Computer Networks

2018A7PS0193P

January 28, 2021

1 Networks

Definition 1.1. A **network** is a shared infrastructure that allows users to communicate with each other.

The basic building blocks of a network are **nodes** and **links**. Nodes may be hosts or forwarding nodes. The end hosts communicate with one another through the **core network**, consisting of forwarding nodes. These nodes are connected via links called **network edges**.

End hosts are physically connected to the core network via the **access network**. This consists of many parts, such as the ethernet switch, router, etc.

1.1 Communication Models

Networks could have multiple communication models:

- **Client-Server model**: The client host requests, and receives the service from an always-on server. This server is also an end host, but has some special privileges.
- **Peer-peer model**: Here, there is minimal or no use of dedicates servers, as in Bit-Torrent. The clients directly communicate with one another.

1.2 Core Network Models

1.2.1 Circuit Switching

How is our core network made? One way to do this is via **circuit switching**. There are end-to-end resources reserved for a "call", like on a telephone network. There is no sharing of resources. Call setup needs to be done as a preparatory step. Circuit switching generally can be implemented by two different methods:

- **FDM**, which stands for Frequency Domain Multiplexing. The total frequency bandwidth is divided among the users, allowing them to send data simultaneously.
- **TDM**, which stands for Time Domain Multiplexing. Here the time is divided among the users (perhaps in a round robin fashion). As such, one user gets access to the entire bandwidth of the circuit, but only for a period in time.

1.2.2 Packet Switching

Another way is through **packet switching**, where data is sent over the net in discrete "chunks", called packets. This is how it is done on the Internet. The host takes the application message, and breaks into packets of length L bits. It then transmits packets into the access networks at transmission rate R, also called the bandwidth. Of course, this means that each packet faces a transmission delay of L/R.

Packet switching uses **store and forward**. The packets are stored at intermediate nodes before sending to the next node. The intermediate node checks for errors in the packets before transmitting, assuring the integrity of the packets.

When using packet switching, there may be four sources of packet delay:

- Nodal processing : The node performs error checking and checks the header for the destination of the packet.
- Queueing: When the arrival rate of the packets is faster than the sending rate, the node will keep the packets in a buffer queue. As such, there is a delay when the packet wait in the node queue.
- Propagation: This is the delay from propagation of the packets from a node to the next node, i.e., the outgoing delay. This depends on the medium of the wire, and is given by d/s, where d is the length of the connection and s is the speed.
- Transmission : This is the delay from transmission of packets into the node, i.e. the incoming delay.

1.3 Performance of a Network

The performance of a network can be measured by the following parameters:

- Delay

- Packet loss : This is the number of packets lost when transmitting. Some applications, like streaming, might not care too much about this.
- Throughput : This is the amount of bits transferred in unit time. This is important in some applications, such as for file transfer.

2 The Internet

The Internet is, in fact, a network of networks. The networks must be able to communicate despite using different applications running on different devices - i.e. it is heterogeneous.

As such, the Internet is full of different access ISP networks. How do end hosts on different access ISPs communicate with one another. Of course, if we directly connect them all, it would not be scalable as it would need $O(N^2)$ connections. We also cannot use a single global hub, since it would be difficult to find a single place to put it and connect the entire world.

Since a single global ISP cannot scale to connect the entire world, we use multiple global ISPs. These must be interconnected themselves. One way to do this is using **peering links**, which directly link two global ISPs. Another is to use **Internet Exchange Points**, called IXPs, to which multiple global ISPs can connect.

The Internet uses this system in a tiered manner - end hosts might connect to a regional ISP, which may then connect to a higher level country ISP, and so on.

Some corporations, like Google, have their own Content Distribution Networks (CDNs), and have their own network to bring services and content closer to users.

2.1 Layered Network Model

The Internet is based on a Layered Network Model known as **OSI**. Any device under OSI can have the following layers:

1. Physical
2. Data Link
3. Network
4. Transport
5. Session

6. Presentation

7. Application

Each of these layers depend on the one below (lower number) and export their services to the ones above (higher number).

The end hosts implement all 7 layers of the model. Those which implement the first 3 layers are called **routers** or Layer 3 devices. Routers are used to connect two different networks. Those which implement the first 2 layers are called **switches** or Layer 2 devices. They connect devices within a network, i.e., in Local Area Networks.

The Internet stack does not actually use all 7 layers - in fact it uses only 5. It removes the Presentation layer, which allows applications to interpret the meaning of data. It also removes the Session layer, which is used for synchronization, check pointing and recovery of data exchange. These functions are generally performed by the Application layer and/or the Transport layer. This Internet stack is known as **TCP/IP model**.

In the context of the Internet, these layers perform the following functions:

1. **Physical:** This layer delivers bits between the two endpoints of a link, e.g. copper, fiber, wireless, etc.

2. **Data Link:** This layer delivers packets between two hosts in a local area network. These are bridges and switches.

3. **Network:** This layer connects multiple networks, e.g. routers. This uses the Internet Protocol (IP).

4. **Transport:** This layer does process-process data transfer. It may use a multitude of protocols, including TCP, UDP, etc.

5. **Application:** This layer supports network applications. It may use FTP, SMTP, HTTP, etc.

2.2 IP Hourglass Architecture

One way to imagine the Internet architecture is as a hourglass. The IP interconnects multiple existing networks, and hides the underlying technology from applications. This provides minimal functionality (has a "narrow waist"). The tradeoff of this approach is that there are no assumptions being made, and as such no guarantee that something works.

2.3 Application Layer

A **Network application** is a program that runs on different end systems and communicates over a network. These are run only on the end hosts - core network devices do not run user application code.

The application architecture can run different application architectures:

- **Client-Server**: The server is an "always on" host, which has a permanent IP address. To be able to scale, there are generally large data centers acting as a virtual server. The clients communicate with the server. Unlike the server, they may be intermittently connected, and may have a changing dynamic IP address. These clients never directly communicate with one another.
- **Peer to peer** : Here, there is not always on server. Instead the end hosts directly communicate with one another. The peers are connected and may change IP addresses dynamically.
- **Hybrid of client server and peer to peer** : This is the case in Instant Messaging and Skype. In instant messaging, the chatting between two users is P2P, but to get the IP addresses of a user's friends, a central server is needed.

Processes communicate within the same host using interprocess communication, but they must communicate with different hosts by exchanging messages.

Definition 2.1. A **socket** is the interface between the application later and the transport layer within the host.

A process (a network application) send and receives messages using it's socket. This is a software entity, not a physical one.

To receive messages, each process must have some identifier. The IP address is not enough since it will only uniquely identify the host, but not the individual processes running on the host. So, we also use the port number to identify a process. For instance, an HTTP server would use port number 80, and a mail server would use port number 25.

2.4 Transport Layer

What transport services does an application need? They are as follows: