CSE2029: Data Communication & Computer Networks

Lecture-4: Application Layer

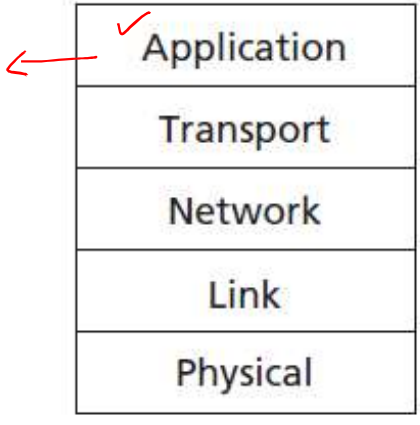
Faculty: Dr. Sandeep Kumar

Outline

- Application Layer
- Network Application Architectures
  - Client-Server Architecture
  - P2P Architecture
- Socket/Application Programming Interface
- Addressing of the Processes
- Functions of Application Layer Protocols

Application Layer

- The application layer is where network applications and their application-layer protocols reside.
- The Internet's application layer includes many protocols, such as the HTTP, SMTP, FTP and DNS etc.
- The packet of information at the application layer as a **message**.



Five-layer Internet protocol stack

Network Application Architectures

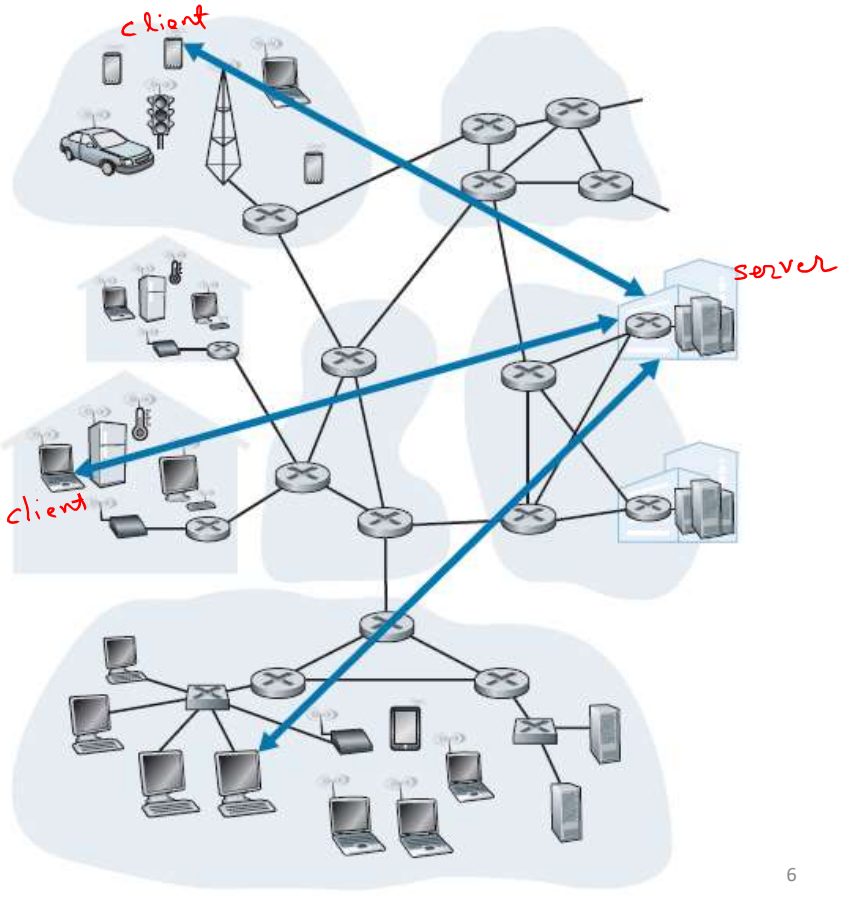
- Before starting the application development (software coding), we should have a broad architectural plan for our application.
- Application's architecture is very different from the network architecture.
- The network architecture is a five-layer Internet architecture to provide specific set of services whereas the application architecture, on the other hand, is designed by the application developer and it decides how the application is structured over the various end systems.
- **Two types of application architectures** are: (1) **client-server architecture**, (2) **peer-to-peer (P2P) architecture**.

Client-Server Architecture

- In this architecture, there is an ALWAYS-ON host called the **server** which gets service requests from many other hosts which are called the **clients**. (Example: Web server services the requests from many browsers running on client hosts).
- The clients do not directly communicate with each other, rather one or more clients communicate with a server. (e.g. in web browsing, two browsers do not directly communicate).
- The server has a fixed, well-known address, called an IP address and server is always on, so, a client can always contact the server by sending a packet to the server's IP address.
- Some of the well-known applications with a client-server architecture are Web, FTP, Telnet, and e-mail. The client-server architecture is shown in Figure (=>)

Client-Server Architecture

- Often in a client-server application, a single-server host is incapable of keeping up with all the requests from clients. So, a data center, housing the large number of servers, is often used to create a powerful virtual server.



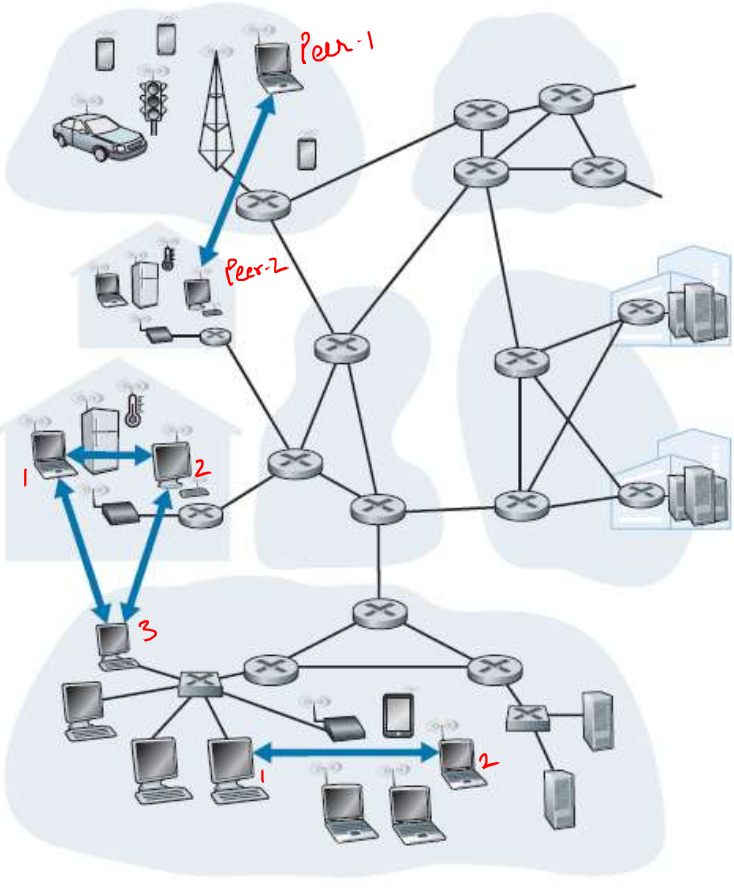
Peer-to-Peer (P2P) Architecture

- In a P2P architecture, there is no dependency on the dedicated servers (data centers), Instead the applications use the direct communications between pairs of intermittently connected hosts, called peers.
- These peers are the desktops and laptops controlled by users, with most of the peers residing in homes, universities, and offices.
- Because the peers communicate without taking any dedicated server in-between, this type of architecture is called the peer-to-peer architecture.
- An example of a popular P2P application is the file-sharing application BitTorrent.

Peer-to-Peer (P2P) Architecture

- One of the most compelling features of P2P architectures is their self scalability. For example, in a P2P file-sharing application, although each peer generates workload by requesting files, each peer also adds service capacity to the system by distributing files to other peers.
- P2P architectures are also cost effective, since they normally don't require significant server infrastructure.
- However, P2P applications face challenges of security, performance, and reliability due to their highly decentralized structure.
- The Peer-to-Peer (P2P) architecture is shown in Figure (=>)

(Peer-to-Peer (P2P) Architecture)

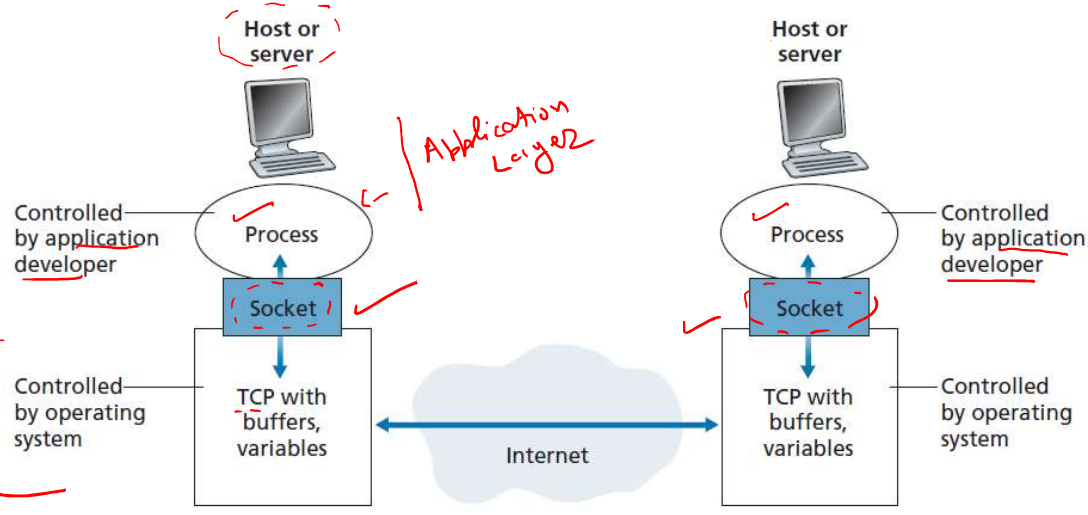


Interfacing our application program (process) with computer network

- Most of the applications consist of pairs of communicating processes (programs) sending messages to each other.
- Any message sent from one process to other process must *go through the underlying network*.
- A process sends messages into, and receives messages from, the network through a software interface called a **socket**.
- Figure (on next slide) illustrates socket communication between two processes that communicate over the Internet.

Interfacing our application program (process) with computer network

- Here we assume that the underlying transport layer protocol used by the processes is TCP protocol.
- A socket is the interface between the application layer and the transport layer within a host. So, it is also referred to as the **Application Programming Interface (API)** between the application and the network.
- The application developer has control of everything on the application-layer side of the socket but has a little control of the transport-layer side of the socket.



♦ Application processes, sockets, and underlying transport protocol

Addressing of the Processes (programs/applications)

- In order to send postal mail to a particular destination, the destination needs to have an address. Similarly, in order for a process (program) running on one host to send packets to another process running on other host, the ***receiving process needs to have an address***.
- To identify the receiving process, two pieces of information need to be specified: (1) the address of the host and (2) an identifier that specifies the receiving process in the destination host.
- The designation (receiving) host can be uniquely identified by its **IP address**.
- However, in addition to knowing the IP address of the destination host to which a message is to be sent, the sending process must also identify the receiving process (more specifically, the receiving socket) running in that host. This information is needed because in general a host could be running many network applications (processes/programs). A **destination port number** serves this purpose. Popular applications have been assigned specific port numbers. For example, a Web server is identified by port number 80. A mail server process (using the SMTP protocol) is identified by port number 25.

Functions of Application Layer Protocols

- An application-layer protocol defines how an application's processes, running on different end systems, pass messages to each other. In particular, an application-layer protocol defines:
  1. The types of messages exchanged, for example, request messages and response messages.
  2. The syntax of the various message types, such as the fields in the message and how the fields are defined.
  3. The semantics of the fields, that is, the meaning of the information in the fields.
  4. Rules for determining when and how a process sends messages and responds to messages.
- Example: The Web's application-layer protocol, HTTP, defines the format and sequence of messages exchanged between browser and Web server.

Thank you.