

Lecture 2: Computer Networks Review

Yuan Xue

Suppose you want to build a computer network. What technologies would serve as the underlying building blocks, what kind of software architecture would you design to integrate these building blocks into an effective communication service, and what would be the weaknesses in the design that may be exploited by attackers? Answering these questions is the overall goal of this lecture.

I. DIRECT LINK NETWORKS

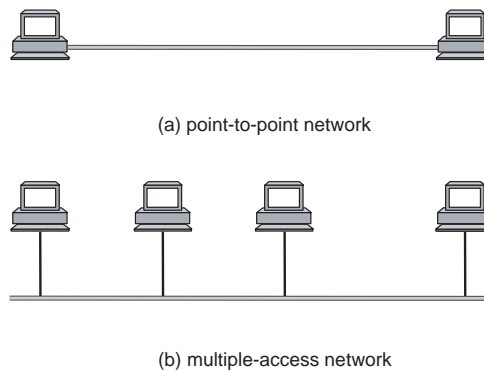


Fig. 1. Direct Link Network.

Starting from the simplest, we study the network in which all the hosts are directly connected by some physical medium (as shown in Fig. 1), such as a wire or a fiber. There are five additional problems that must be addressed before the hosts can successfully communicate with each other.

- *Encoding.* The first step to establish communication between two hosts is to turn binary data into the signals that the links are able to carry, and then to transform the signal back into the corresponding binary data at the receiving node. Let us ignore the details of modulation, and assume that we are working with two discrete signals: high and low. As shown in Fig. 2, different encoding mechanisms can encode bits into signals.

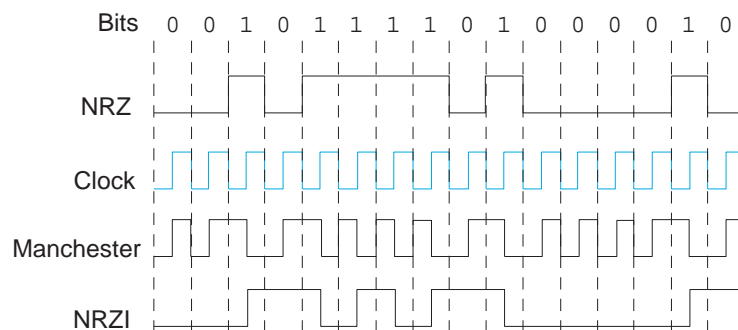


Fig. 2. Encoding Schemes.

- *Framing.* With encoding, we know how to transmit a sequence of bits over a point-to-point link. Framing then delineates the sequence of bits transmitted over the link into complete messages that

can be delivered. There are several approaches to address the framing problem, including the sentinel approach (*e.g.*, PPP) and the byte-counting approach for byte-oriented protocols; and high-level data link control (HDLC) for bit-oriented protocols.

- *Error detection.* Bit errors can be introduced into frames due to electrical interference or thermal noise. Error detection detects transmission errors. Several commonly techniques for error detection include cyclic redundancy check (CRC) (*e.g.*, Fig. 3 (a)), two-dimensional parity (*e.g.*, Fig. 3 (b)), and Internet checksum. Some error codes can only detect the errors, some codes are strong enough to correct errors.

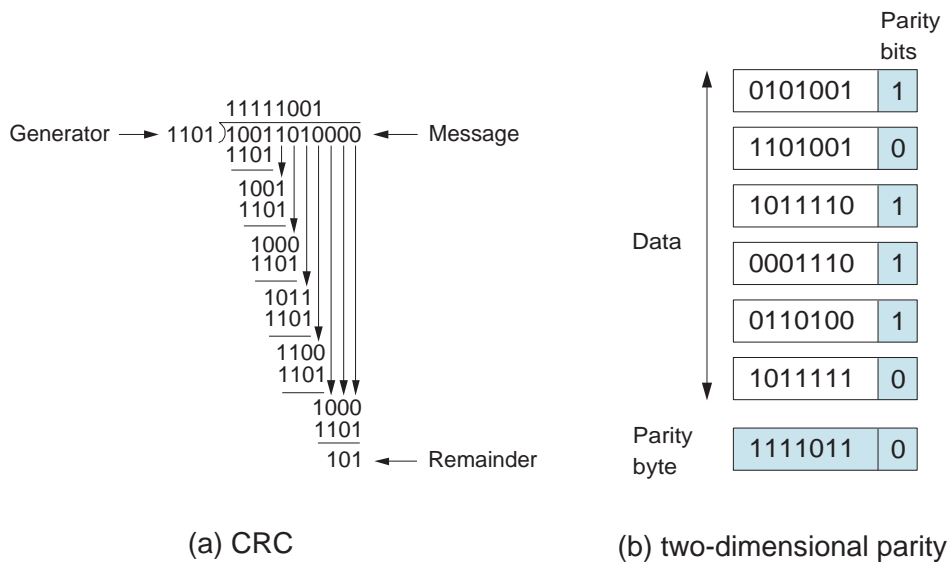


Fig. 3. Error Detection Schemes.

- *Reliability delivery.* When errors are detected, and can not be corrected, the corrupted frames must be discarded. Reliability delivery is sometimes implemented in point-to-point link network to recover from these lost frames.
- *Media access control.* When the link is shared by multiple hosts, their accesses to the link need mediation. Take Ethernet as an example. CSMA/CD (Carrier Sense Multiple Access / Collision Detection) is used to provide media access control. Essentially, participating hosts monitor the traffic on the link. If no transmission is taking place at the time, the particular host can transmit. If two hosts attempt to transmit simultaneously, this causes a collision, which is detected by all participating hosts. After a random time interval, the hosts that collided attempt to transmit again. If another collision occurs, the time intervals from which the random waiting time is selected are increased step by step. This is known as exponential back off.

Nearly all the networking functionalities described above are implemented in the network adaptor (as shown in Fig. 4): encoding, framing, error detection, and the media access control. In Ethernet, each adaptor has a unique Ethernet address, which is also the MAC address of the corresponding host. Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet. Each adaptor recognizes those frames addressed to its own address, and passes only those frames to the host. An adaptor can also be programmed to run in *promiscuous* mode, in which case it delivers all received frames to the host.

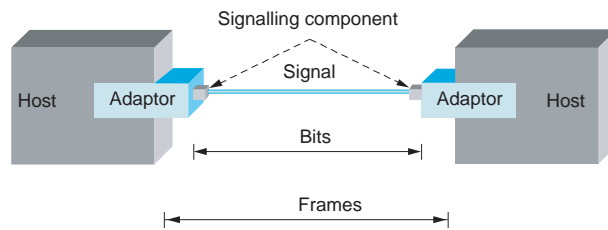


Fig. 4. Network Adaptor.

Even in such a simple direct link network, security threats still exist. Some examples include frequency jamming, eavesdropping (*e.g.*, packet sniffing), MAC address spoofing.

II. INTERNETWORKING

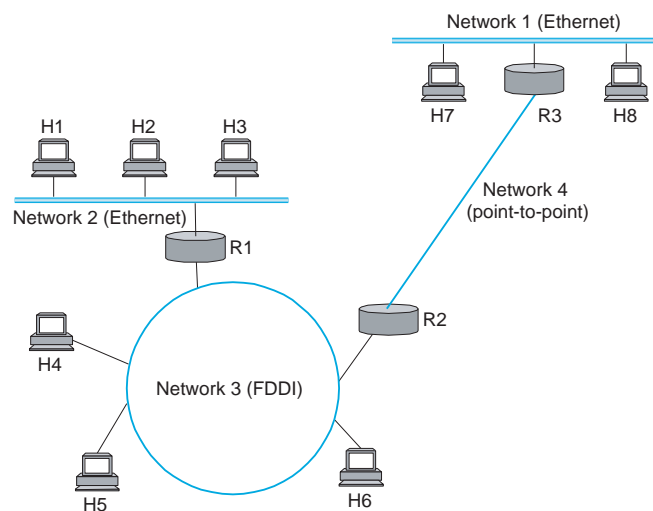


Fig. 5. Internetwork.

In Lecture 2, we have seen how to build a simple network using direct links. Direct link network can only connect a limited number of hosts and cover a small geographical area. To establish a network that has the potential to grow to global scale, we need to connect the direct-link networks. There are two important problems that must be addressed when connecting networks: *heterogeneity* and *scale*. As shown in Fig. 5, internetwork connects networks (*e.g.*, Network 1 – Network 4) with different technologies with *routers* (*e.g.*, R1, R2, R3). This presents the problem of *heterogeneity*. To understand the problem of scale, let's consider the growth of the Internet, which has roughly doubled in size each year for 20 years. The Internet Protocol (IP) is the key tool to build scalable, heterogenous internetworks.

A. Addressing

Addressing is the task of providing suitable identifiers for all these hosts in internetworks. To address the problem of scalability, IP uses *hierarchical* addresses. Specifically, IP addresses has 32 bits consisting of two parts: a network part and a host part. The network part of an IP address identifies the network to which the host is attached; all hosts attached to the same network have the same network part in their IP addresses. The host part then identifies each host uniquely on that particular network. Moreover, IP

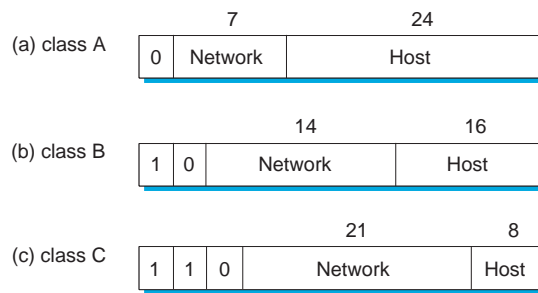


Fig. 6. IP addresses.

addresses are divided into three different classes, as shown in Fig. 6. This address scheme has a lot of flexibility, allowing networks of vastly different sizes to be accommodated fairly efficiently.

B. Fragmentation and Reassembly

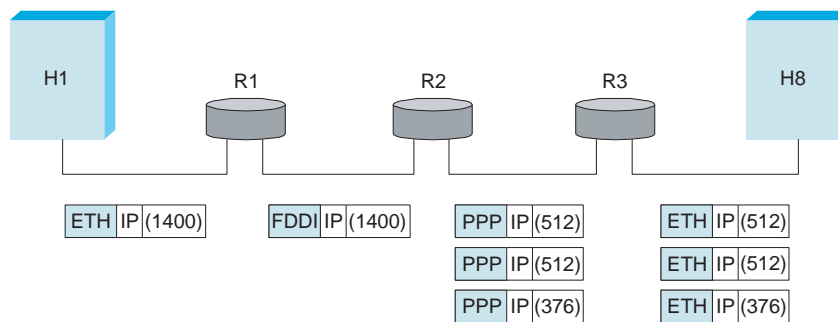


Fig. 7. IP datagrams traversing a sequence of different physical networks as in Fig. 5.

One of the problems to provide a uniform host-to-host service over a heterogeneous collection of networks is that each network technology has its own definition of packet size. IP provides a means by which packets can be fragmented and reassembled when they are too big to cover a given network. As shown in Fig. 7, each fragment is a self-contained IP datagram that is transmitted, independent of the other fragments; Each IP datagram is re-encapsulated for each physical network over which it travels.

C. Routing and Forwarding

Now let's look at the basic mechanism of IP by which packets are delivered among different networks: *forwarding* is the process of IP router to take a packet from an input and send it out on the appropriate output based on its forwarding table; *routing* is the process of building up the forwarding table.

R2

Network#	nexthop
1	R3
2	R1

Fig. 8. Forwarding table of R2 in the network shown in Fig. 5 .

Forwarding IP datagrams can be handled in the following way. A datagram that carries the IP address of the destination host, is sent from a source host, possibly passing through several routers along the way, and arrives at its destination. Any node (a host or a router) that handles the datagram, first compares the network part of the destination address with the network part of its addresses (routers may have more than one addresses, as they have multiple network interfaces that are connected to two or more networks). If a match occurs, then that means the destination lies on the same physical network, and the packet can be directly delivered over that network. If the node is not connected to the same physical network as the destination node, then it needs to send the datagram to a router based on its forwarding table, which is known as the *next hop* router. The forwarding table of Fig. 5 is shown in Fig. 8.

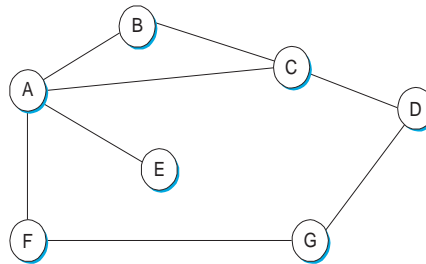


Fig. 9. Network as a graph.

Now the question is how do routers acquire the information in their forwarding table? This is addressed by the routing process. Routing is essentially a problem of graph theory. If we view a network as a graph, as in Fig. 9, the nodes of the graph may be either hosts, routers, or networks; the edges of the graph correspond to the network links. Each edge has an associated cost, which gives some indication of the desirability of sending traffic over that link. The basic problem of routing is to find the lowest-cost path between any two nodes. In more practical networks, routing is achieved by running routing protocols among the nodes. *Distance vector* protocol and *link state* protocol are the two main classes of routing protocols.

The idea of *distance vector* protocols (e.g., RIP) is that: each node constructs a vector containing the “distance” (cost) to all other nodes and distributes that vector to its immediate neighbors. Upon receiving the distance vector from neighbors, each node updates their own distance vectors and routing tables accordingly. Fig. 10 shows an example of the development of distance vector and routing table.

The idea behind *link state* protocols (e.g., OSPF) is that: every node knows how to reach its direct neighbors. And if this knowledge is disseminated to every node in the network, then every node will have enough knowledge to build a complete topology of the network, which is sufficient to find the shortest path to any node in the network and build the forwarding table.

In practice, Internet is organized into autonomous systems (or routing domain) to provide hierarchically aggregate routing information in a large internetwork to improve scalability. Here the routing problem is divided into two parts: intra-domain routing within a single autonomous system and inter-domain routing between autonomous systems. RIP and OSPF are used for intra-domain routing; while BGP (Border Gateway Protocol) is the routing protocol used in Internet for inter-domain routing.

The complexity of an internetwork brings a lot more security issues. For example,

- *IP spoofing* is one of the most common forms of camouflage in IP network. In IP spoofing, an attacker gains unauthorized access to a computer or a network by making it appear that a malicious

	Global								A's vector		
Initial		A	B	C	D	E	F	G	dest	cost	nexthop
	A	0	1	1		1	1		B	1	B
	B	1	0	1					C	1	C
	C	1	1	0	1				D		N/A
	D			1	0			1	E	1	E
	E	1				0			F	1	F
	F	1					0	1	G		N/A
	G				1		1	0			
Final		A	B	C	D	E	F	G	dest	cost	nexthop
	A	0	1	1	2	1	1	2	B	1	B
	B	1	0	1	2	2	2	3	C	1	C
	C	1	1	0	1	2	2	2	D	2	C
	D	2	2	1	0	3	2	1	E	1	E
	E	1	2	2	3	0	2	3	F	1	F
	F	1	2	2	2	2	0	1	G	2	F
	G	2	3	2	1	3	1	0			

Fig. 10. Distance vector routing.

message has come from a trusted machine by “spoofing” the IP address of that machine.

- *Authentication of routing messages.* Routing messages are used to deliver correct topology information for the construction of forwarding table. If an attacker modifies the routing messages, it will disrupt the routing operation of the whole network.

III. END-TO-END PROTOCOL

We have established an internetwork connecting together a collection of hosts. The next problem is to turn this host-to-host packet delivery service into a communication channel between application processes. This is sometimes called the end-to-end protocol. Two end-to-end protocols of Internet are UDP and TCP.

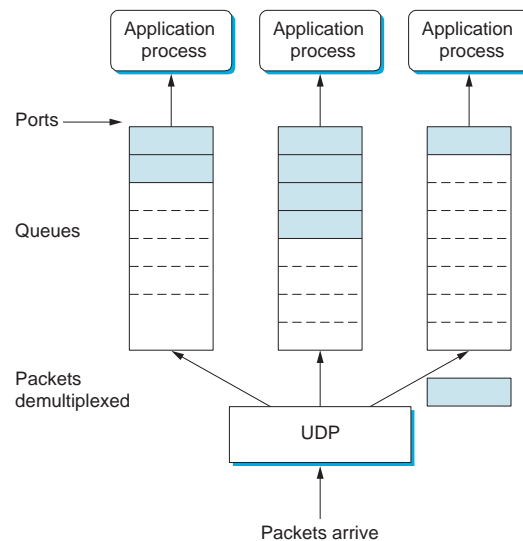


Fig. 11. Port.

Since many processes may run on the same hosts, the protocols need to add a level of de-multiplexing. To identify the processes, an abstract locator, often called a *port* is assigned to each process, as shown in Fig. 11. Port provides a window to a networked system. Attackers often use port scanning to gain knowledge of a system before they launch attacks.

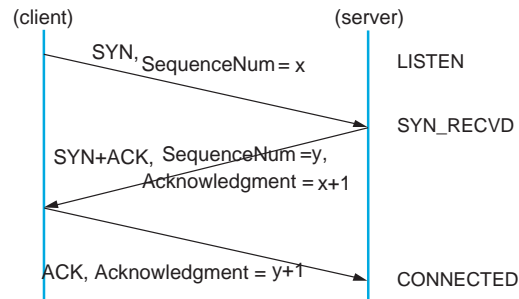


Fig. 12. 3-way handshake in TCP.

Besides the de-multiplexing functionality, which is also provided by UDP, TCP provides a more sophisticated connection-oriented, reliable network service. Connection-oriented network service means that the participants in a TCP session must first build a connection - via the 3-way handshake procedure as shown in Fig. 12. TCP also guarantees the reliable, in-order delivery of a stream of bytes via sequences and acknowledgements. Finally TCP provides flow control/congestion control to determine an appropriate sending rate of the byte stream.

IV. ARCHITECTURE

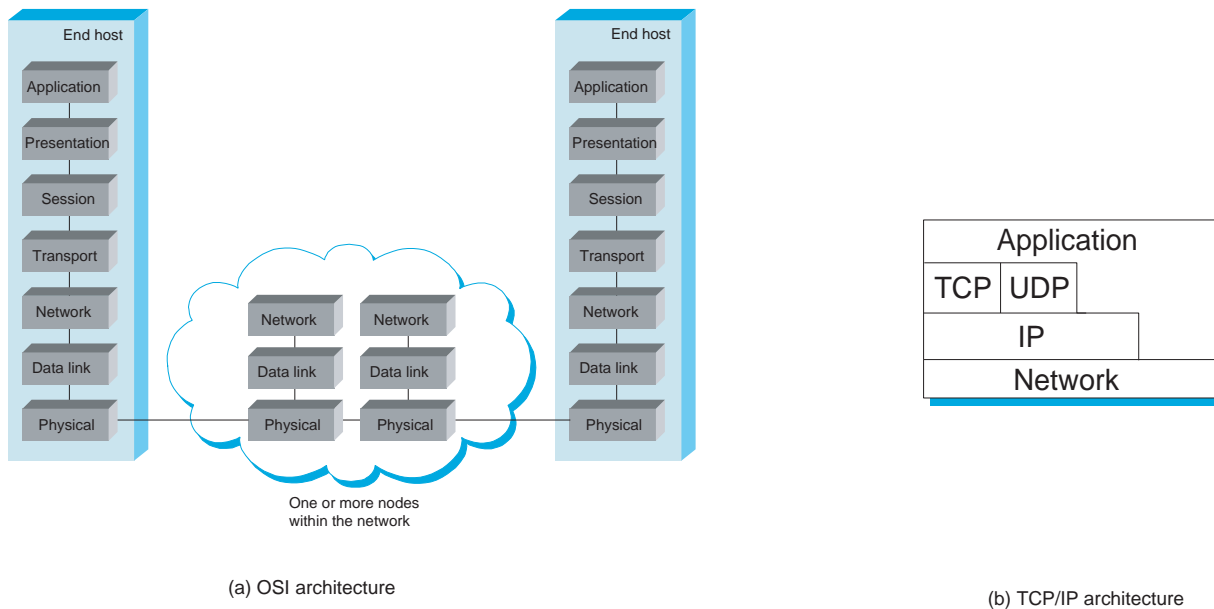


Fig. 13. Network Architecture.

To organize all the above networking protocols in a systematic way, an architecture needs to be defined. Open Systems Interconnection (OSI) (Fig. 13 (a)) architecture is one of the first defined architectures for

connecting computers. The Internet architecture, which is also called the TCP/IP architecture (Fig. 13 (b)), is the most widely adopted architecture.