TCP/IP Suite

Prabhaker Mateti, Wright State University

Introduction 543

Layers 543
The OSI Model 543
The DoD Model 544
The Hourglass Model 544
Protocol Stack 544
Lower Layers 544

Internet Protocol 547
IP Addresses 547
IP Header 548
Routing Protocols 548
IP Fragments 548
Mobile IP 549

Transmission Control Protocol 549
Ports and Connections 549
Reliable Transmission 549
State Diagram 550
Three-Way Handshake 550
Four-Way Handshake 550
Timers 550
Flow and Congestion Control 551

UDP, ICMP, DNS, ARP, and RARP 551
User Datagram Protocol 551
Internet Control Message Protocol 551
Domain Name Service 552
Address Resolution Protocol 552
Applications 553
File Transfer Protocol, Telnet, and rlogin 553
File Transfer Protocol 553
Telnet 554
rlogin 554
Secure Shell 554
Dynamic Host Configuration Protocol 554
HyperText Transfer Protocol 554
Security 555
Security Exploits 555
Security Enhancements 558
Conclusion 559
Glossary 559
Cross References 559
References 559
Further Reading 560

INTRODUCTION

It is difficult to imagine modern living without the Internet. It connects all kinds of computer systems from supercomputers, costing millions of dollars, to personal computers, worth no more than a couple of hundred. The networks that connect them are varied, from wireless to wired, from copper to fiber. All of this is enabled by protocols and software collectively known as the TCP/IP Internet protocol suite or simply TCP/IP. TCP/IP is an open system. Its protocol specifications are public documents freely downloadable. Many of the implementations of the protocols are also open source. TCP/IP details constitute one or more college courses on computer networks. Entire textbooks have been written on the topic that this chapter is attempting to cover. Our goal is to present the practical TCP/IP landscape as it is today. This chapter describes the core protocols known as IP; TCP; essential protocols named UDP, ARP, ICMP, and DNS; and a few application protocols based on these. This chapter also highlights the security issues.

LAYERS

A computer system communicates with another system by sending a stream of bytes. The communication is actually between a process running on one system and one running on the other system. The two processes communicate information in a preagreed form known as a protocol. Computer protocols are is easier to understand as a stack of layers, each layer providing the functionality needed by the layer above it. In each layer, there are one or more protocols. There are three models of layers: the OSI, the DoD, and the “hourglass” models.

The OSI Model

The OSI model is officially recognized by the ISO (International Standards Organization). The OSI (open systems interconnection) model of computer networks has seven layers.

1. The bottom-most layer provides the physical means of carrying the stream of bits. Ethernet, Fast Ethernet, Wireless 802.11, T-carrier, DSL (digital subscriber line), and ATM are examples of this layer. All media are considered functionally equivalent. The differences are in speed, convenience, and cost. Converters from one media to another exist and make it possible to have different physical layers in a computer network.

2. The data link layer takes the raw stream of bits of the physical layer and, using its encoding functionality, sends and receives a meaningful message unit called a frame and provides error detection functions. A frame includes checksum, source and destination addresses, and data. The frame boundaries are special patterns of bits. Software of this layer will retransmit a frame if it is damaged, say because of a burst of noise on the physical layer. The data link layer is divided into the media access control (MAC) sublayer, which controls how a computer on the network gains access to the data and permission to transmit it, and the logical link control (LLC) sublayer, which controls frame synchronization,
The TCP/IP Suite

1. Flow control, and error checking. The data link layer describes the specification of interface cards to specific types of networks (e.g., Ethernet, and token ring). Example protocols from the TCP/IP suite that occupy this layer are SLIP and PPP.

2. The network layer accepts messages from the source host, converts them into packets of bytes, and sends them through the data link. This layer deals with how a route from the source to the destination is determined. This layer also deals with congestion control. The IP, address resolution protocol (ARP), reverse ARP (RARP), Internet control message protocol (ICMP), and IGMP belong to this layer.

3. The transport layer transfers data and is responsible for host-to-host error recovery and flow control. The TCP and UDP belong to this layer. UDP provides "connectionless service" and TCP provides "connection-oriented service."

4. The session layer establishes, manages, and terminates connections between the programs on the two hosts that are communicating. The concepts of ports and connections belong to this layer.

5. The presentation layer provides independence from possibly different data representations of the host machines. The HTTP (hypertext transfer protocol), FTP (file transfer protocol), telnet, DNS, SNMP (simple network management protocol), NFS (network file system), and so on belong to this layer.

6. The application layer supports the end-user invoked programs. FTP, HTTP, IMAP (Internet Message Access Protocol), SMTP (Simple Mail Transfer Protocol), SNMP, SOCKS, telnet, X-Window, Web services, and so on are part of this layer.

The DoD Model

The practical world of TCP/IP networking was in full use by the time the OSI model was formulated. Its unofficial model is referred to as the TCP/IP model, the DoD (U.S. Department of Defense) model, or even more simply the Internet model. The DoD model organizes the networks into four layers.

1. The link (or network access) layer deals with delivery over physical media. This layer maps to the data link and physical layers of the OSI model.

2. The network (or Internet) layer deals with delivery across different physical networks that connect source and destination machines. IP, ICMP, and IGMP are in this layer.

3. The transport (or host-to-host) layer deals with connections, flow control, retransmission of lost data, and so on. TCP, UDP are in this layer.

4. The application (or process) layer deals with user-level services, such as SMTP, FTP, login, SSH, SSL, POP, and HTTP. This layer corresponds to the session, presentation and application layers of the OSI model.

The Hourglass Model

Internet protocols can be described as following an hourglass model with IP at the neck of the hourglass. The hourglass illustrates dependencies among the underlying networks, IP, and the applications.

Protocol Stack

The computer network literature talks of protocol stacks. Each item in the stack is a layer of software that implements a collection of protocols. In Figure 2, the relative heights indicate the level of functionality and the dependency of the layers. Except for the layer marked "physical" all others are software layers. The protocol stack is an integral component in a modern operating system.

In each protocol, there is a stream of bytes known as a frame, a datagram, a packet, or a segment depending on the "level." The data unit of each layer is encapsulated by the layer below it, similarly to how sheets of paper are enclosed in an envelope. Each layer adds control information typically prefixed to the data before passing on to the lower layer. An application data unit (AP) is encapsulated by the TCP layer, which prefixes a TCP header (TCPH) as (TCPH (AP)). The IP layer encapsulates it as (IPH (TCPH (AP))), where IPH is the IP header. Assuming Ethernet, it encapsulates it as (ETH (IPH (TCPH (AP)))) FSC, where ETH is the Ethernet header and FSC is a frame check sequence generated by the Ethernet hardware.

Lower Layers

The OSI physical, data link, and network layers are collectively called the lower layers. In this section, we describe a few prominent entries from the lower layers.

Figure 1: The hourglass model.

Figure 2: Protocol stack.
All hosts attached to Ethernet are connected to a shared signaling medium. Ethernet signals are transmitted serially, one bit at a time, over the shared medium that every attached host can observe. When a frame is sent out on the Ethernet, every controller on the local unswitched network can see the frame. Thus, Ethernet is a broadcast medium. To send data, a host waits for the channel to become idle and then transmits its frame. All hosts on the network contend equally for the transmission opportunity. Access to the shared medium is governed by the MAC mechanism based on the carrier sense multiple access with collision detection (CSMA/CD) system. This ensures that access to the network channel is fair and that no single host can lock out other hosts. If two or more devices try to transmit at the same instant, a transmit collision is detected, and the devices wait a random (but short) period before trying to transmit again.

Switched Networks. Today a typical end user connects to a full duplex switched Ethernet that uses switches instead of hubs to connect individual hosts or segments. A hub is an OSI physical layer device. It transmits the frames received from one port to all other ports it has. A switch is an OSI data link layer or network layer device. It builds ("learns") a table of ports and the MAC address it is connected to, reads the destination address of each frame, and forwards a frame it receives to only the port connected to the destination MAC address. Switched Ethernets extend the bandwidth by replacing the shared medium of legacy Ethernet with a dedicated segment for each station. Switched networks use either twisted pair or fiber optic cabling, with separate conductors for sending and receiving data. Collision detection is not necessary because the network channel is fair and that no single host can access the medium. End stations and the switch can transmit at will achieving a collision-free environment. Switched Ethernets also mitigate sniffing. However, many commercial switches can be “overwhelmed” into behaving as though they are hubs.

IEEE 802.11 a/b/g Wireless Networks
This section briefly describes wireless networks known as the IEEE 802.11 family (http://www.ieee802.org/11/). For a detailed treatment, read chapter 44 Wireless Local Area Networks. The 802.11a operates at a theoretical maximum...
The IEEE 802.11 MAC header.

Frames help in the delivery of data. The AP (access point) is a station that establishes and maintains communications. A dot-11 frame consists of a MAC header (see Figure 4) followed by a frame body of 0 to 2312 bytes and an FCS. There are three classes of frames. The management frames establish and maintain communications. The control frames help in the delivery of data. The data frames encapsulate the OSI network layer packets. These contain the source and destination MAC address, the BSSID, and the TCP/IP datagram.

**Stations and Access Points.** A wireless network station provides a radio link to another station. The station has a MAC address, a world-wide-unique 48-bit number, assigned to it at the time of manufacture, just as wired network cards do. An access point (AP) is a station that provides frame distribution service to stations associated with it. Each AP has a 0- to 32-byte-long service set identifier (SSID) that is used to segment the airwaves for usage. The AP itself is typically connected by wire to a LAN.

**Frames.** A dot-11 frame consists of a MAC header (see Figure 4) followed by a frame body of 0 to 2312 bytes and an FCS. There are three classes of frames. The management frames establish and maintain communications. The control frames help in the delivery of data. The data frames encapsulate the OSI network layer packets. These contain the source and destination MAC address, the BSSID, and the TCP/IP datagram.

**Authentication and Association.** Data can be exchanged between the station and AP only after a station is authenticated and associated with an AP. The association is a two-step process. A station that is currently unauthenticated and unassociated listens for management frames known as Beacon frames. The station and the AP mutually authenticate themselves by exchanging authentication management frames. In the second step, the station sends an association request frame, to which the AP responds with an association response frame that includes an association ID to the station. A station can be authenticated with several APs at the same time but associated with at most one AP at any time.

**WEP and IEEE 802.11i.** Wired equivalent privacy (WEP) is a shared-secret key system encrypting the payload part of the frames transmitted between a station and an AP. The WEP is intended to protect wireless communication from eavesdropping and to prevent unauthorized access to a wireless network. Unfortunately, WEP is insecure. The IEEE 802.11i was ratified in 2004. It provides robust encryption and authentication.

**Asynchronous Transfer Mode**

Asynchronous transfer mode (ATM) is widely deployed as a backbone technology. ATM uses 53-byte fixed-length packets called cells for transport. A 48-byte payload divides the data into different types. The ATM layer contains 5 bytes of additional information, referred to as overhead. Information is divided among these cells, transmitted, and then reassembled at their final destination. ATM is connection-oriented. ATM itself consists of a series of layers. Its physical layer is based on various transmission media that range in speed from kilobits per second to gigabits per second. The layer known as the adaptation layer holds the bulk of the transmission.

**Serial Line Internet Protocol**

A serial network is a link between two computers over a serial line, which can be a dial-up connection over telephone lines or a direct connection between the serial ports of two computers. Serial line Internet protocol (SLIP; RFC 1055) defines the encapsulation protocol, just as an Ethernet frame enwraps an IP packet. Unlike Ethernet, SLIP supports only the IP and not multiple protocols across a single link. The serial link is manually connected and configured, including the specification of the IP address. SLIP provides no mechanisms for address negotiation, error correction, or compression. However, many SLIP implementations record the states of TCP connections at each end of the link, and use header compression that reduces the size of the combined IP and TCP headers from 40 to 8 bytes.

**Point-to-Point Protocol**

Point-to-point protocol (PPP; RFC 1661, RFC 2131) replaces the older SLIP, and is an encapsulating protocol for IP and other protocol datagrams over serial links. The encapsulation and framing adds 2, 4, or 8 bytes depending on the options chosen. PPP includes a link control protocol.
Dotted Quads and Octets

IP addresses are typically written in a dotted-quad notation, such as a.b.c.d, where a is the first byte, b the second, c the third, and d the fourth byte. Each of a to d is an octet, a number in the range 0 to 255.

Three address ranges known as class A, class B, and class C are of importance. In a class A address, the 0-th bit is always a 0, bits 1 through 7 identify the network, and bits 8 through 31 identify the host, permitting 2^24 hosts on the network. In a class B address, the bit 0 is always a 1, but 1 is always a 0, bits 2 through 15 identify the network, and bits 16 through 31 identify the host, permitting 2^16 hosts on the network. In a class C address, bits 0 and 1 are both always 1, bits 2 is a 0 always, bits 3 through 23 identify the network, and bits 24 through 31 identify the host, permitting 2^8 hosts on the network.

CIDR Nomenclature

CIDR (classless interdomain routing) model, first published in 1993, solves the problems of efficient utilization of IP address space (RFC 1520). Class C, with a maximum of 254 host addresses, is too small; whereas class B is too large to be densely populated, and making routing tables more compact.

A subnet is a collection of hosts whose IP addresses match in several bits indicated by the ones in a sequence of 32 bits known as a subnet mask, also written in the dotted-decimal notation. Thus, 255.255.255.0 is a mask of 24 ones followed by 8 zeroes. Because of this structure, the mask is also written as /24. Nodes and routers use the mask to identify the address of the network on which the specific host resides. The address of the network is the bitwise AND of the IP address and the mask. The host ID is the bitwise AND of the IP address and the complement of the mask. Occasionally, a network node X needs to discover certain information from other nodes, but the node X does not know the addresses of these others. In such situations, X broadcasts using special destination IP addresses. The direct broadcast address of X is the address whose host ID is all ones and whose network address equals that of X. The limited broadcast address is 255.255.255.255.

Public and Private Address Ranges

The following three blocks of the IP address space is intended for private internets: 10.0.0.0 to 10.255.255.255 (10/8 prefix, class A), 172.16.0.0 to 172.31.255.255 (172.16/12 prefix), and 192.168.0.0 to 192.168.255.255 (192.168/16 prefix, class C). That is, on the Internet at large, there must never be IP packets whose source or destination addresses are from the above ranges.

Most operating systems are internally structured to depend on the presence of a network layer. To facilitate this, the address 127.0.0.1 is assigned as the so-called address of the localhost (spelled as one word) and 127.0.0.0 as the localnetwork (spelled as one word). Packets sent to this address do not actually travel onto the external network. They simply appear as received on the local (artificial) device. When a machine is physically moved from one network to another, we must reassign an IP address that belongs to the new network. This is one of the problems that mobile IP solves.
IP Header
An IP header is a sequence of bytes that the IP layer software prefixes to the data it receives from the higher layers. The resulting IP header plus the data is given to the lower layer (e.g., the Ethernet card device driver). The byte layout of IP headers is shown in Figure 5. The header may or may not have the IP Options field. Except for this field, all other fields are fixed in length as shown. Minimally (i.e., without the options), the IP header is 20 bytes in length. With IP options, an IP header can be as long as 60 bytes. IP version 6 is discussed later. The value of IHL multiplied by 4 is the length of the IP header in bytes. The type of service field specifies the "relative urgency" or importance of the packet. Total length is a 2-byte field giving the length, in bytes, of the entire packet including the header, options (if any), and the packet data.

The maximum length of an IP datagram is 65535 bytes. (IP over Ethernet limits this to 1500.) The identification field, flags, and fragment offset are used to keep track of the pieces when a datagram must be split up as it travels from one router to the next. IP fragmentation is discussed further below. The time to live (TTL) is a number that is decremented by 1 whenever the datagram passes through a router node. When it goes to 0, the datagram is discarded, and an error message is sent back to the source of this packet. The protocol field identifies the protocol of the data area. The header checksum field is a one's complement arithmetic sum of the entire header viewed as a sequence of 16-bit integers. The source address is the datagram's sender IP address, and destination address is the IP address of the intended recipient. The IP options may or may not be present. When present, its size can be one or more bytes.

Routing Protocols
When the source S and the destination D are on the same network, we have a direct delivery of the packet that does not involve routers. When the two hosts are not on the same network, in general, there can be multiple paths between the source and destination. Because of failures, maintenance, and other reasons, the intermediate nodes known as routers may come on or off during the delivery of packets. Thus, consecutive packets sent by a host S to a destination D may (have to) travel entirely disjoint routes depending on how the network is connected. The typical network host has only one NIC and hence is on only one network, and sending and receiving of network traffic is secondary to its main functionality.

Routers are specialized computer systems whose primary function (often their sole function) is to route network traffic. Routers must have multiple NICs, each on a separate network. A router examines the destination IP address of a packet and consults its routing tables that record information regarding where to deliver a packet next so that definitive progress is made in moving the packet closer to its final destination. Routers are layer 3 devices.

Every network host (including routers) has a routing table, which can be visualized as a table of two columns: To send the packet to a final destination given in column 1, send the packet to the next hop whose IP address is given in column 2. The size of such a table can be greatly reduced by parameterizing the column 1 by its network address and also by including a default row in the table that acts as a catch-all. The default row indicates the next hop IP address for any packet whose destination network address does not match that of any other row. Once the next hop IP address is determined, the router uses the lower layer address (such as the Ethernet MAC) to deliver the packet to the next hop.

The routing table of an ordinary host rarely changes from boot-up to shut down. The tables of routers, however, must be dynamic and adjust to changing conditions, perhaps by the millicisecond, of the Internet. Routing protocols keep the routing tables up-to-date. The Internet is a network of autonomous networks. Interior gateway protocols (IGPs) maintain the routing tables within an autonomous network. Routing information protocol (RIP) and open shortest path first (OSPF) are examples of IGPs. Border gateway protocol (BGP) is the most common protocol in use for routing among autonomous networks.

IP Fragments
When datagrams are too large to be sent in a single IP packet, because of interface hardware limitations, for
example, they can be split up by an intermediate router unless prohibited by the Don’t Fragment flag. IP fragmentation occurs when a router receives a packet larger than the maximum transmission unit (MTU) of the next network segment. All such fragments will have the same identification field value, and the fragment offset indicates the position of the current fragment in the context of the pre-split-up packet. Intermediate routers are not expected to reassemble the fragments. The final destination will reassemble all the fragments of an IP packet and pass it to higher protocol layers (such as TCP or UDP).

Mobile IP
As the mobile network host moves, its point of attachment may change, and yet to maintain existing transport-layer connections, it must keep its IP address the same.

The mobile node uses two IP addresses. The home address is static and is used to identify TCP connections. The care-of address changes at each new point of attachment. Whenever the mobile node moves, it registers its new care-of address with its home agent. The home agent redirects the packets to the current care-of address by constructing a new IP header that contains the care-of address as the destination IP address. This new header encapsulates the original packet, causing the home address to have no effect on the routing of the encapsulated packet until it arrives at the care-of address. When the packet arrives at the care-of address, the effect of this “tunneling” is reversed so that the packet once again appears to have the home address as the destination IP address. Mobile IP discovery of the care-of address uses an existing standard protocol called router advertisement (RFC 1256). A router advertisement carries information about default routers, and in addition carries further information about one or more care-of addresses. Home agents and care-of agents typically broadcast these advertisements at regular intervals (say, once every few seconds). A mobile node that needs a care-of address will multicast a router solicitation. An advertisement also informs the mobile node whether the agent is a home agent, a care-of agent, or both and therefore whether it is on its home network or a care-of network and about special features provided by care-of agents (for example, alternative encapsulation techniques).

The registration of the new care-of address begins when the mobile node, possibly with the assistance of the care-of address, sends a registration request to the home agent. The home agent typically updates its routing table. Registration requests contain parameters and flags that characterize the tunnel through which the home agent will deliver packets to the care-of address. The triplet of the home address, care-of address, and registration lifetime is called a binding for the mobile node. The home agent authenticates that registration was originated by the mobile node.

Each mobile node and home agent compute an unforgeable digital signature using one-way hash algorithm MD5 ([Message Digest 5 (RFC 1321)] with 128-bit keys on the registration message, which includes either a time stamp or a random number carefully generated. Occasionally a mobile node cannot contact its home agent. The mobile node tries to register with another home agent by using a directed broadcast IP address instead of the home agent's IP address as the target for the registration request.

**TRANSMISSION CONTROL PROTOCOL**

TCP (RFC 793, RFC 3168) offers a client process a connection to a server process. This connection needs to be established, as needed. Once this connection is established, the TCP protocol guarantees the correct (both in content and in order) delivery of the data. TCP sends its message content over the IP layer and can detect and recover from errors. TCP, however, does not guarantee any speed of delivery; even though it offers congestion control.

**Ports and Connections**

Port numbers are used by the transport layer for multiplex communication between several pairs of processes. To each message, this layer adds addresses, called port numbers. The port numbers would have been assigned by the OS to certain processes. Thus, a connection is uniquely identified by a quadruple: source and destination IP addresses and source and destination port numbers. In this context, we often use quintuple, including the specific protocol (TCP/UDP/etc.) in use as the fifth element to make it clear that the port namespaces for UDP and TCP are separate. The IP addresses are supplied by the IP layer.

The ports 0-1023 are reserved for specific well-known services provided by privileged processes. For example, HTTP officially uses port 80, telnet officially uses port 23, and DNS officially uses port 53. The dynamic or private ports range from 1024 to 65535. Client processes and non-standard services are assigned port numbers by the operating system at run time. On most computer systems, there is a list of these port numbers and service names in a file named etc/services.

**Reliable Transmission**

TCP requires that every segment include an acknowledgment of the last data segment received in the other direction. TCP is a sliding window protocol with time-out and retransmits. If the sender does not receive an acknowledgment within the time-out period, it retransmits the segment. Acknowledgments are piggybacked on reply data. The window size specifies the number of bytes the receiver has as buffer space. The sender continues to send and slides the window ahead as long as acknowledgments are being received for bytes within the window.

TCP messages, called segments, are sent as one or more IP datagrams. A TCP header follows the IP header, supplying information specific to the TCP protocol. Figure 4 contains the details of the TCP segment. The letters (U/A/P/R/S/F) in the fourth row of the segment are abbreviated names for control bit flags: URG, urgent pointer field significant; ACK, acknowledgment field significant; PSH, push function; RST, reset the connection; SYN, synchronize sequence numbers; and FIN, sender is finished with this connection. These are further explained below.

The sequence number, together with the acknowledgment number, serves as a ruler for the sliding window protocol. While establishing the connection, the SYN flag
is set to 1, and the client and the server exchange their initial sequence numbers.

The acknowledgment number is valid only when the ACK bit is set. This field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established, this is always included. The DataAck number multiplied by 4 is the number of bytes in the TCP header. This indicates where the data begin. Window size is described under Congestion Control. Urgent pointer is valid when Urg is 1. Its value is a positive offset from the sequence number in this segment. Options, if any, are given at the end of the TCP header and are always a multiple of 8 bits in length. All options are included in the checksum. An option can be just a single byte, or it can be a byte of option-kind, followed by a byte of option-length and the actual option-data bytes. The option-length counts the two bytes of option-kind and option-length as well as the option-data bytes.

State Diagram

A TCP server process starts its life by passively opening a port and starts to listen to connection attempts from clients. This process causes a number of changes in the information maintained by the TCP layer software. These transitions are described by the state diagram shown in Figure 5. Below we describe two handshakes that establish a connection and close a connection.

Three-Way Handshake

This establishes connection between the initiating node (say A, the client) and the receiving node (say B, the server) of packets as follows:

1. A: “I would like to talk to you, B.” A sends a packet with SYN = 1, and the initial sequence number (ISN) chosen by A to B.
2. B: “OK, let’s talk.” B replies with a SYN + ACK packet (i.e., SYN = 1, ACK = 1, acknowledgment number = sequence number received +1, and SYN =1 with sequence number set to the ISN of the server). If B was unwilling, it responds with a RST = 1 packet refusing the request for service. At this point, the state of the connection is known as half-open.
3. A: “Thanks for agreeing!” A sends a packet with ACK = 1, acknowledgment number = ISN of B + 1, SYN = 0, sequence number = previous sequence number + 1. The sequence number for SYN = 1 can be a zero, but that is not secure, so the sequence number is randomly chosen.

Here is an example where the client is on port 1037, establishing a connection with a service on port 80 (typically HTTP).

<table>
<thead>
<tr>
<th>SYN</th>
<th>ACK</th>
<th>src</th>
<th>dst</th>
<th>sequence-acknowledgement number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1037</td>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1037</td>
<td>80</td>
<td>102723770</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1037</td>
<td>80</td>
<td>102723770</td>
</tr>
</tbody>
</table>

Four-Way Handshake

This terminates a previously established connection, between A and B, as follows:

1. A sends to B a packet with FIN = 1, which indicates “no more data from A.” This flag is used when closing a connection down the normal way. The receiving host B enters the CLOSE WAIT state and starts the process of gracefully closing the connection. Each end of the connection sends a packet with the FIN = 1. The receiver is expected to acknowledge a received FIN packet by sending a FIN = 1 packet.
2. B sends to A a packet with ACK = 1, acknowledging the FIN packet received.
3. B sends to A another packet, but now with FIN = 1.
4. A sends to B a packet with ACK = 1. No further packets are exchanged.

So, four packets are used to close a TCP connection in the normal situation. This is a teardown of two half-closes.

Closing a connection can also be done by using the RST flag set to 1, which indicates to the receiver that a reset should occur. The receiving host accepts the RST packet provided the sequence number is correct, and enters the CLOSED state and frees any resource associated with this instance of the connection. The RST packet is not acknowledged. A host H sends a connection resetting RST packet if host X requested a connection to a nonexistent port p on host H, or for whatever reason (idle for a long time, or an abnormal condition, etc.), the host H (client or the server) wishes to close the connection. Resetting is unilateral. Any new incoming packets for that connection will be dropped.

Timers

TCP depends on many timers that the host must maintain per (attempted) connection as it follows the state diagram (see Figure).

The connection establishment timer is started on receiving the first packet of the three-way handshake. A typical value of this timer is 75 s. If a time-out occurs, the connection is aborted.

A FIN-WAIT timer is started when there is a transition from the FIN-WAIT 1 state to the FIN-WAIT 2 state. The initial value of this timer is 10 min. If a packet with FIN = 1 is received, the timer is canceled. On expiration of the 10 min, the timer is restarted with a value of 75 s. The connection is dropped if no FIN packet arrives within this period.

TIME-WAIT timer is started when the connection enters the TIME-WAIT state. This is to allow all the segments in transit to be removed from the network. The value of the timer is usually set to 2 min. On expiration of the timer, the connection is terminated.

A retransmission timer is started when a segment is sent. Its value, known as RTO retransmission timeout, is dynamically computed (RFC 2988). If the timer expires before an ACK is received, the segment is resent, and the timer is restarted.
A persistence timer is used to detect if any window size updates were lost. The KEEP-ALIVE timer lets us distinguish between the silences caused because there is no data to send from that caused by a broken connection. Setting a KEEP-ALIVE timer allows TCP to periodically probe the other end. The default value of this timer is 2 hr. After the expiration of the timer, probes are sent to the remote end. The connection is dropped if the remote does not respond.

Flow and Congestion Control

The size of the sliding window is dynamically adjusted because of flow or congestion issues. Flow control prevents the sending process from overwhelming the receiving process. Each acknowledgment segment from a receiver advertises the buffer size it has available. If this size is larger than the current sliding window size, the sender can increase it. If it is smaller, the sender should decrease.

When a router begins to accumulate too many packets, it can send ICMP source quench messages to the senders of these packets. These messages should cause the rate of packet transmission to be slowed.

Congestion is a condition of significant delay caused by overload of datagrams at one or more routers. A congestion window size is dynamically computed by the sender based on network congestion. The TCP sliding window size is the minimum of the receiver window advertisement and the congestion window. When a segment loss is detected, we assume that the loss is due to congestion, and the congestion window size is reduced by half. On observing that segments are not getting lost, the congestion window size is doubled. TCP congestion control has undergone major improvements resulting in TCP variants known as TCP Vegas, FastTCP, and so on that are soon to be adopted in actual implementations.

UDP, ICMP, DNS, ARP, AND RARP

User Datagram Protocol

UDP is a connectionless transport protocol. It is a thin protocol on top of IP providing high speed but low functionality. Delivery of UDP datagrams is not guaranteed.

Nor can it detect duplicate datagrams. The UDP protocol is used mostly by application services where squeezing the best performance out of existing IP network is necessary, such as Trivial File Transfer (TFTP) and NFS and by the DNS.

The port numbers appearing in the UDP header are similar to the TCP port numbers (see Figure 6), but the OS support required by UDP ports is much simpler and less resource consuming than that of TCP ports. The source port is the port of the sending process. When not meaningful, this field is set to 0. The destination port is the UDP port on the receiving machine, whose IP address is supplied by the IP layer. Length is the number of bytes in the datagram, including the UDP header and the data. Checksum is the 16-bit one’s complement of the one’s complement sum of the UDP header, the source and destination IP addresses obtained from the IP header, and the data, padded with zero bytes at the end (if necessary) to make a multiple of 2 bytes.

Internet Control Message Protocol

ICMP (RFC 792, 1981) manages and controls the IP layer, as in reporting network errors, such as a host or entire portion of the network being unreachable or a packet being directed at a closed port, reporting network congestion, assisting in trouble-shooting, reporting time-outs, or forcing routing options. In general, much of the best effort in delivering IP datagrams is associated with ICMP. The purpose of the ICMP messages is to provide feedback and suggestions about problems, for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, or when the gateway can direct the host to send traffic on a shorter route. To avoid the infinite regress, no ICMP messages are sent about ICMP messages. Also ICMP messages are sent only about errors in handling fragment zero of fragmented datagrams.

An ICMP message is sent as the data portion of an IP datagram. These IP datagrams are treated like all other IP datagrams. Each ICMP message begins with a 1-byte ICMP type field, which determines the format of the remaining data, a one-byte code field, and a 2-byte
configured with at least one local name server if it is to name and others perhaps more mnemonic thus providing load distribution. A single IP address can be associated with multiple names, one being a domain name into its IP address. This is done by retrieving the PTR record. Some network services use this to verify the identity of the client host. The process repeats until N receives the address for the domain name. N then caches the record and returns it to the querying host.

A domain name server maintains the name space as a database. It can delegate the maintenance of any subdomain to another server. A delegated subdomain in the DNS name space is called a zone. N can respond to a query directly. Otherwise, if the name is in the local zone, the local name server N can respond to a query directly. Otherwise, N queries one of the root servers. The root server gives a referral with a list of name servers for the top-level domain of the query. N now queries a name server on this list and receives a list of name servers for the second-level domain name. The process repeats until N receives the address for the domain name. N then caches the record and returns it to the querying host.

An iterative DNS query to a name server D receives a reply with either the answer or the IP address of the next name server. If the name is in the local zone, the local name server N can respond to a query directly. Otherwise, N queries one of the root servers. The root server gives a referral with a list of name servers for the top-level domain of the query. N now queries a name server on this list and receives a list of name servers for the second-level domain name. The process repeats until N receives the address for the domain name. N then caches the record and returns it to the querying host.

A recursive DNS query to D will make D obtain the requested mapping on behalf of the querying host. If D does not have the answer, it forwards the query to the next name server in the chain, and so on until either an answer is found or all servers are queried and hence returns an error code. Because recursive look-ups take longer and need to store many records, it is more efficient to provide a recursive DNS server for LAN users and an iterative server for Internet users.

Address Resolution Protocol

ARP (RFC 826, 1982) is typically used to determine the Ethernet MAC address of a device whose IP address is unknown. This needs to be done only for outgoing IP packets, because IP datagrams must be Ethernet framed with the destination hardware address. The translation is performed with a table look-up. Reverse ARP (RARP) (RFC 903) allows a host wishing to discover its own IP address to broadcast its Ethernet address and expect a server to reply with its IP address. The ARP cache accumulates as the host continues to network (see Table 1). If the ARP cache does not have an entry for an IP address, the outgoing IP packet is queued, and an ARP request packet that effectively requests “If your IP address matches this target IP address, then please let me know what your Ethernet address is” is broadcast. Once the table is updated because of receiving a response, all the queued IP packets can now be sent.

Table 1 A Small Portion of an ARP Cache

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Ethernet Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>130.108.2.23</td>
<td>08-00-69-05-28-99</td>
</tr>
<tr>
<td>130.108.2.1</td>
<td>00-10-2f-le-e4-00</td>
</tr>
<tr>
<td>130.108.2.27</td>
<td>08-00-69-0d-99-12</td>
</tr>
<tr>
<td>130.108.2.20</td>
<td>00-69-31-cb-3d</td>
</tr>
<tr>
<td>130.108.2.10</td>
<td>00-60-cf-21-2c-4b</td>
</tr>
<tr>
<td>192.168.17.221</td>
<td>00-50-8e-51-55-56</td>
</tr>
<tr>
<td>192.168.17.112</td>
<td>00-A0-C5-F5-7C-4f</td>
</tr>
</tbody>
</table>
The entries in the table expire after a set time period in order to account for possible hardware address changes for the same IP address. This change may have happened (e.g., because of the NIC being replaced).

ARP is an OSI layer-3 protocol, but it does not use an IP header. It has its own packet format as shown in Figure 7. The ARP request packet has zeroes in the target hardware address fields. It is broadcast on the local LAN without needing to be routed. The destination host sends back an ARP reply with its hardware address so that the IP datagram can now be forwarded to it by the router. An ARP response packet has the sender/target field contents swapped as compared to the request.

**APPLICATIONS**

Nearly all network applications are based on a client/server architecture where one process, the client, requests services from a second process, the server. Typically, the client and server processes are on different machines, but they need not be.

**File Transfer Protocol, Telnet, and rlogin**

The three application protocols described in this section are all based on TCP. They send authentication information and data in the clear (i.e., unencrypted) and hence are easily compromised by network sniffers. In addition, their authentication of host is simply the IP address. Consequently, utilities based on these protocols should not be used in situations where security is a concern. The SSH described later provides near equivalent functionality at a higher level of security.

**File Transfer Protocol**

FTP (RFC 959, 1985) uses two TCP connections, one called the control connection and the other the data connection. The client can issue a number of commands on the control connection that change various settings of the FTP session. All content transfer occurs on the data connection. The FTP client opens a control connection to port 21 of the FTP server machine. This connection persists for the entire session. The format of data passed over the control connection is the same as that of telnet NVT. The GET command requests for the transfer of the contents that the server has (popularly known as downloading), and the PUT command requests the server to receive and store the contents that the client is about to send (popularly known as uploading). FTP is an insecure protocol. User name, password, and all data are transmitted in the unencrypted form.

**Passive and Active modes of FTP**

The data connection can be opened in two modes. In the active mode FTP, the server initiates a data connection as needed from its port 20 to a port whose number is supplied by the client via the PORT command. In the passive mode FTP, the server informs the client a port number higher than 1024 that the server has chosen, to which the

---

**Figure 7:** TCP state diagram.
The rlogin protocol (RFC 1282) is similar in functionality to telnet and also operates by opening a TCP connection on the rlogin server machine at port 513. It is widely used between UNIX hosts because it provides transport of more of the UNIX terminal environment semantics than does the telnet protocol and because on many UNIX hosts it can be configured not to require user entry of passwords when connections originate from trusted hosts. Rlogin is an insecure protocol. User name, password, and all data are transmitted in the unencrypted form.

### Secure Shell

SSH provides the functionality of telnet and rlogin but with greater security. There are three primary advantages in using ssh. (1) Telnet and rlogin do not authenticate the remote machine; SSH does. The SSH client maintains a database of server names and their authentication keys that the server offers the first time an SSH session is opened to the server. Subsequent SSH sessions compare the authentication key offered by the server with that stored in the client database. (2) The password that the user types as part of the login ritual is sent as clear text by telnet and rlogin; SSH sends it encrypted. (3) The data being sent and received by telnet and rlogin is also sent as clear text; SSH sends and receives it in encrypted form. The main disadvantages are the following. (1) Encryption and decryption consumes computing and elapsed time. (2) If the remote system has been legitimately reinstalled, and the installer was not careful to use the same authentication keys for the host, a false alarm may be raised. (3) SSH is susceptible to man-in-the-middle attack.

### Hypertext Transfer Protocol

HTTP (RFC 2616, 1999) is at the core of the World Wide Web. The Web browser on a user’s machine and the Web server on a machine somewhere on the Internet communicate via HTTP using TCP usually at port 80. HTTPS (RFC 2660, 1999) is a secure version of HTTP. A Web browser displays a file of marked-up text with embedded commands following the syntactic requirements of the hypertext markup language (HTML). There are several ways of invoking these commands, the most common one being the mouse click. Most of the clickable links displayed by a Web browser are the so-called links that associate a URL (universal resource locators) with a visible piece of text or graphic. URLs have the following syntax:

```
scheme://[username[:password@]]serverMachineName [/port][/path][/resource][?parm1=param1&parm2=param2]
```
A simple example of the above is http://www.cs.wright.edu/~pmateti/InternetSecurity, where the scheme was chosen to be http, the port defaults to 80, and the path given is –pmateti/InternetSecurity. A click on such a link generates a request message from the browser process to the Web server process running on the remote machine www.cs.wright.edu, whose name is obtained from the link clicked. The browser then displays the page it receives from the server.

HTTP Message Format
The request and response are created according to the HTTP message format, which happens to be a sequence of human-readable lines of text. The first line identifies the message as a request or response. The subsequent lines are known as header lines until an empty line is reached. Following the empty line are lines that constitute the “entity body.” Each header line can be divided into two parts, a left- and right-hand side, separated by a colon. The left-hand side names various parameters. The right-hand side provides their values. The request line has three components: a method (one of GET, POST, or HEAD), a URL, and the version number of HTTP (either 1.0 or 1.1) that the client understands. GET is the most common one among the methods. The POST method is used when the client sends data obtained from a user-filled HTML form. The HEAD method is used in program development. The response line also contains three components: HTTPVersion-number, a status code (such as the infamous 404), and a phrase (such as Not Found, OK, or Bad Request). The entity body in a response message is the data, such as the content of a Web page or an image, that the server sends.

Authentication and Cookies
Web servers requiring user authentication send a WWWAuthenticate: header. The Web client prompts the user for a username and password, and sends this information in each of the subsequent request messages to the server. HTTP is stateless in that the HTTP server does not act differently to request based on previous requests. Occasionally, a Web service wishes to maintain a minor amount of historical record of previous requests. Cookies RFC 29650 create a stateful session with HTTP requests and responses. The response from a server can contain a header line such as “Set-cookie: value.” The client then creates a cookie stored on the client’s storage. In subsequent requests sent to the same server, the client includes the header line “Cookie: value.”

SECURITY
The TCP/IP suite has many design weaknesses so far as security and privacy are concerned, all perhaps because of the era (1980s) when the development took place; network attacks were unknown. For example, the ICMP redirect message, intended to improve routing performance, has often been used maliciously. In this section, we summarize some of these issues from a practical perspective. Some of these are protocol design weaknesses per se, whereas the rest are defects in the software that implements the protocols. All major OS have made improvements in their implementations of the protocol stack that mitigate or disable many of the attacks described below. Of course, the attack tools also improve.

Security Exploits
This section is an overview of security attacks in the core protocols. Space and scope considerations prevent us from discussing such attacks as the recent (2004) TCP reset attack leading to severe concerns in the routing protocol BGP used in large routers, and the Shrew denial of service attack exploiting the congestion control algorithms. Numerous chapters of this volume are devoted to a discussion of different subtopics of security. See, for example, chapters 143 (Hacking Techniques in Wired Networks), 144 (Hacking Techniques in Wireless Networks), and 156 (Network Attacks).

Sniffing
Sniffing is eavesdropping on the network. A (packet) sniffer is a wire-tap program. Sniffing is the act by machine A of making copies of a network packet sent by machine B intended to be received by machine B. Such sniffing,
TCP/IP Note

strictly speaking, is not a TCP/IP problem, but it is en-
able by the near-universal choice of Ethernet, a broad-
cast media, as the physical and data link layers. Sniffing
can be used for monitoring the health of a network as well
as capturing the passwords used in telnet, rlogin, and FTP
connections. Attackers sniff the data necessary in the ex-
plots described below.

Depending on the equipment used in a LAN, a sniffer
needs to be run either on the victim machine whose traffic
is of interest or on some other host in the same subnet as
the victim. In the normal mode, an NIC captures only
those frames that match its own MAC address. In the so-
called promiscuous mode, an NIC captures all frames that
pass by it. The volume of such frames makes it a real
challenge for an attacker to either immediately process
all such frames fast or clandestinely store them for later
processing.

An attacker at large on the Internet has other tech-
niques that make it possible to install remotely a sniffer
on the victim machine.

Illegal Packets

Packets containing “unexpected” values in some of the
fields are illegal in the sense that a legitimate sender would
not have constructed them. Software in the receiver ought
to check for such illegal packets, but legacy software was
not cautious. Attackers have written special programs that
construct illegal packets and cause the receiving network
hosts to crash or hang. The so-called Ping of Death attack
of 1996 sent an ICMP echo request (ping) packet that was
larger than the maximum permissible length (2^{16}-1).

TCP segments have a number of flags that have, col-
lectively, a strong influence on how the segment is pro-
cessed. However, not all the flags can be independently
set or reset. For example, SYN FIN, SYN FIN PSH, SYN
FIN RST, and SYN FIN RST PSH are all illegal combina-
tions. Past implementations have accounted only for valid
combinations, ignoring the invalid combinations as “will
not happen.”

An IP packet should not have source address and port
equaling the destination address and port. The 1997 attack
tool called land exploited this vulnerability.

IP Fragment Attacks

A well-behaving set of IP fragments is nonoverlapping.
Malicious fragmentation involves fragments that have il-
legal fragment offsets. A fragment-offset value gives the
index position of this fragment's data in a reassembled
packet. For example, the fragments may be so crafted
that the receiving host in its attempts to reassemble calcu-
lates a negative length for the second fragment. This value
is passed to a function [such as memcp()] that copies
from/to memory, which takes the a negative number to
be an enormous unsigned (positive) number. A pair of
carefully crafted but malformed IP packets thus causes
a server to "panic" and crash. The 1997 attack tool called
teardrop exploited this vulnerability.

The RFCs require no intermediate router to reassemble
fragmented packets. Obviously the destination must re-
assemble. Many firewalls do not perform packet reassem-
bly in the interest of efficiency. These only consider the
fields of individual fragments. Attackers create artificially
fragmented packets to fool such firewalls. In a so-called
tiny fragment attack, two fragments are created where the
first one is so small that it does not even include the des-
tination port number. The second fragment contains the
remainder of the TCP header, including the port number.
A variation of this is to construct the second fragment
packet with an offset value less than the length of the
data in the first fragment so that upon packet reassem-
ble it overrides several bytes of the first fragment (e.g.,
if the first fragment was 24 bytes long, the second fragment
may claim to have an offset of 20). Upon reassembly, the
data in the second fragment overwrites the last 4 bytes of
the data from the first fragment. If these were fragments
of a TCP segment, the first fragment would contain the
TCP destination port number, which is overwritten by the
second fragment. Such techniques do not cause a crash
or hang of a targeted system but can be used to bypass
simple filtering done by some firewalls.

Fragmentation attacks are preventable. Unfortunately,
in the IP layer implementations of nearly all OS, there are
bugs and naive assumptions in the reassembly code.

IP Address Spoofing

The IP layer of the typical OS simply trusts that the source
address, as it appears in an IP packet is valid. It assumes
that the packet it received indeed was sent by the host
officially assigned that source address.

Replacing the true IP address of the sender (or, in rare
cases, the destination) with a different address is known
as IP spoofing. Because the IP layer of the OS normally
adds these IP addresses to a data packet, aspoof must
circumvent the IP layer and talk directly to the raw net-
work device. IP spoofing is used as a technique aiding
an exploit on the target machine. Note that the attacker's
machine cannot simply be assigned the IP address of an-
other host T using ifconfig or a similar configuration tool.

Other hosts, as well as T, will discover (through ARP, for
example) that there are two machines with the same IP
address. IP spoofing is an integral part of many attacks. For
example, an attacker can silence a host A by sending fur-
ther packets to B by sending a spoofed packet announcing
a window size of zero to A as though it originated from B.

TCP Sequence Number Prediction

TCP exploits are typically based on IP spoofing and se-
quence number prediction. In establishing a TCP connec-
tion, both the server and the client generate an initial se-
quency number (ISN) from which they will start counting
the packets transmitted. Host V accepts the packets from
X only when correct SEQ/ACK numbers are used.

The ISN is (should be) generated at random and should
be hard to predict. However, some implementations of
the TCP/IP protocol make it rather easy to predict this
sequence number. The attacker either sniffs the current
SEQ+ACK of the connection or can algorithmically pre-
dict them.

Closing a Connection by FIN

The attacker constructs a spoofed FIN packet. It will have
the correct SEQ numbers so that is accepted by the tar-
geted host. This host would believe the (spoofed) sender
Connection Hijacking
Suppose X and Y have a TCP connection. An attacker Z can send packets to Y spoofing the source address as X, at a time when X was silent. Y would accept these data and update ACK numbers. X may subsequently continue to send its segments using old SEQ numbers, as it is unaware of the intervention of Z. As a result, subsequent packets from X are discarded by Y. The attacker Z is now effectively impersonating X, using "correct" SEQ/ACK numbers from the perspective of Y. This results in Z hijacking the connection: host X is confused, whereas Y thinks nothing is wrong as Z sends "correctly synchronized" packets to Y. If the hijacked connection was running an interactive shell, Z can execute any arbitrary command that X could. Having accomplished his deed, a clever hijacker would bow out gracefully by monitoring the true X. He would cause the SEQ numbers of X to match the ACK numbers of Y by sending to the true X a segment that it generates of appropriate length, spoofing the sender as Y, using the ACK numbers that X would accept.

The SYN Flood
The SYN flood attack occurred in 1996. In the TCP protocol as designed, there is no limit set on the time to wait after receiving the SYN in the three-way handshake. An attacker initiates many connection requests with spoofed source addresses to the victim machine. The victim machine maintains data related to the connection being attempted in its memory. The SYN+ACK packets that the victim host sends are not replied to. Once the limit of such half-open connections is reached, the victim host will refuse further connection establishment attempts from any host until a partially opened connection in the queue is completed or times out. This effectively removes a host from the network for several seconds, making it useless at least as a stepping tool to other attacks, such as IP spoofing.

Storm Creation
There have been several attacks that generate enormous numbers of packets rendering (portions of) a network ineffective. The attackers send source spoofed packets to intermediary machines. These amplify the numbers of packets into a "storm." ACK storms are generated in the hijack technique described above. A host Y, when it receives packets from X after a hijack has ended, will find the packets of X to be out of order. TCP requires that Y must send an immediate reply with an ACK number that it expects. The same behavior is expected of X. So, X and Y send each other ACK messages that may never end.

The attack tool of 1997, called smurf, sends ICMP ping messages. There are three machines in smurfing: the attacker, the intermediary router, and the victim. The attacker sends to an intermediary an ICMP echo request packet with the IP broadcast address of the intermediary's network as the destination. The source address is spoofed by the attacker to be that of the intended victim. The intermediary puts it out on that network. Each machine on that network will send an ICMP echo reply packet to the source address. The victim is subjected to network congestion that could potentially make it unusable.

ARP Poisoning
ARP poisoning is an attack technique that corrupts the ARP cache that the OS maintains with wrong Ethernet addresses for some IP addresses. An attacker accomplishes this by sending an ARP response packet that is deliberately constructed with a "wrong" MAC address. The ARP is a stateless protocol. Thus, a machine receiving an ARP response cannot determine if the response is because of a request it sent or not. ARP poisoning enables the so-called man-in-the-middle attack that can defeat cryptographic communications such as SSH, SSL [see chapter 65: SSL/TLS (Secure Sockets Layer/Transport Layer Security)], and IPSec. An attacker on machine M inserts him- or herself between two hosts A and B by (1) poisoning A so that B's IP address is associated with M's MAC address, (2) poisoning B so that A's address is associated with M's MAC address, and (3) relaying the packets M receives A from/to B. ARP packets are not routed, and this makes it very rewarding to the attacker if a router can be ARP poisoned.

Route Spoofing
An attacker can spoof routing information by three main mechanisms.
In the first mechanism, an attacker sends an ICMP redirect packet with the source address set to the regular router. The packet also contains the "new" router to use. An ICMP route redirect is normally sent by the default router to indicate that there is a shorter route to a specific destination. A host adds a host-route entry to its routing table after some checking all of which is ineffective. Unlike ARP cache entries, host route entries do not expire. Name servers are obvious targets for this attack.
In a second way, RIP-based attacks work by broadcasting illegitimate routing information to passive RIP hosts and routers via UDP port 520. In both of the above cases, the redirection can be made to any host chosen by the attacker.
Third, source routing allows the sending host to choose a route that a packet must travel to get to its destination. Traffic coming back to that host will take the reverse route. The attacker designs a route so that the packets go through his site.

DNS Spoofing

The DNS answers that a host receives may have come from an attacker who sniffs a query and answers it with misleading data faster than the legitimate name server answers. The attacked host may in fact be a DNS server. Such DNS spoofing results in DNS cache poisoning, and all the clients of this server will receive false answers. During the reconnaissance stage of an attack, DNS zone transfers help map the targeted network. Some of these issues are specific to a software package called BIND that implements the DNS service. The DNS protocol is improved in the DNSSEC, which is expected to be deployed widely by adding authentication.

Covert Channels

Covert channels are the principle enablers in a distributed denial of service (DDoS) attack. A DDoS attacker covertly distributes (portions of) his attack tools over many machines spread across the Internet and later triggers these intermediary machines into beginning the attack and remotely coordinates the attack.

Covert channels are possible in nearly all the protocols of the TCP/IP suite. For example, ICMP echo request packets should have an 8-byte header and a 56-byte payload. ICMP echo requests should not be carrying any data. However, significantly larger ICMP packets can be generated carrying covert data in their payloads. Covert channels can be setup using the 137 field of IP packets. IP checksums, TCP initial sequence numbers, or TCP timestamps.

ICMP Exploits

The ICMP protocol is a simple protocol with one message per packet. ICMP is also one of the easiest to exploit. In addition to the exploits described above, namely smuggling, route redirection, and covert channels, it has enabled several other exploits via reconnaissance and scanning.

Finger Printing a System

An attacker wants to identify the exact version of an OS running on a targeted victim. Nuances in the TCP/IP stacks implemented in the various OS, and versions of the same OS, make it possible to remotely probe the victim host and identify the OS. Such probing deliberately constructs illegal packets, and attempts to connect to each port and observe the responses it gets. The tool called smtp is comprehensive in this regard.

Buffer Overflows in Servers

A large number of TCP/IP server programs suffer from a class of programming errors known as buffer overflows. Many of these server programs run with the privileges of a super user. Among the many servers that suffer from such bugs are several implementations of FTP servers, the ubiquitous DNS server program called bind, the popular mail server called sendmail, and the Web server IIS, to name a few. An attacker supplies cleverly constructed inputs to such programs causing them to transfer control to executable code he or she has supplied. A byte or two of executable code he supplies produces a shell that he can interact with from a remote machine with all the privileges of the super user.

Security Enhancements

A number of enhancements for TCP/IP have been made that are not yet in common use. Several of them (e.g., VPN and IP6) involve heavy use of encryption and require more computing power. As computing power in end-user hosts increases, we expect to see these universally deployed.

Several chapters of this volume are devoted to security enhancements. Virtual private networks (VPN) enable secure communication through public networks using cryptographic channels. See chapters 184 (Virtual Private Networks Basics) and 185 (VPN Architecture).

Authentication

Authentication is the process of verifying the credentials of a user, a node, or a service. Authentication protocols enable such procedures. Authentication protocols send or receive messages in encrypted form. Without encryption, it is like having a paper-thin door to a house. Some well-known authentication protocols are Kerberos, RADIUS, PAP and CHAP.

PAP [Password Authentication Protocol, (RFC 1334)] is a two-way handshake protocol designed for use with PPP. It sends the user name and password in plain text, obviously vulnerable to sniffing. CHAP [Challenge Handshake Authentication Protocol (RFC 1426)] is a three-way handshake protocol. The CHAP server sends the user client a challenge, which is a randomly generated sequence of bytes unique to this authentication session. The client encrypts the challenge using a previously issued secret key that is shared by both the client and CHAP server. The result, called a response, is then returned to the CHAP server. The CHAP server performs the same operation on the challenge it sent with the shared secret key and compares its result, the expected response, with the response received from the client. If they are the same, the client is assumed authentic.

IP6 and IPsec

The IP version 4 that currently dominates the Internet sends all its payload and headers in clear text. A determined attacker can install remote sniffers along every path that a communication from host B to host C and assemble full messages being sent at the application level. The IPsec protocol adds authentication and encryption. The IP version 6 includes IPsec and other enhancements to IP4. See chapters 183 (IPsec: Authentication Header and Encapsulating Security Payload) and 64. (IPsec: Internet Key Exchange).

TCP/IP Traffic Scrubbing

Scrubbing refers to forcing the TCP/IP traffic to obey all the rules of the RFCs. Reserved fields can be set to a random value; illegal combinations of flags are checked, and so on. Scrubbing is expected to be done not only at the originating hosts but also on the routers and especially in firewalls. Scrubbing adds to the computational burden of
the hosts. Because of hidden assumptions made by programs beyond the specifications of the RFCs, scrubbing may disrupt interoperability.

CONCLUSION

The Internet and the World Wide Web are based on a suite of protocols and software collectively known as TCP/IP. It includes not only the transmission control protocol and Internet protocol but also other protocols such as UDP, ARP, DNS, and ICMP; and applications such as telnet, FTP, Secure Shell, and Web browsers and servers. We surveyed these topics starting from the seven-layer OSI model to recent improvements in the implementations of the protocol stack and security.

GLOSSARY

Big Endian A 32-bit integer is stored in four consecutively addressed bytes $a$, $a + 1$, $a + 2$, and $a + 3$. In a big-endian system, the most significant byte of the integer is stored at $a$. In a little-endian system, the least significant byte is stored at $a$.

Byte A byte is a sequence of 8 bits. Viewed as an unsigned number, it is in the range of 0 to 255.

Checksum A checksum is a function of the sequence of bytes in a packet. It is used to detect errors that may have altered some of the numbers in the sequence. The IP checksum field is computed as the 16-bit one’s complement of the one’s complement sum of all 16-bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.

Client The process that establishes connections for the purpose of sending requests.

Connections In the connectionless communication, one process sends data to another without prior negotiation. The recipient does not acknowledge the receipt of the message, and the sender has no guarantee that the message is indeed delivered. In the connection oriented communication, there are three well-defined phases: connection establishment, data transfer, and connection release.

Datagram is a sequence of bytes that constitutes the unit of transmission in the network layer (such as IP).

Frame The unit of transmission at the data link layer, which may include a header and/or a trailer, along with some number of units of data.

Host A device capable of sending and receiving data over a network. Often, it is a computer system with an NIC, but it can be a much simpler device.

Network is a collection of links in which the hosts are connected either directly or indirectly.

Network Applications Programs that operate over a network.

Network Operating Systems These systems have network software built in and are aware of byte order issues.

Octet An 8-bit quantity on older computer architectures where the smallest addressable unit of memory was a word and not a byte.

Packet A generic term used to designate any unit of data passed between communicating entities and is usually mapped to a frame.

Process The dynamic entity that can be summarized as a “program during its execution on a computer system.”

Program is a file of binary data in a certain rigid format that is specific to each platform, capable of being both a client and a server. Our use of these terms refers only to the role being performed by the program for a particular connection rather than to the program’s capabilities in general.

Protocol A formal and preagreed set of rules that govern the communications between two or more entities. The protocol determines the meaning of specific values occurring in specific positions in the stream, the type of error checking to be used, the data compression method, how the sender will indicate that it has finished sending a message, and how the receiver will indicate that it has received a message.

RFC Request for Comments documents are Internet standards, proposed designs, and solutions published by researchers from universities and corporations soliciting feedback and archived at http://www.rfc-editor.org/.

Server A process that accepts connections to service requests by sending back responses. It is also called a daemon.

Spoofing In IP spoofing, either the source or the destination address is fake. In DNS spoofing, a query receives false response.

Tunneling The process of sending packets of a certain protocol embedded in the packets of another protocol.

CROSS REFERENCES

See Internet Security Standards; IP-Based VPN; Security in Circuit, Message, and Packet Switching; VPN Architecture; VPN Basics.

REFERENCES


TCP/IP suite


Krishnamurthy, B., & Rexford, J. (2001). Web protocols and practice: HTTP/1.1, networking protocols, caching, and traffic measurement. Reading, MA: Addison Wesley.


NOTE: RFCs are archived at http://www.rfc-editor.org/

FURTHER READING

TCP/IP details are part of many college courses on computer networks. There are several textbooks. Of these, the three authoritative volumes of Comer’s Internetworking with TCP/IP are classic technical references in the field aimed at the computer professional and the degree student. Volume I surveys TCP/IP and covers details of ARP, RARP, IP, TCP, UDP, RIP, DHCP, OSPF, and others. There are errata at http://www.cs.purdue.edu/homes/dec/tcpip1.errata.html The Internet Book: Everything You Need to Know about Computer Networking and How the Internet Works is a gentler introduction. The books listed above by Tannenbaum and Kurose and Ross are also popular textbooks. The book by Stevens discusses from a programming point of view.

Routing protocols are discussed briefly in the above books. The books by Halabi and Kurose and Ross cover this topic very extensively.

The HTTP protocol and related issues are thoroughly discussed in the books of Krishnamurthy and Rexford and Gourley and Totty.

The book by Denning and Denning is a high-level discussion of how the vulnerabilities in computer networks are affecting society. Mateti has an extensive Web site that has lab experiments and readings online. The book by Garfinkel and Spafford explores security from a practical UNIX systems view.

All the RFCs are archived at http://www.rfc-editor.org/. The Usenet newsgroup comp.protocols.tcp-ip is an active group and maintains a frequently asked questions (FAQ) document that is worth reading. The Technical Committee on Computer Communications of the IEEE Web site (http://www.comsoc.org/) maintains an extensive collection of conference listings. The IEEE/ACM Transactions on Networking is a peer-reviewed archival journal that publishes research articles.

The Web site www.cert.org issues timely and authoritative alerts regarding computer exploits and has a comprehensive collection of guides on security. The Web site http://www.phrack.org publishes detailed descriptions, often with ready-to-compile source code of vulnerabilities and tutorials.
Author Queries

Q1: Au: pls. provide fig. 1 citation?
Q2: Au: Pls check "Figure".
Q3: Au: Pls. provide citation of Fig. 8
Q4: Au: Pls. provide citation of Fig. 9
Q5: Au: Pls. check "NOTE".