a Raju Alluri presentation

Unix Internals

Module 08

Raju Alluri
askraju @ spurthi.com

Unix Internals, Module 08

- Networking

Network Interfaces

- Network Interfaces
  - An interface for each network connection
    - Has an IP address associated with it
  - There is a default interface called “loopback” interface
    - Using loopback interface, Network programs can execute (using the same host) even when the host is not connected to a “network”
    - 127.0.0.1

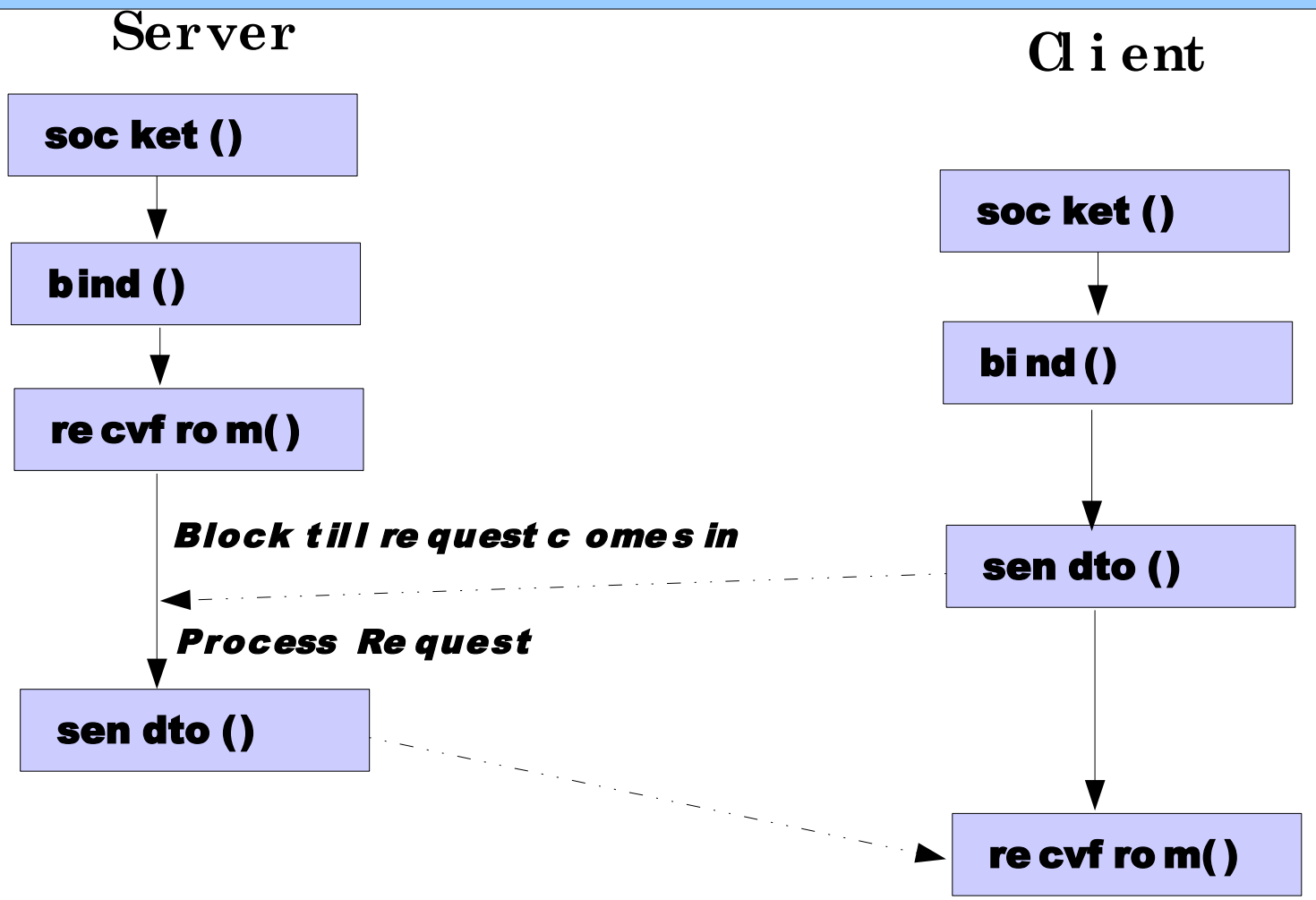
Client Server Programming

- Client
  - The program that uses the service
  - Can be relatively shortlived (not all the time)
- Server
  - The program that serves the requests
    - Serves each incoming request from each client
  - Typically runs as a daemon process

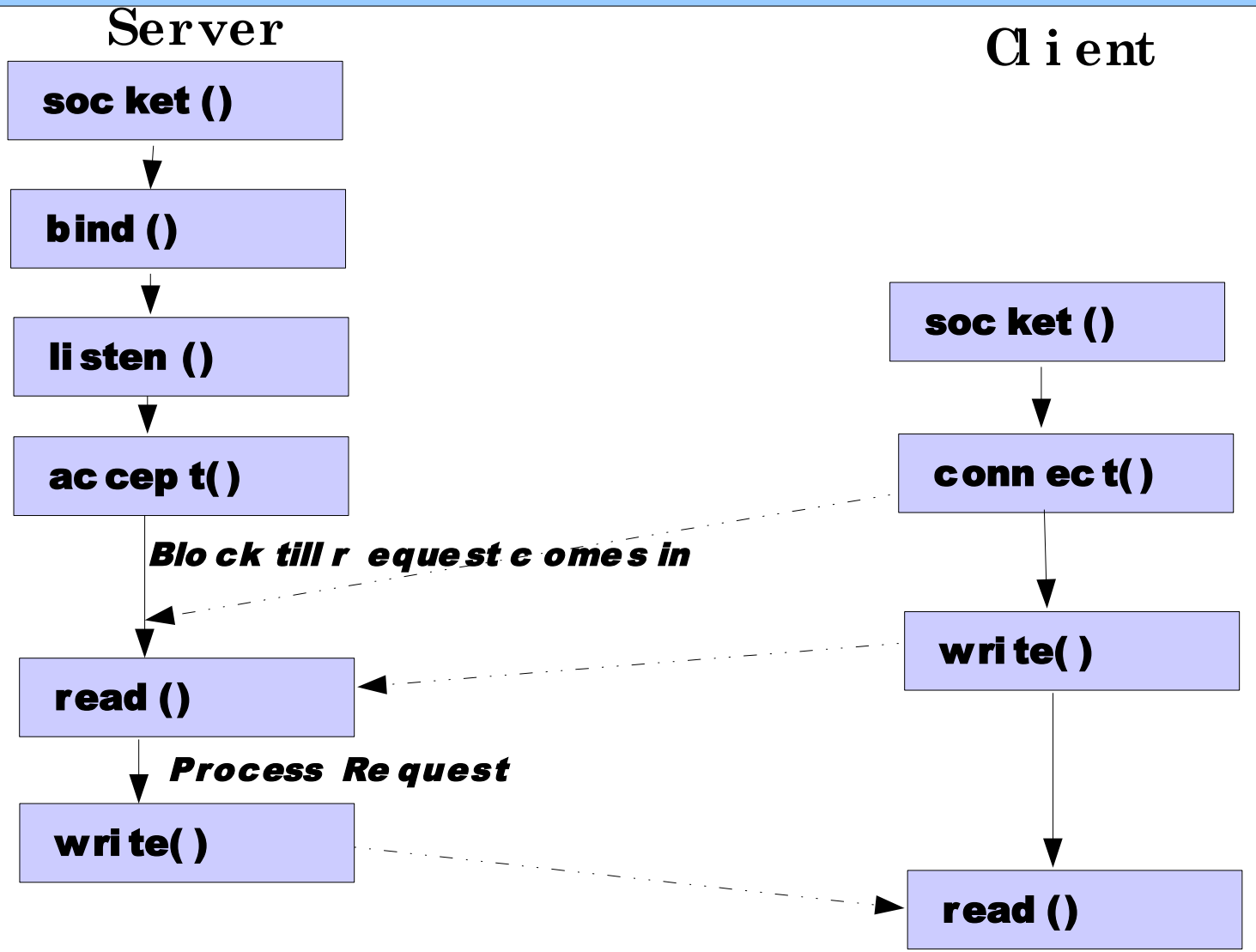
Service Types

- Connection-less services (UDP)
  - Doesn't establish a two-way communication for the entire life of request/response
  - Light weight (network resources)
  - Not very reliable
- Connection-oriented services (TCP)
  - Establishes two-way communication for the life of request/response
  - Network resource intensive
  - Relatively reliable

Connectionless Service



Connection-Oriented Service



socket

- Socket system call
  - Gets an unnamed socket

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

    - Family
      - AF_UNIX, AF_INET, AF_NS, AF_IMPLINK
    - Type
      - SOCK_DGRAM, SOCK_STREAM, SOCK_RAW, ...
    - Protocol
      - Typically 0
      - IPPROTO_UDP, IPPROTO_TCP, IPPROTO_ICMP, IPPROTO_RAW, NSPROTO_SPP

bind

- bind system call
  - Assigns name to an unnamed socket

```
int bind(int sockfd, struct sockaddr
  *myaddr, int addrlen);
```

    - Myaddr
      - Pointer to protocol specific address
  - Servers to register on a well known address
  - Client to register on a specific address for itself
  - For connectionless clients to assure unique return address

connect

- connect() system call
  - Connects to a socket descriptor

```
int connect(int sockfd, struct sockaddr
  *servaddr, int addrlen);
```

    - servaddr
      - Pointer to protocol specific address of the server

listen

- listen() system call
  - Connection-oriented server's willingness to accept connections

```
int listen(int sockfd, int backlog);
```

    - backlog
      - Number of connection requests that can be queued by the system while the server is processing a request.

accept

- accept() system call
  - Returns a socket with the first connection request in the queue. The returned socket has same properties as s. Blocks until a connection request is available

```
int accept(int s, struct sockaddr *addr,
  socklen_t *addrlen);
```

    - addr and addrlen
      - Give the address info of the peer (client)

send*/recv*

```
int send(int sockfd, char *buf, int nbytes,
 int flags);
int sendto(int sockfd, char *buf, int
 nbytes, int flags, struct sockaddr *to, int
 addrlen);
int sendmsg(int sockfd, struct msghdr *msg,
 int flags
int recv(int sockfd, char *buf, int nbytes,
 int flags);
int recvfrom(int sockfd, char *buf, int
 nbytes, int flags, struct sockaddr *from,
 int addrlen);
```

- Flags
  - MSG_OOB, MSG_PEEK, MSG_DONTROUTE

readv/writev

```
ssize_t readv(int fd, struct iovec *vector,
 int count);
ssize_t writev(int fd, struct iovec *vector,
 int count);
   struct iovec {
      void *iov_base;
      size_t iov_len;
   };
```

- vector: array of records of type struct iovec
- count: number of records in vector