SSC150 – Sistemas Computacionais Distribuídos

Comunicação em Sistemas Distribuídos
Sockets

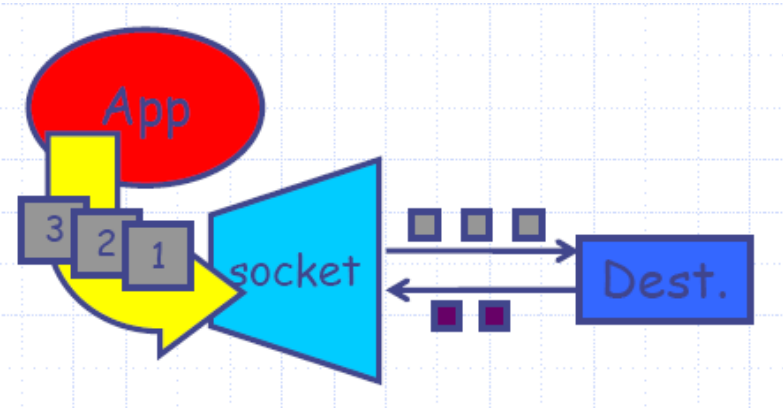
3ª aula
18/03/2010

Profa. Sarita Mazzini Bruschi
sarita@icmc.usp.br

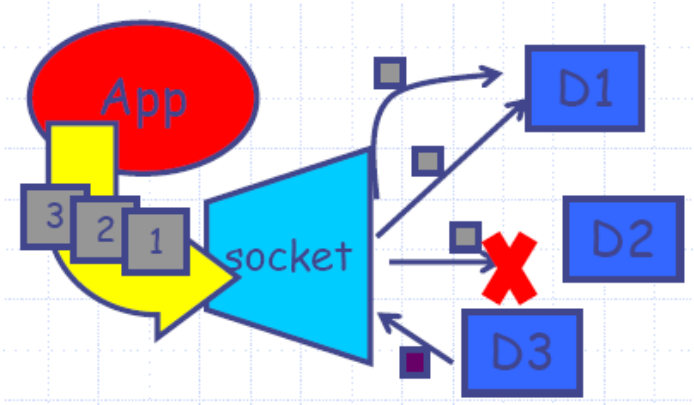
Sockets usando C

Tipos de Sockets

- Stream Sockets
  - Entrega confiável
  - Garantia de ordenação das mensagens
  - Orientado à conexão
  - Bidirecional



- Datagram Sockets
  - Entrega não confiável
  - Sem garantias de ordenação das msgs
  - Não orientado à conexão
  - Pode enviar ou receber



Estruturas e manipulação de dados

- Estrutura sockaddr

```
struct sockaddr {
    u_short    sa_family;      // endereço da família AX_xxx
    char       sa_data[14];    // endereço específico do protocolo
    }
```

- Para a domínio Internet (TCP/IP), utiliza-se a estrutura sockadd_in

```
struct sockadd_in {
    short              sin_family      // AF_INET
    u_short            sin_port        // número da porta
    struct in_addr     sin_addr        // endereço do host
    char               sin_zero[8]     // não usado
}
```

Estruturas e manipulação de dados

- Apesar de existirem várias estruturas (sockaddr_in, sockaddr_ns, sockaddr_un), algumas funções, tais como *connect* e *bind* precisam do endereço da estrutura genética (sockaddr) e do tamanho da estrutura
- É necessário fazer *casting*:

```
struct sockaddr_in serv_addr;
connect(sockfd, (struct sockaddr *) &serv_addr,
    sizeof(serv_addr));
```

Estruturas e manipulação de dados

- Existem dois tipos de ordenação do endereço:
  - Bits mais significantes primeiro
    - Network Byte Ordering
  - Bits menos significantes primeiro
- Quanto uma função precisa que o endereço esteja no formato Network Byte Ordering, precisa-se usar uma função de conversão

Estruturas e manipulação de dados

- Funções para conversão do formato “host” para o formato “network”:
  - htonl (h-to-n-l): host to network; long integer
  - htons (h-to-n-s): host to network; short integer
  - ntohl (n-to-h-l): network to host; long integer
  - ntohs (n-to-h-s): network to host; shot integer

Estruturas e manipulação de dados

- Funções para conversão de endereços:
  - unsigned long inet_addr (char **ptr*)
    - Converte uma string em formato de endereço Internet (decimal com ponto)
  - char *inet_ntoa (struct in_addr *inaddr*)
    - Função inversa

Principais funções

- int socket (int *family*, int *type*, int *protocol*):
  - Cria um descritor
  - *family*: AF_UNIX
    AF_INET
    AF_NS
    AF_IMPLINK
  - *type*: SOCK_STREAM
    SOCK_DGRAM
    SOCK_RAW
    SOCK_SEQPACKET
    SOCK_RDM
  - *protocol*: normalmente 0

Principais funções

- int bind (int *sockfd*, struct sockaddr **my_addr*, int *addrlen*)
  - Associação de um socket a uma porta local
  - Necessário quando for utilizar a função listen() depois
  - sockfd: descritor do socket
  - my_addr: ponteiro para uma struct sockaddr que contém informação sobre seu endereço, nome, porta e endereço IP
  - addrlen: tamanho da estrutura sockaddr

Principais funções

- bind()
  - O processo de pegar o próprio endereço IP e/ou porta pode ser feito automaticamente:
    - Fazendo my_addr.sin_port = 0
      - Escolha da porta
    - Fazendo my_addr.sin_addr.s_addr = INADDR_ANY
      - Preenche automaticamente o endereço IP da máquina que o processo está sendo executado
  - bind() retorna -1 se ocorreu um erro
  - Portas abaixo de 1024 são reservadas
  - Pode-se utilizar qualquer porta até 65535

Principais funções

- connect(int sockfd, struct sockaddr *serv_addr, int addrlen)
  - Conecta com um servidor
  - sockfd: descritor do socket
  - serv_addr: ponteiro para uma struct sockaddr que contém informação sobre endereço, nome, porta e endereço IP do destino
  - addrlen: tamanho da estrutura sockaddr

Sockets em C
Principais funções

- listen(int sockfd, int backlog)
  - Espera por conexões
  - Processo que:
    - Primeiro “escuta”
    - Depois “aceita”
  - sockfd: descritor do socket
  - backlog: número de conexões que são aceitas na fila de entrada

Principais funções

- accept(int sockfd, void *addr, int *addrlen)
  - Algum processo irá tentar se conectar a uma porta que o servidor está “ouvindo”
  - A conexão será colocada na fila de entrada esperando que seja “aceita”
  - Quando se faz um accept() está aceitando uma conexão pendente
  - sockfd: descritor do socket
  - addr: ponteiro para uma struct sockaddr_in que contém informação sobre a conexão que está chegando
  - addrlen: tamanho da estrutura sockaddr

Principais funções

- send(int sockfd, const void *msg, int len, int flags)
- receive(int sockfd, void *buf, int len, unsigned int flags)
  - Para envio e recebimento
  - sockfd: descritor do socket
  - msg: ponteiro para a mensagem que se quer enviar
  - buf: buffer onde a mensagem será colocada
  - len: tamanho da mensagem em bytes
  - flags = 0

Principais funções

- sento(int sockfd, const void *msg, int len, unsigned int flags, const struct sockaddr *to, int tolen)
- int recvfrom(int sockfd, void *buf, int len, unsigned int flags, structlen, unsigned int flags, struct sockaddr *from, int *fromlen);

Principais funções

- close(sockfd)
  - Fecha uma conexão

Principais funções

- int gethostname(char *hostname, size_t size);
  - Retorna o nome do computador que o processo está sendo executado
- struct hostent *gethostbyname(const char *name);
  - Retorna um ponteiro para uma estrutura hostent
  - struct hostent {

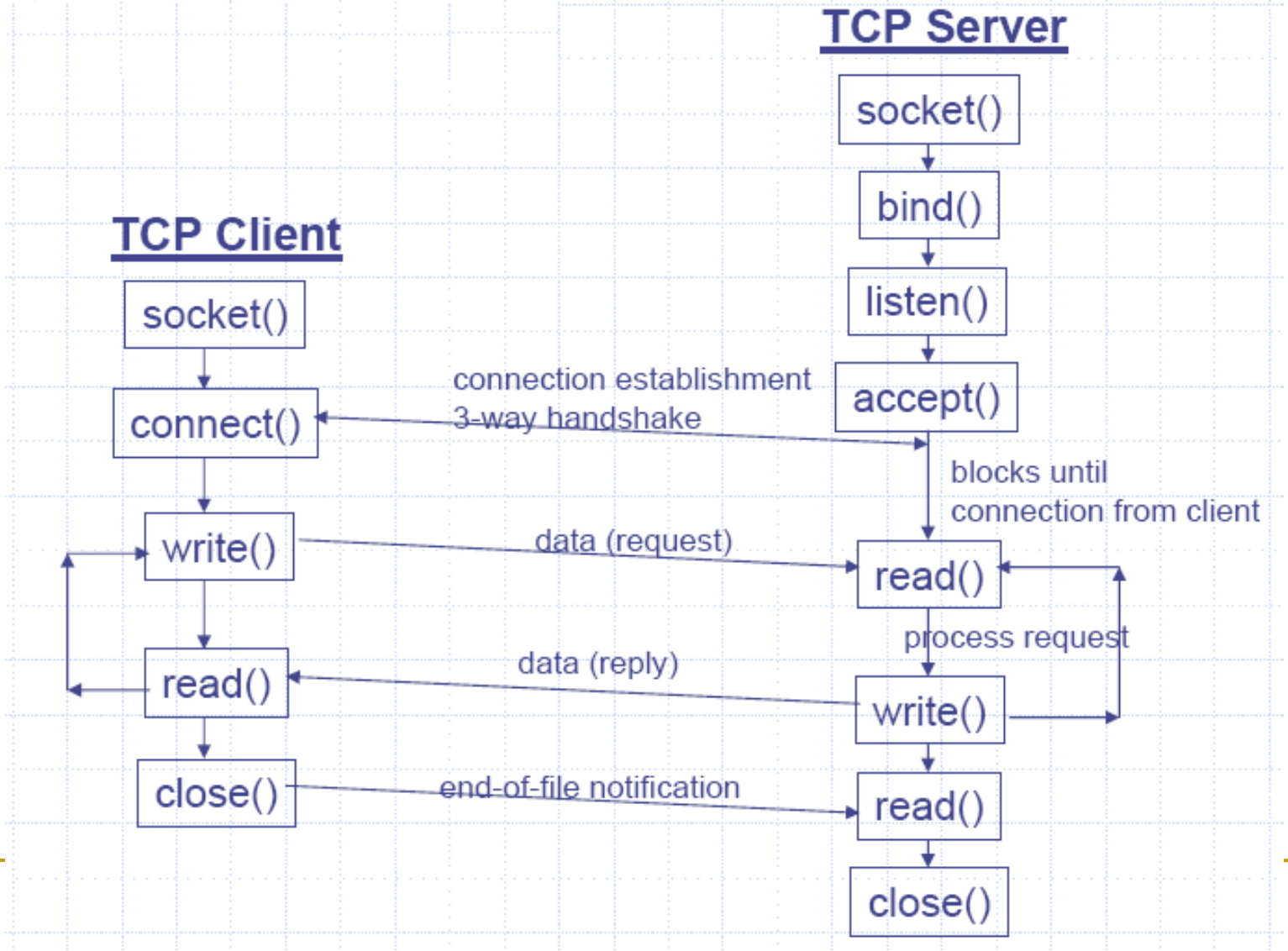
```
char *h_name;
char **h_aliases;
int h_addrtype;
int h_length;
char **h_addr_list;
};
```

#define h_addr h_addr_list[0]

Principais funções

- h_name – nome do host
- h_aliases – lista alternativa de nomes.
- h_addrtype – tipo do endereço utilizado; normalmente AF_INET.
- h_length – tamanho do endereço em bytes.
- h_addr_list – Array de endereços de rede para o host. Os hosts são endereçados em Network Byte Order.
- h_addr – Primeiro endereço de h_addr_list.

Cliente/Servidor TCP



Cliente/Servidor UDP