# CCN: Network Applications

Dr. E.SURESH BABU
Assistant Professor
Computer Science and Engineering Department
National Institute of Technology, Warangal.
Warangal, TS, India.

## Outline

- ❖ Conceptual And Implementation Aspects Of Network Applications
- ❖ Application-layer Concepts
  - ✓ Network services required by applications,
  - ✓ Clients And Servers Concepts
  - ✓ Transport-layer Interfaces.
- ❖ Background Information- Network Programming
- ❖ Brief Look At Network Programming Using Sockets.
- ❖ Applying Sockets using Python,
- ❖ Building the Python's modules for Networked Applications.

## User Oriented Level: Application Layer

## Network Application

- ❖ The **network applications** have been the **driving force** behind the **Internet's success** motivating people in
  - ✓ **Homes, Schools, Governments,** and **Businesses** to make the **Internet an integral part** of their daily activities.

The **Internet** does not provide **services.** Instead, the **Internet** only provides **communication**, and **application programs** provide all **services.**

## Network Applications

- ❖ **Internet Applications** include the
  - ✓ **Classic Text-based Applications(1970s -1980s): Text Email, Remote Access To Computers, File Transfers**
  - ✓ **Killer application (mid-1990s): The World Wide Web, Web Surfing, Search, and Electronic Commerce.**

## Network Applications

- ❖ Internet Applications include the
  - ✓ **Voice And Video Applications(2000) :**
    - ➢ **Voice-over-IP (VoIP) and video conferencing over IP** such as **Skype;**
    - ➢ **User-generated video distribution** such as **YouTube;**
    - ➢ **Movies on demand** such as **Netflix.**

## Network Applications

❖ **The emergence of a new generation (2010)**

✓ **Social Networking Applications**, such as **Facebook and Twitter**,

## Creating a Network Applications

## Application Layer:

❖ The application layer is responsible for **creating** set of applications running on the **End Hosts**.

✓ User **write Applications Programs** that run on (different) end systems

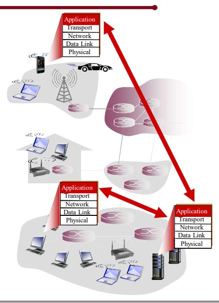
✓ Communicate over network

✓ E.g., **Web Server software** communicates with **Browser Software**

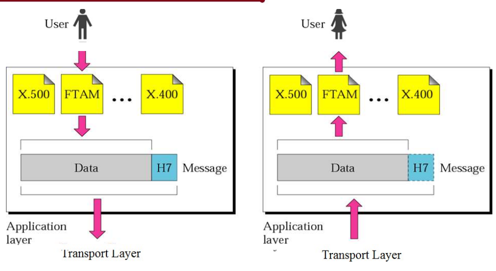
## Application Layer:

- The application layer is responsible for providing services to the user.
- No need to write software for network-core devices
  - ✓ Network-core devices do not run user applications

## Application Layer:



## Application Layer:



## Application Layer:

- ❖ The **Application Layer** is a particularly good place to start **our study of protocols.**
- ❖ Some of the better-known **Application layer protocols** are
  - ✦ **DNS (Domain Name System)** for resolving Internet domain names.
  - ✦ **HTTP (Hypertext Transfer Protocol)** which is Web's application layer protocol
  - ✦ **FTP (File Transfer Protocol)** for file transfers.
  - ✦ **SMTP (Simple Mail Transfer Protocol)** for e-mail.
  - ✦ **NFS (Network File System)** for file sharing in UNIX networks.
  - ✦ **Telnet** for terminal emulation.

## Application Architectures

## Application Architecture

- ❖ The **Network Architecture** is fixed and provides a **specific set of services to applications**. (e.g., **the five-layer Internet architecture** )
- ❖ The **application architecture** are is designed by the **application developer** and dictates
  - ✓ how the **application is structured** over the various end systems.

## Application Architecture

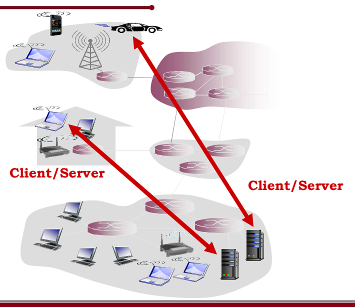
- In **Application Architecture**, There are **Two predominant architectural paradigms** used in modern network applications:
  1. **The Client-Server Architecture**
  2. **The Peer-to-Peer (P2P) Architecture (Self Study)**

## The Client-Server Architecture

## Client/Server Model

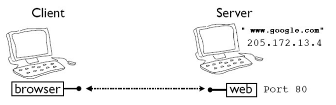
- **Client-Server Model** used by applications to **establish the communication**
- **One application acts as a SERVER**
  - ✓ Starts execution first
  - ✓ Awaits contact from the client
- **The other application becomes a CLIENT**
  - ✓ Starts after server is running
  - ✓ Initiates contact

## Client/Server Model



## Client/Server Model

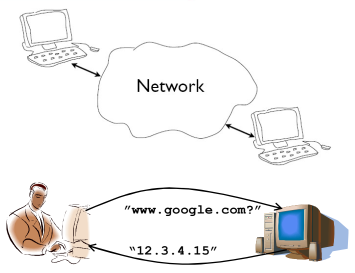
- ❖ **Client-Server** means doing different things with different people
  - ✓ **Servers** wait for incoming connections and **provide a service** (e.g., web, mail, etc.)
  - ✓ Clients make connections to servers



## Client/Server Model

❖ Important Concept: **Once communication** has been established, **data (e.g., requests and responses) can flow** in either direction between a **client and server**

## Logical Communication : Client/Server Model



## Client/Server Concepts : Example

- Suppose Client sends a request message (e.g., HTTP)

```
GET /index.html HTTP/1.0
```

- Server sends back a response message

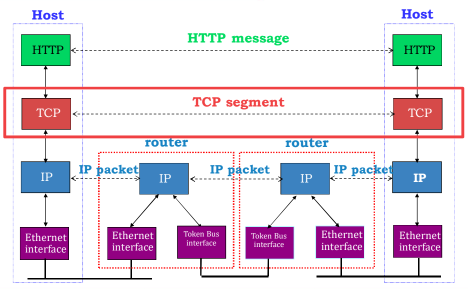
```
HTTP/1.0 200 OK
Content-type: text/html
Content-length: 48823
<HTML>
...
```

- The **exact format** depends on the application

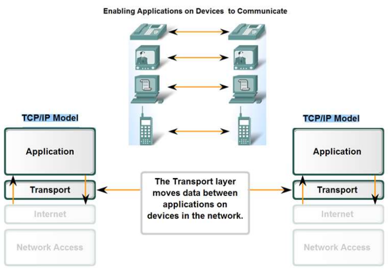
## A Summary Of The Client-server Model.

| Server Application | Client Application |
|---|---|
| Starts first | Starts second |
| Does not need to know which client will contact it | Must know which server to contact |
| Waits passively and arbitrarily long for contact from a client | Initiates a contact whenever communication is needed |
| Communicates with a client by both sending and receiving data | Communicates with a server by sending and receiving data |
| Stays running after servicing one client, and waits for another | May terminate after interacting with a server |

## Recall the Internet Layering Model

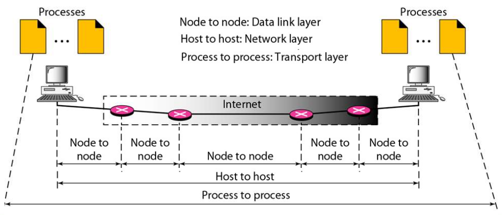


## Application-to-Transport Interface


Enabling Applications on Devices to Communicate
TCP/IP Model
TCP/IP Model
Application
Application
Transport
Transport
Internet
Internet
Network Access
Network Access
The Transport layer moves data between applications on devices in the network.

## Process-to-Process Communication

- The transport layer is **responsible for process-to-process delivery or Application-to-Application Delivery.**



## Two Basic Internet Communication Paradigms

❖ The Internet supports **two basic communication paradigms**

1. **A Stream Paradigm (TCP)**
2. **A Message Paradigm (UDP).**

## The Two Paradigms That Internet Applications Use.

| Stream Paradigm | Message Paradigm |
|---|---|
| Connection-oriented | Connectionless |
| 1-to-1 communication | Many-to-many communication |
| Sequence of individual bytes | Sequence of individual messages |
| Arbitrary length transfer | Each message limited to 64 Kbytes |
| Used by most applications | Used for multimedia applications |
| Built on TCP protocol | Built on UDP protocol |

## Addressing: Port Numbers

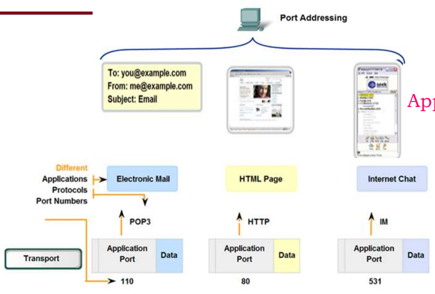
## Addressing: Port Numbers

- ❖ One of the specific responsibilities of transport layer protocol is to **create a process-to-process communication;**
- ❖ To accomplish the process-to-process communication
  - ✓ Transport layer protocol uses **port numbers.**
- ❖ **Port numbers** provide end-to-end addresses at the transport layer

## Addressing: Port Numbers

- ❖ Port Numbers play a important role in the **TCP and UDP protocols.**

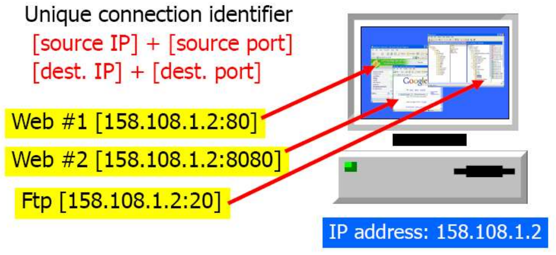
## Addressing: Port Numbers



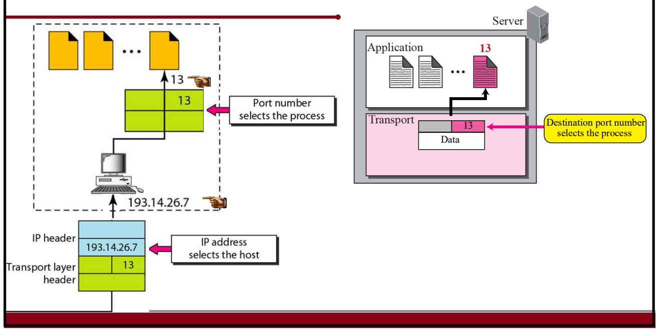
Applications

## Port Address Example

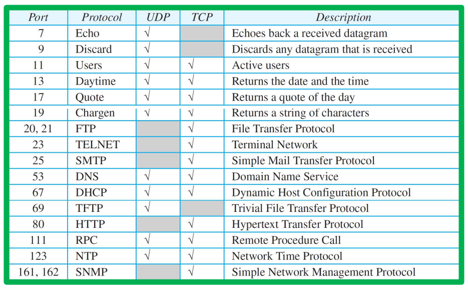
Unique connection identifier
[source IP] + [source port]
[dest. IP] + [dest. port]



## IP addresses Versus Port Numbers


13
13
Port number selects the process
193.14.26.7
IP header
193.14.26.7
Transport layer header
13
IP address selects the host
Server
Application
13
Transport
13
Data
Destination port number selects the process

## Some Well-Known Ports

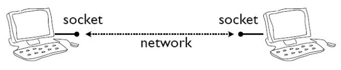


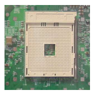
| Port | Protocol | UDP | TCP | Description |
|---|---|---|---|---|
| 7 | Echo | √ | | Echoes back a received datagram |
| 9 | Discard | √ | | Discards any datagram that is received |
| 11 | Users | √ | √ | Active users |
| 13 | Daytime | √ | √ | Returns the date and the time |
| 17 | Quote | √ | √ | Returns a quote of the day |
| 19 | Chargen | √ | √ | Returns a string of characters |
| 20, 21 | FTP | | √ | File Transfer Protocol |
| 23 | TELNET | | √ | Terminal Network |
| 25 | SMTP | | √ | Simple Mail Transfer Protocol |
| 53 | DNS | √ | √ | Domain Name Service |
| 67 | DHCP | √ | √ | Dynamic Host Configuration Protocol |
| 69 | TFTP | √ | | Trivial File Transfer Protocol |
| 80 | HTTP | | √ | Hypertext Transfer Protocol |
| 111 | RPC | √ | √ | Remote Procedure Call |
| 123 | NTP | √ | √ | Network Time Protocol |
| 161, 162 | SNMP | | √ | Simple Network Management Protocol |

## Addressing: Socket Address

## What is Socket?

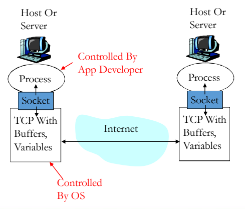
- ❖ **Sockets** are **computer networking data structures** which embody the concept of the **"Communication Endpoint“**
- ❖ Each **Endpoint of a network connection** is always represented by a **host and port #**



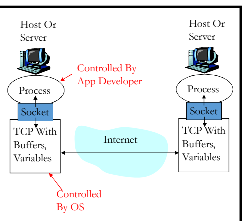


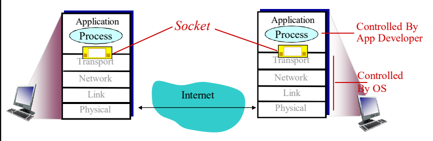
## Purpose…

- ❖ Socket API is the programming interface you need to use for **transmitting and receiving messages**
- ❖ Socket API help you
  - ✓ To send and receive the messages,
  - ✓ Act as a buffer,
  - ✓ Provides an interface between user and lower layers.

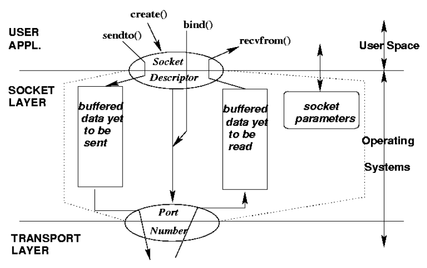

Host Or Server
Host Or Server
Controlled By App Developer
Process
Process
Socket
Socket
TCP With Buffers, Variables
TCP With Buffers, Variables
Internet
Controlled By OS

## Socket Representation

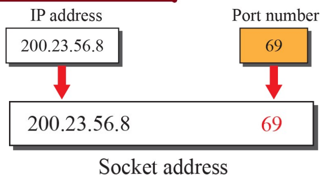




## Socket: Conceptual View



## Port Address : Socket Address (Host-Port Pair)



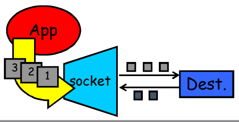
❖ **In Python**, you write it out as a **tuple (host,port)**

**("www.python.org",80)**

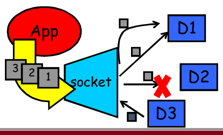
**("205.172.13.4",443)**

## Two Essential Types of Sockets

- SOCK_STREAM
  - ✓ TCP
  - ✓ Reliable Delivery
  - ✓ In-order Guaranteed
  - ✓ Connection-oriented
  - ✓ Bidirectional



- SOCK_DGRAM
  - ✓ UDP
  - ✓ Unreliable Delivery
  - ✓ No Order Guarantees
  - ✓ Connection Less



## Outline

- Conceptual And Implementation Aspects Of Network Applications
- Application-layer Concepts
  - ✓ Network services required by applications,
  - ✓ Clients And Servers Concepts
  - ✓ Transport-layer Interfaces.
- Background Information- Network Programming
- Brief Look At Network Programming Using Sockets.
- Applying Sockets using Python,
- Building the Python's modules for Networked Applications.

## Thank You