
 <h2 style="text-align: center;">A Course on Internetworking & Network-based Applications</h2>	
---	--

<p>CS 6/75995 Internet-based</p> <p>Applications & Systems Design</p>	 <p>Kent State University Dept. of Computer Science <u>LECT-2</u></p>
--	---

Application Architecture


3

Today's Topic

How to Build a Network Application

We will build a baby Server/Client today!

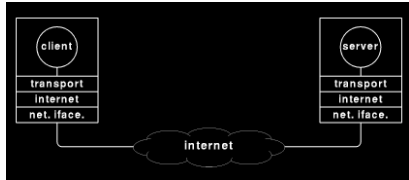





INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-4
IA003F_jpvred@kent.edu
Javed I. Khan@2004

Client Server Interaction

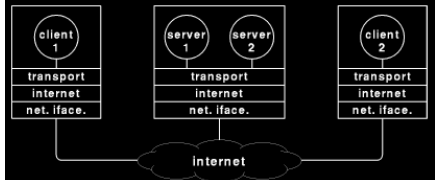





INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-5
IA003F_jpvred@kent.edu
Javed I. Khan@2004

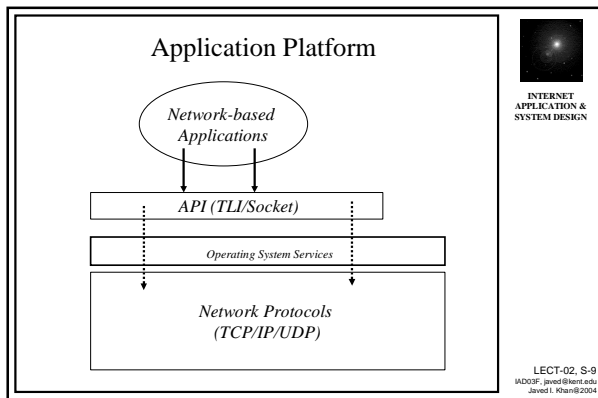
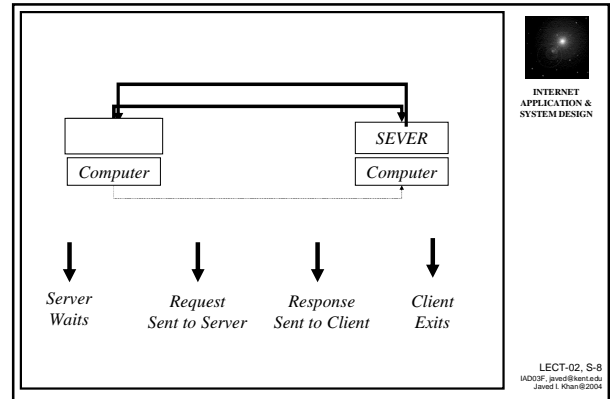
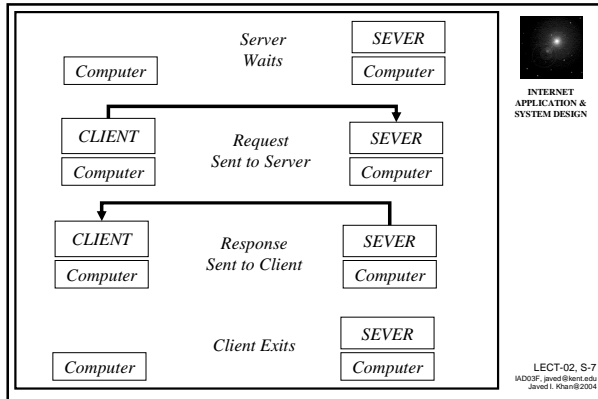
Multiple Services on One Computer





INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-6
IA003F_jpvred@kent.edu
Javed I. Khan@2004



API to Network

10

- ## Network APIs
- **Transport Layer Interface (TLL)** was developed in mid 1980s by AT&T with release 3 of system V UNIX. Later it became a part of Sun systems. Now this is available everywhere.
 - **Socket API** is the original API developed by Berkeley UNIX group in late 70's and early 80s. Available on BSD UNIX Systems.
 - **WinSock** is the API version provided for Microsoft Windows.
- LECT-02, S-11
 IAD03F_j.pvred@kent.edu
 j.pvred1_Khan@2004

- ## What is Socket API?
- Socket Application Program Interface can be used to access a network protocol stack from any programming language.
 - Socket API has been inspired by Unix *open-read-write-close* file access paradigm (original is Multics).
 - However, accessing network is substantially more complex than file access.
- LECT-02, S-12
 IAD03F_j.pvred@kent.edu
 Javed I. Khan@2004

Sockets & Files

- An integer called file descriptor is returned, when a file is opened.
- In the same way a socket descriptor is returned when a socket is opened.
- A file descriptors binds to a file when it is open.
- But, a socket can be created without binding to a specific destination. Applications can choose when to bind
 - Datagram binds each time when it sends, therefore same socket can be used to send to many.
 - TCP binds once, and it remains, thus avoids repeated binding.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-13
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Socket Creation & Closing

- **descriptor=socket(pfamily, type, protocol)**
 - pfamily =PF_INET|PF_APPLETALK | PF_UNIX| PF_PUP
 - type=SOCK_STREAM, SOCK_DGRAM, SOCK_RAW, etc.
 - protocol=subtype of protocol family if any.
- Unix uses fork() and exec() to create and spawn new program. Child always inherits all parent sockets. UNIX maintains a count of owners.
- A process can close a socket by **close(socket)**



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-14
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Specifying a Local Address

- Initially a socket is created without any association to a local or remote address.
- For TCP/IP it means no protocol port number.
- Some application may not care (clients generally). Some do (all servers). The call:
- **bind(socket, localaddress, addrlen)**
 - socket is the socket descriptor returned by socket()
 - localaddress is a complex structure with several fields, and may vary for protocols.
 - For TCP/IP it contains both the port number and the IP address of the host is in it.
 - addrlen is the length of the address.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-15
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Address Structures

- Berkeley code defines a generic sockaddr structure to represent address of a connection end-point.

```
struct sockaddr {
    u_char sa_len;           /*total length of the address*/
    u_char sa_family;       /*family of address*/
    char sa_data[14];       /*the address itself*/
};
```
- Example: TCP/IP defines its own exact sockaddr_in:

```
struct sockaddr_in {
    u_char sin_len;         /*same as sa_len*/
    u_char sin_family;     /*same as sa_family*/
    u_short sin_port;      /*protocol port number*/
    struct in_addr sin_addr /*6B IP address of the host*/
    char sin_zero[8];      /*not used 6+8=14*/
};
```
- Generally server calls bind to specify a server port number at which it will accept connection.
- A server on a multi-homed host can write down INADDR_ANY instead of the IP address to say it will accept the connection in any of the computers IP addresses.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-16
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Listening for Connection

- listen allows a server to wait for a connection request from a remote client.
- Listen(socket, queue length)
 - socket is the descriptor that has been created and is bound to a local address.
 - queue length specifies how many request can wait while server is busy with one.
 - OS maintains a separate request queue for each socket. If the queue is full, OS refuses new requests.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-17
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Waiting for Accepting a Connection

- After a server executes *socket-bind-listen*, it can go to sleep by calling *accept*. The Operating System will wake up the server when there is a request in the request queue.
- **newsock=accept(socket, &addr, &addrlen)**
 - socket is on which it was waiting.
 - addr is a pointer, in which the address of the client will be returned by the OS. Addrlen is length of this address.
 - newsock is a new socket created by system which has its destination pre-connected to the client.
- The server can keep on communicating with the requesting client with the newsocket, and close it when done. Meanwhile, the original socket remains intact to accept request from other clients.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-18
IA003F_jpv01@kent.edu
Javed I. Khan@2004

Connecting to a Destination Address

- Initially a socket is created also without any destination address. An client application program must call connect to establish connection.
- connect(socket, destaddr, destaddrlen)**



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-19
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

Sending Data

- write(socket, buffer, bytelength)**
- writev(socket, iovec, vectorlen)**
 - iovec is an array of addresses and their lengths. The system gathers all the data.
- send(socket, message, length, flags)**
- sendto(socket, message, length, flags, daddr, daddrlen)**
- sendmsg(socket, messagestruct, flags)**



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-20
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

Receiving Data

- read(descriptor, buffer, length)**
- readv(descriptor, iovec, vectorlen)**
- recv(socket, buffer, length, flags)**
- recvfrom(socket, buffer, length, flags, fromaddr, addrlen)**
- recvmsg(socket, messagestruct, flags)**



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-21
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

Obtaining Local & Peer Address

- getpeername(socket, &destaddr, &addlen)**
- getsockname(socket, &localaddr, &addlen)**

Obtaining & Setting Socket Options

- Getsockopt(socket, level, optionid, &optionval, &length)**
- Setsockopt(socket, level, optionid, optionval, length)**
- Example options are timeout parameters, allocated buffer space etc.



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-22
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

Handling Multiple Request

- nready=select(ndesc, indesc, outdesc, excdesc, timeout)**
 - a call to select will allow server to wait till one of the descriptor is ready.
 - ndesc=number of descriptors. System checks descriptors from 0 to ndesc-1. There are 3 bit masks to wait on a selected subset.
 - Timeout says how long to wait. 0 means wait indefinitely.

Obtaining & Setting Hostnames

- gethostname(name, length)**
- sethostname(name, length)**



INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-23
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

Obtaining & Setting Domain Names

- setdomainname(name, length)**
- getdomainname(name, length)**

Network Byte Order

- Localshort=ntohs(netshort)**
- Locallong=ntohl(netlong)**
- Netshort=htons(localshort)**
- Netlong=htonl(locallong)**




INTERNET
APPLICATION &
SYSTEM DESIGN

LECT-02, S-24
IA003F_j.pvred@kent.edu
Javed I. Khan@2004

IP Address Manipulation


String to 32 bit network byteordered address

- **Address=inet_addr(string)**
- **Address=inet_network(string)**
 - address is a 32 bit IP address in network byte order.
 - string is an ASCII string with IP in dotted decimal notation
 - inet_network returns 0 for host part.
- **Str=inet_ntoa(internetaddr)**
- **Internetaddr=inet_makeaddr(net,local)**
- **Net=inet_netof(internetaddr)**
- **Local=inet_lnaof(internetaddr)**


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-25
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004

Accessing Domain Name System

- Each computer now also have a symbolic domain name.
 - Such as www.kent.edu or shimana.facnet.mcs.kent.edu
- A set of designated computers (known as DNS servers) scattered across the internet maintains the mapping of DNS to the actual IP.
- Translation from domain name to IP address or the opposite, requires communication with these servers.


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-26
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004

Accessing Domain Name System

- **res_init()**
- **res_mkquery(op,dname,class,type, data, datalen,newrr, buffer, buflen)**
- **res_send(buffer, buflen, answer, anslen)**
- **dn_expand(msg,eom,compressed,full,fullen)**
- **dn_comp(full,compressed,cmprlen,prevptr,lastptr)**
- **ptr=gethostbyname(namestr)**
- **ptr=gethostbyaddr(addr,len,type)**


Initialize DNS comm.

Form Query.

Send Query.


Conversion between ASCII name and compressed domain name format.

Takes a domain name and returns a structure with information about the domain.


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-27
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004

Obtaining Information about Network Services


- WHOIS a special server service, which allows a client in one machine to obtain information about user who has account in server machine. It runs on Port 43.
 - **Ptr=getservebyname(name,proto)**
- Name is the address of a desired service, and proto is usually TCP or UDP. It returns a structure which contains name of the service, a list of aliases, protocol identifier for this service, and an integer protocol port number.
 - **Ptr=getservbyport(port,proto)**


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-28
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004

Obtaining information about Network & Protocol


- **Ptr=getnetbyname(namestr)**
- **Ptr=getnetbyaddr(netaddr, addrtype)**
- Each protocol has official name, number and registered aliases. These routines can be used to obtain complete information from name or port number of it.
- **Ptr=getprotobyname(name)**
- **Ptr=getprotobynumber(number)**

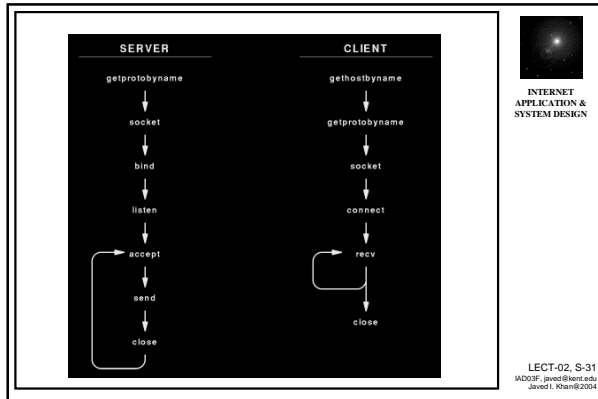
Namestr is the name of the network in ASCII, ptr is a data structure which contain 32 bit IP address and other information about the net.


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-29
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004

An Example Service

- A client connect to a server, and waits for output.
- Server returns the count of the times it has been contacted by any client.
- Upon receiving the data the client prints it to screen.
- Command line arguments:
 - client <hostname> <portname>
 - server <portnumber>
 - hostname and portnumbers are optional.
 - Default host is localhost
 - Default port is 5193.
- Output on client machine:
 - This server has been contacted 10 times.


 INTERNET APPLICATION & SYSTEM DESIGN
 LECT-02, S-30
 W003F_j.pvred@kent.edu
 Javed I. Khan@2004



server.c

[Click Here](#)

<http://www.animasters.com/menu/vrml/estuary/estuaryintro.html>

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-32
 IAD03F_jpvred@kent.edu
 Javed I. Khan@2004

client.c

[Click Here](#)

<http://www.animasters.com/menu/vrml/estuary/estuaryintro.html>

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-33
 IAD03F_jpvred@kent.edu
 Javed I. Khan@2004

Reference

- Internetworking with TCP/IP Volume –III, Comer & Stevens.

INTERNET APPLICATION & SYSTEM DESIGN

LECT-02, S-34
 IAD03F_jpvred@kent.edu
 Javed I. Khan@2004