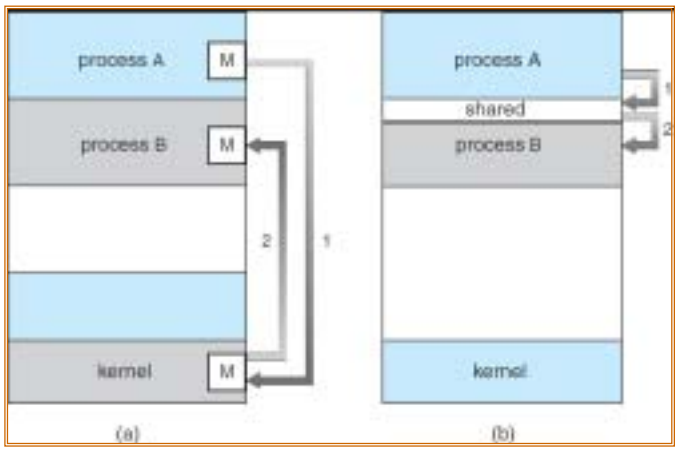
CSc 360
Operating Systems
Inter-Process Communication

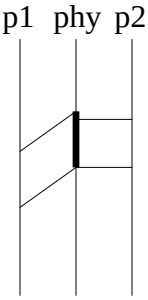
Jianping Pan

Summer 2006

Review: message passing vs shared memory



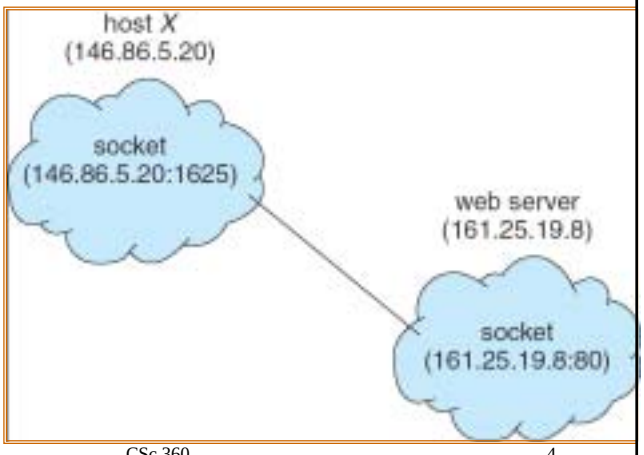
Shared memory



- Create shared memory
  - segment_id = **shmget** (key, size, flag);
  - round up to a multiple of PAGE_SIZE
- Attach shared memory (memory mapping)
  - shared_memory = **shmat** (segment_id, addr, flag);
- Detach shared memory
  - **shmdt** (shared_memory);
- Destroy shared memory
  - **shmctl** (segment_id, IPC_RMID, 0);

Socket

- Socket: an endpoint of communication
- Internet socket
  - IP address
  - port number
  - protocol ID
- Socket pair
  - unique 5-tuple
- Client-server



Socket()

- Create a socket
  - int **socket** (int *domain*, int *type*, int *protocol*);
    - domain
      - PF_INET (Internet protocol family)
      - PF_UNIX (Unix socket) , etc
    - type
      - SOCK_STREAM (supported by TCP)
      - SOCK_DGRAM (supported by UDP) , etc
    - protocol
      - normally implied by socket type

Server

- Listen at a local endpoint
  - int **bind** (int *sockfd*,
    struct sockaddr *my_addr*,
    socklen_t *addrlen*);
  - int **listen** (int *s*, int *backlog*);
    - backlog: maximal # of pending connections
  - int **accept** (int *s*,
    struct sockaddr **addr*,
    socklen_t **addrlen*);
    - return a new and *accepted* socket fd

Client

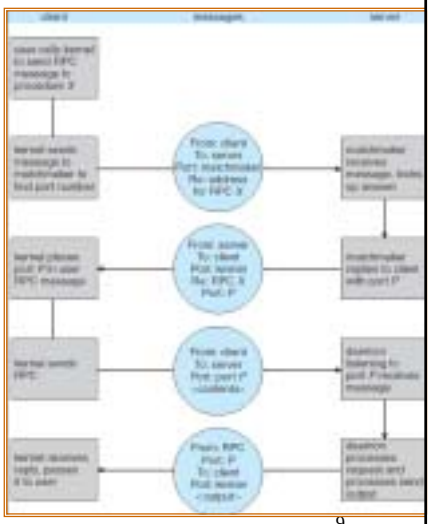
- Connect to a remote endpoint
  - int **connect** (int *sockfd*, const struct sockaddr **serv_addr*, socklen_t *addrlen*);
- Exchange messages
  - int **send**(int *s*,const void ***msg*,size_t *len*, int *flags*);
  - int **recv** (int *s*, void **buf*, size_t *len*, int *flags*);
- Close a socket
  - int **close** (int s);
    - int **shutdown** (int s, int how);

Client-server with socket

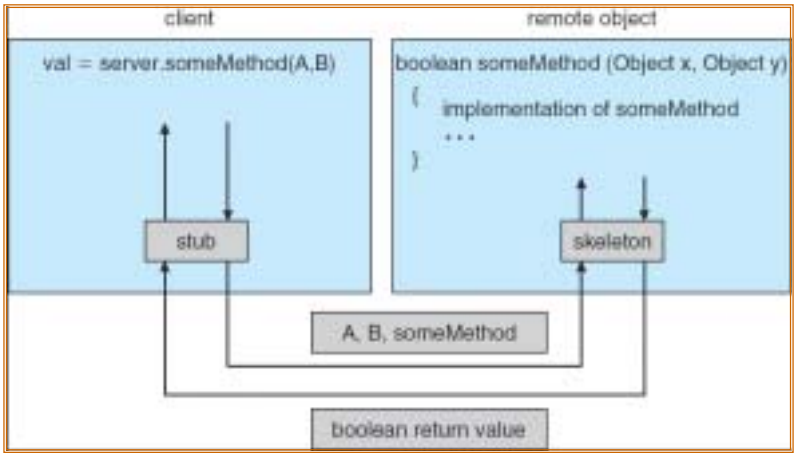
- Client
  - socket()
  - connect()
  - send()
  - recv()
  - close()
- Server
  - socket()
  - bind()
  - listen()
  - accept()
  - recv()
  - send()
  - close()

Remote procedure call

- RPC
  - procedure calls over network
- Client stub
  - locate server
    - matchmaker (e.g., portmap)
  - pass parameters
    - marshall (e.g., XDR)
- Server stub



Marshall parameters



This lecture

- IPC examples
  - POSIX shared memory
  - Internet socket
    - client-server model
  - RPC
- Explore further
  - how to reuse a port?
  - how to do non-blocking send()/recv()?

Next lecture

- Thread
  - read OSC7 Chapter 4 (or OSC6 Chapter 5)