reliable
  - UDP -- Packet based protocol
    - No acknowledgement
    - No guarantee of data transfer order
    - Packets may be lost
    - Faster

Important networking commands

```
sudo apt install net-tools

netstat -tln
# list all tcp listening sockets

netstat -tn
# list all tcp client sockets

netstat -uln
# list all udp listening sockets

netstat -un
```

```
# list all udp client sockets

netstat -a
# list all sockets

sudo netstat -ap
# list all sockets
# -p : name of process

ifconfig
# get ip-address of system
```

Socket System Calls

- fd = socket(addr_family, sock_type, protocol);
  - arg1: addr_family = AF_UNIX or AF_INET
  - arg2: sock_type = SOCK_STREAM or SOCK_DGRAM
  - arg3: protocol = 0
- Socket information is maintained in struct sockaddr. It will differ for each address family.
  - struct sockaddr
    - sa_family -- address family (AF_UNIX or AF_INET)
  - struct sockaddr_un
    - sun_family -- address family = AF_UNIX
    - sun_path -- UNIX socket file path
  - struct sockaddr_in
    - sin_family -- address family = AF_INET
    - sin_port -- port number in big endian format
    - sin_addr -- struct in_addr -- ip address in big endian format
- ret = bind(fd, &sock_addr, sock_length)
  - arg1: socket fd (created by socket() call)
  - arg2: address of sockaddr_un or sockaddr_in
  - arg3: sizeof(sockaddr_un) or sizeof(sockaddr_in)

  - returns 0 on success.
- ret = connect(fd, &sock_addr, sock_length)
  - arg1: socket fd (created by socket() call)
  - arg2: address of sockaddr_un or sockaddr_in
  - arg3: sizeof(sockaddr_un) or sizeof(sockaddr_in)
  - returns 0 on success.
- cli_fd = accept(fd, &sock_addr, &sock_length)
  - arg1: socket fd (created by socket() call)
  - arg2: (out-param) address of sockaddr_un or sockaddr_in
  - arg3: (in-out param) sizeof(sockaddr_un) or sizeof(sockaddr_in)
  - returns 0 on success.
- listen(fd, pending);
  - arg1: socket fd (created by socket() call)
  - arg2: max pending socket conection requests
- shutdown(fd, SHUT_RDWR);
  - arg1: socket fd (created by socket() call)
  - arg2: which connections to be closed -- SHUT_RDWR.

Socket Programming Flow

- https://youtu.be/3RwFfquNgNc

Shared memory

- OS can create a special memory region which is accessible to multiple processes (willing to communicate with each other). This region is called as "Shared memory".
- This memory space/region is created in user space. Whole communication happends in user space itself. So this is fastest IPC mechanism.
- The allocated shared memory region needs to be mapped to virtual address space of the process.
  - cmd> cat /proc/pid/maps
- This is Sys V IPC mechanism (like Message queue and Semaphore).
  - cmd> ipcs
- Shared memory programming steps

  - Create shared memory region -- shmget().
  - Associate it with current process and get pointer to it -- shmat().
  - Read/Write in memory using the pointer.
  - Release pointer to the shared memory -- shmdt().
  - Destroy shared memory region -- shmctl().

Shared Memory SysCalls

shmget() syscall

- Create the shared memory object.
- shmid = shmget(shm_key, size, flags);
  - arg1: unique key for shared memory object
  - arg2: size of shared memory region (bytes)
  - arg3: to create shared memory -- IPC_CREAT | perm
    - perm --> octal number --> 0600 (rw- --- ---)
  - returns: shm id (index to shared memory table) on success.

shmctl() syscall

- To control shared memory object/region e.g. mark shm object for deletion, get info about shared memory.

Mark for deletion

- ret = shmctl(shmid, IPC_RMID, NULL);
  - arg1: shared memory id to be deleted.
  - arg2: command = IPC_RMID to delete the shared memory object
  - arg3: for deletion third arg is not required.
- This will mark shared memory object for destruction.
- The shared memory object will be deleted when no processes are attached to it (nattach = 0).

Get info about shared memory

- ret = shmctl(shmid, IPC_STAT, &shmid_ds);
  - arg1: shared memory id whose info to be retrieved.
  - arg2: command = IPC_STAT to get the info about shared memory.
  - arg3: out param to collect info about shared memory.

shmat() syscall

- Get the address of the shared memory region i.e. attach shm region to the current process.
  - Internally increments nattach count in shared memory object.
- ptr = stmat(shmid, virt_addr, flags);
  - arg1: shared memory id of region to be attached to the process.
  - arg2: base virtual address in address space of the current process to be mapped to shared memory region.
    - NULL means use any available address.
  - arg3: flags (extra information/behaviour).
    - 0 means default behaviour.
  - returns pointer to the shared memory region on success. -1 is returned on failure.

shmdt() syscall

- Release the shared memory pointer i.e. detach shm region from the current process.
  - Internally decrements nattach count in shared memory object.
  - If nattach count become zero and shared memory region is marked for deletion, delete the shared memory.
- shmdt(ptr);
  - arg1: shared memory pointer to be detached.

Assignments

1. Create message queue to send the fifo name and file name from client to server. Send given file to the client via the given fifo. Client should accept that file and save to disk.
2. The client process send two numbers and operation (+, -, *, or /) to the server process via unix socket. The server process calculate the result and return it via same socket. The client process print the result.

3. The client process send two numbers and operation (+, -, *, or /) to the server process via inet socket. The server process calculate the result and return it via same socket. The client process print the result.
4. Execute the INET server-client chat application on two different computers in a network.