TCP/IP Suite

[for *Computer Networks Handbook*]

Prabhaker Mateti, Wright State University, pmateti@wright.edu

Key Words ARP, DNS, DoD models, HTTP, Internet protocols, ICMP, IP, IPv6, OSI, security, TCP, TCP6, UDP.

The Internet and the World Wide Web are based on TCP/IP. The term *TCP/IP* not only refers to the TCP (transmission control protocol) and IP (Internet protocol) but also includes other protocols, applications, and even the network medium. These protocols include UDP, ARP, and ICMP. These applications include telnet, FTP, Secure Shell, NFS, Web browsers and servers, and the many items collectively called the Web services. This chapter is an encyclopedic survey of these topics starting from the seven-layer OSI model to recent developments that improve security and privacy such as IPv6, improvements in the implementations of the protocol stack.

1 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.

The two primary protocols in the suite are IP (Internet Protocol) and TCP (Transmission Control Protocol). IP delivers packets from machine to machine using Ethernet or other physical layers and routers. TCP, running on top of IP, reliably delivers packets from process to process. IP requires the support of the infrastructure protocols. ARP maps addresses of the physical layer to IP addresses. ICMP informs errors and controls packet traffic. UDP is a thin layer over IP serving those applications where throughput is more important, and unreliability issues are secondary. The size of the Internet is so large that routing tables must be constructed with the help of routing protocols such as RIP, OSPF, and BGP. Domain name service (DNS) brings not only mnemonics but also indirectly stabilizes the association of services to IP addresses.

TCP/IP details constitute one or more university courses on computer networks. Entire textbooks have been written on the topic that this chapter is attempting to cover. Different views exist as to what is to be included under the heading of TCP/IP suite. Our goal is to present the practical TCP/IP landscape as it is today. To limit this chapter to some 30 pages in print requires that certain protocols are treated only briefly, and several other protocols are barely mentioned.

This chapter describes the core protocols known as IP, TCP, UDP; infrastructure protocols named ARP, ICMP, and DNS; application protocols such as DHCP, SNMP, HTTP, SMTP, POP, IMAP, FTP, and SSH; and the next generation of TCP/IP.

1.1 Protocols

A computer system communicates with another system by sending messages. The Internet is a packet-switched network; messages are split up into *packets*, which are logically viewed as sequences of fields. A *field* is a grouping of certain consecutive bytes or bits from the packet. Depending on the protocol, a packet is re-termed as a frame, a datagram, or a segment.

The communication of messages is actually between a process running on one system and one running on the other system. Many of these processes are standard components of an operating system (OS); others are invoked when necessary as independent programs.

The rules of the protocol govern the communications between two or more processes, and their behavior in terms of semantics, timing, and ordering. The protocol also determines the data compression method, how the sender will indicate that it has finished sending, etc. A protocol describes not only the syntax or structure of the message, i.e., of the various fields of a packet, but also the interrelationships of the fields, the type of error checking to be used, and validity conditions.

1.2 RFCs

TCP/IP is an open system. Its protocols are written up as technical reports known as RFC (Request for Comments) documents and receive wide scrutiny. RFCs are freely downloadable public documents archived at http://www.rfc-editor.org/. In general, these are proposed designs and solutions published by researchers from universities and corporations soliciting feedback. Some of these become Internet standards. Protocols evolve through many drafts and versions. In this chapter, we cite only the latest and/or the most relevant RFC for each protocol.

RFCs depict a packet as stacked rows of adjacent rectangles, with tick marks for bits. The field names are written inside the rectangles. When the row is too long to fit in the page width, the row is folded into several 32-bit wide rows, as in Figure 1. The indices of the bits start from 00 at the left to 31 at the right. The unit digits of these bit indices are shown in the second row and the tens digits are shown in the first row. Indices of the bytes start from top-left at 0, but are not shown.

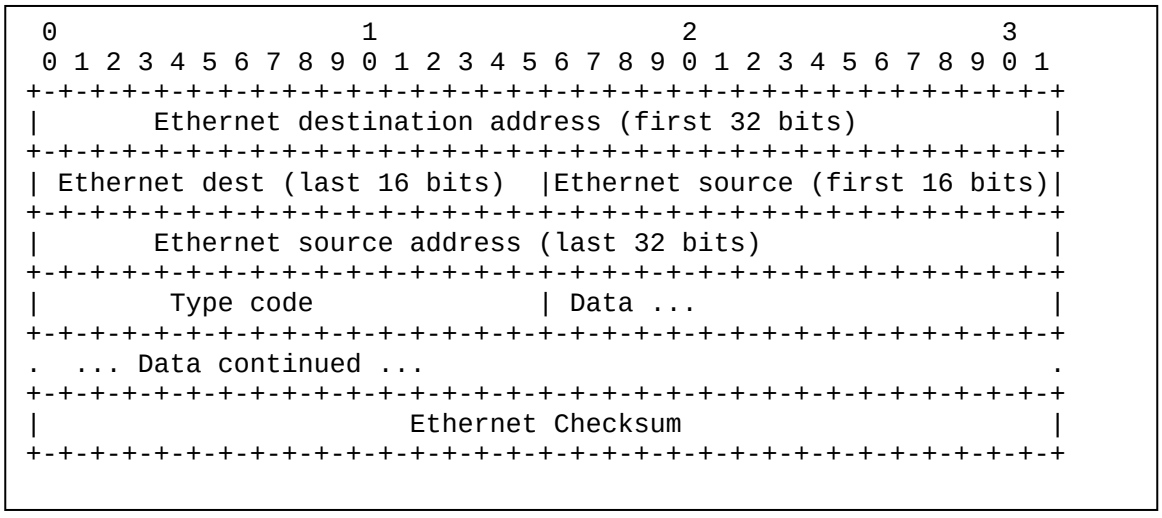

Figure 1: Ethernet frame

1.3 Design Weaknesses

The TCP/IP suite has many design weaknesses, particularly in the context of security and privacy, all perhaps because of the era (1980s) when the development took place; network attacks were then unknown.

The presence of covert channels in nearly all the protocols of the TCP/IP suite is due to lack of concern. Cover channels are methods of communicating data using fields in a surprisingly clever way unintended by the protocol design. Covert channels are the principle enablers in a distributed denial of service (DDoS) attack. An 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.

This chapter briefly describes these weaknesses as we proceed.

1.4 Implementation Issues

In general, there are multiple implementations of each protocol. They have become interoperable; this is often attributed to the fact that many of the implementations of the protocols are open source. But, their behaviors do differ in performance, resource usage, and in other details, often omitted in the RFCs, e.g., in dealing with invalid 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/firmware 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. All major OSs have made improvements in their implementations of the protocols. Security attacks have often exploited these details.

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, attempts to connect to each port, and observes the responses it gets.

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 typical "code" he inputs produces a shell that he can interact with from a remote machine with all the privileges of the super user.

This chapter does not focus on vulnerabilities, but we do hint at the specifics of these issues. There are several chapters in this Handbook that cover security issues, such as Chapter 177 Network Attacks, Chapter 176 Intrusion Detection Systems, Chapter 181

Virtual Private Networks, Chapter 182 Cryptography, Chapter 186 Computer and Network Authentication, and Chapter 187 Password Authentication.

2 LAYERS

Network protocols are 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.

The data unit of each layer is *encapsulated* by the layer below it (Figure 2), 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 by computing a TCP header TCPH and prefixing it as [TCPH [AP]]. The IP layer encapsulates it as [IPH [TCPH [AP]]], where IPH is the IP header. Ethernet encapsulates it as [EH [IPH [TCPH [AP]]] FSC], where EH is the Ethernet header and FSC is a frame check sequence generated by the Ethernet hardware.

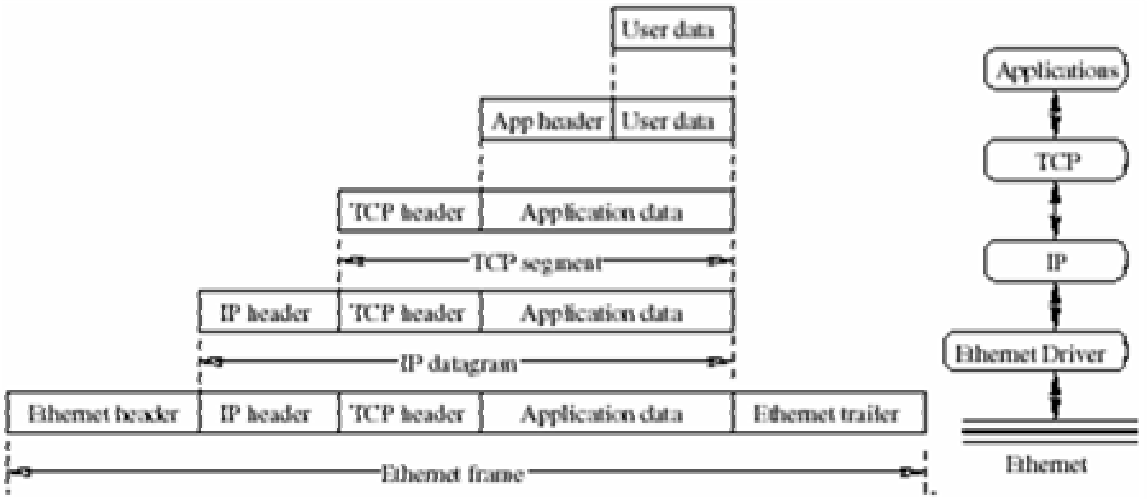

Figure 2: Encapsulations

The layers are used in both directions: receiving and sending. While sending, each layer encapsulates the data payload supplied by the above layer. While receiving, each layer strips of the headers and suffixes of the layer producing the data payload for the layer above it.

There are three models of layers: the OSI, the DoD, and the “hourglass” models.

2.1 The OSI Model

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

1. The bottom-most layer, known as the *physical layer*, provides the physical means of carrying the stream of bits. Ethernet, Fast Ethernet, Wireless 802.11, T-carrier, and DSL (digital subscriber line) are examples of this layer. All media are considered functionally equivalent. The differences are in speed, convenience, and cost. It is possible to have different physical layers in a computer network.
2. The *data link layer* structures 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. This layer describes the specification of interface cards to specific types of networks (e.g., Ethernet, and token ring). The data link layer is divided into the media access control (MAC) sub layer, which controls how a computer on the network gains access to the data and permission to transmit it, and the logical link control (LLC) sub layer, which controls frame synchronization, flow control, and error checking. Example protocols from the TCP/IP suite that occupy this layer are SLIP and PPP.
3. 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. IP, address resolution protocol (ARP), reverse ARP (RARP), Internet control message protocol (ICMP), and IGMP belong to this layer.
4. The *transport layer* transfers data and is responsible for host-to-host error recovery and flow control. TCP and UDP belong to this layer. UDP provides connectionless service and TCP provides connection-oriented service.
5. 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.
6. The *presentation layer* provides independence from possibly different data representations of the host machines. 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.
7. The *application layer* supports the end-user invoked programs. FTP, HTTP, IMAP (Internet Message Access Protocol), NTP (Network Time Protocol), POP (Post Office Protocol), rlogin (Remote Login), SMTP (Simple Mail Transfer Protocol), SNMP, SOCKS, telnet, X-Window, Web services, and so on are part of this layer.

2.2 The DoD Model

The practical world of TCP/IP networking was in full use by the time the OSI model was formulated. It is unofficial, but this model is widely used and referred to as (i) the TCP/IP model, (ii) the DoD (U.S. Department of Defense) model, or even more simply (iii) the Internet model. The DoD model organizes 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 and UDP are in this layer.
4. The *application* (or process) layer deals with user-level services, such as FTP, telnet, SSH, SSL, SMTP, and HTTP. This layer corresponds to the session, presentation and application layers of the OSI model.

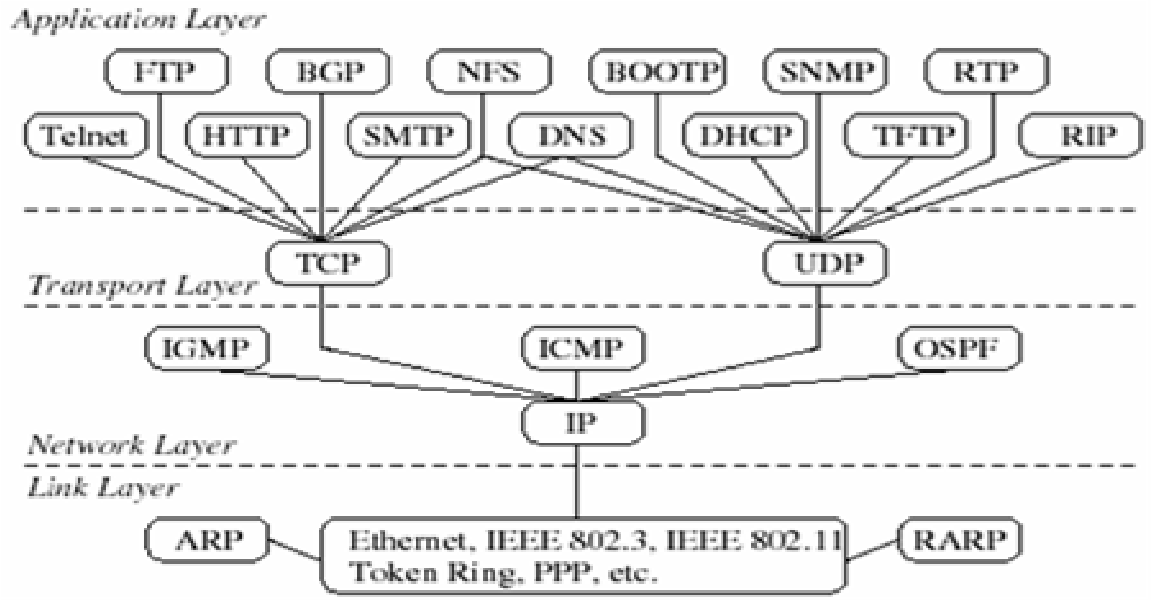

Figure 3: The DoD Layers

2.3 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.

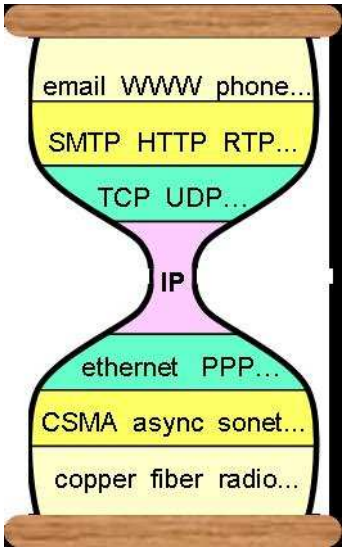

Figure 4: The hourglass model.

2.4 Lower Layers

In this section, a few prominent entries from the OSI physical, and data link layers are described.

2.4.1 Ethernet

Ethernet has been standardized by the Institute of Electrical and Electronics Engineers as IEEE 802.3.

There are variations of Ethernet, and the plain Ethernet now refers to 10-megabits-per-second (Mbps) transmission speed, Fast Ethernet refers to 100 Mbps, and Gigabit Ethernet refers to 1000 Mbps. The connecting media varieties include twisted pair (1000Base-T, 100Base-T, with RJ45 connectors), thick coaxial system (10base5), thin coaxial (10base2), and fiber optic systems (10basesF).

2.4.1.1 Ethernet Frames

Every Ethernet controller is assigned a world-wide-unique 48-bit MAC address by the factory. For communication among humans, the MAC address is written with each byte in hexadecimal, separating bytes with either a hyphen or a colon, as in 00-0A-E6-9B-27-AE. Every Ethernet frame (see Figure 1) has a 14-byte header that begins with the 6-byte MAC addresses of the destination and source followed by a 2-byte long type code. Ethernet can support IP and other protocols simultaneously. The type code identifies the protocol family (such as IP, ARP, and NetBEUI). The data field is from 46 to 1500 bytes in length. Following the data, there is a checksum computed by the Ethernet controller for the entire frame.

Every device is expected to listen for Ethernet frames containing its MAC address as the destination. All devices also listen for Ethernet frames with a wild-card destination address of FF-FF-FF-FF-FF-FF, called a broadcast address. When these packets are received by the Ethernet network interface card (NIC), it computes the checksum and throws the packet away if an error is detected by the checksum. If the type code is IP, the Ethernet device driver passes the data portion of the frame up to the IP layer of the OS.

Under OS control, the NIC can be put into a so-called promiscuous state wherein the NIC listens to all frames regardless of their destinations.

2.4.1.2 Unswitched Networks

In an unswitched network, such as when each host is connected to a hub or the obsolete thin cabled Ethernet, every NIC can see the frame being sent. 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. To send data, a host waits for the channel to become idle and then transmits its frame. If two or more devices do 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. This ensures that access to the network channel is fair and that no single host can lock out other hosts.

2.4.1.3 Switched Networks

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 station and the switch are the only devices that can access the medium. End stations and the switch can transmit at will, achieving a collision-free environment.

Unswitched Ethernet segments have all but disappeared. Switched Ethernets replace the shared medium of legacy Ethernet with a dedicated segment for each host and extend the bandwidth. Today, a typical end user connects to a full duplex switched Ethernet, 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.

A switch 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; the other ports do not see the frame.

2.4.1.4 Sniffing

Sniffing is eavesdropping on the network by a process, the sniffer, on a machine S that records copies of packets sent by machine A intended to be received by machine B. Such sniffing, strictly speaking, is not a TCP/IP problem, but it is enabled by 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.

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. In an unswitched Ethernet segment, sniffing is therefore easy. The sniffer can be run on some other host in the same subnet as the victim.

Switched Ethernets mitigate sniffing attacks. If the LAN segment is switched, a sniffer needs to be run either on the victim machine whose traffic is of interest, or on the router. However, switches can be "overwhelmed" into behaving as though they are hubs.

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

2.4.2 IEEE 802.11 a/b/g/n Wireless Networks

This section briefly describes wireless networks known as the IEEE 802.11 family. For a detailed treatment, read the chapters of Part 3: Wireless Networks of this Handbook. The 802.11a operates at a theoretical maximum speed of 54 Mbps, 802.11b at 11 Mbps, and 802.11g at 54 Mbps.

2.4.2.1 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.

2.4.2.2 Frames

A dot-11 frame consists of a MAC header (see Figure 4) followed by a frame body (payload) 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.

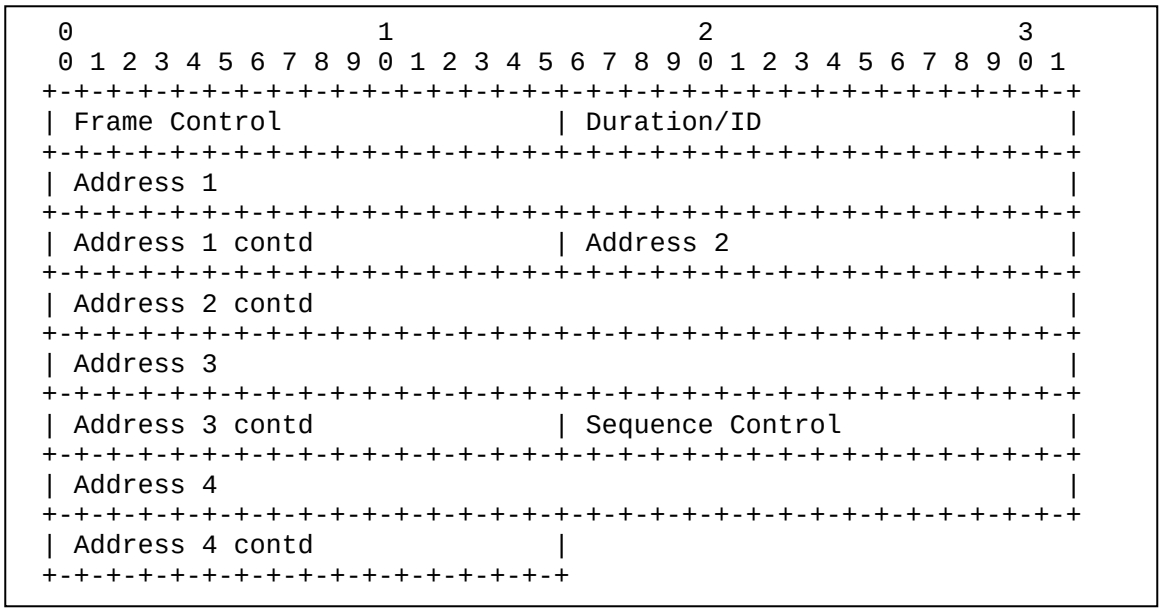

Figure 5: An IEEE 802.11 MAC header.

2.4.2.3 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.

2.4.2.4 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.

2.4.3 Asynchronous Transfer Mode

Asynchronous transfer mode (ATM) is widely deployed as a backbone technology. ATM uses fixed-length packets called *cells* for transport. Information is divided among these cells, transmitted, and then reassembled at their final destination. Each cell consists of a 48-byte payload and 5 bytes of additional information, referred to as overhead.

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.

2.4.4 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 envelopes an IP packet. Unlike Ethernet, SLIP supports only the IP 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.

2.4.5 Point-to-Point Protocol

Point-to-point protocol (PPP; RFC 1661, RFC 2153) 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 (LCP) that negotiates the encapsulation format, sizes of packets, authentication methods, and other configuration options. The CCP (compression control protocol) used by PPP negotiates encryption. The IP control protocol (IPCP) included in the PPP configures the IP address and enables the IP protocol on both ends of the point-to-point link.

3 INTERNET PROTOCOL

This section describes IP version 4 (RFC 791), which currently (2006) dominates the Internet. The next generation of TCP/IP, numbered v6, is described later.

IP delivers a sequence of bytes, called a *datagram*, from a source host S to a destination host D, even when the hosts are on different networks, geographically vastly separated. The IP layer forms an IP datagram from the byte sequence and the destination given by the upper layer during a send; the reverse of this happens during receive.

The IP layer software discovers routes that the packet can take from S to various intermediate nodes, known as *routers*, ultimately arriving at D. Thus, IP is routable. Section 7 describes routing.

Each datagram travels independently, even when S wishes to send several datagrams to D; each datagram delivery is made independently of the previous ones. The route that

each packet takes may change. The sender S, the receiver D, and intermediary routers treat a datagram independently of others. Thus, IP is connectionless.

The IP layer is designed deliberately not to concern itself with guaranteed delivery (e.g., datagrams may be lost or duplicated), but instead it is a "best effort" system. ICMP described later, aids in this effort.

3.1 IP Addresses

An OS, during boot-up, assigns a unique 32-bit number known as its IP v4 address to each NIC located in the host system. There is no rigid relationship between the MAC address and the IP address. The IP address is obtained either by looking it up in a configuration file or via dynamic host configuration protocol (DHCP). When a machine is 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.

3.1.1 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.

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, bit 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 1 always, bit 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. Class C, with a maximum of 254 host addresses, is too small, whereas class B is too large to be densely populated.

3.1.2 CIDR Nomenclature /24

The classless inter-domain routing (CIDR) model (RFC 1518; 1993) solves the problems of efficient utilization of IP address space.

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-quad 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 32-bit number whose host ID portion is all ones and whose network address equals that of X. The limited broadcast address is 255.255.255.255.

3.1.3 Public and Private Address Ranges

The public IP addresses are carefully controlled worldwide. The IANA, Internet Assigned Numbers Authority (http://www.iana.org), assigns the so-called public IP addresses to organizations and individuals upon application.

There are three blocks of the IP address space intended for private internets: (i) 10.0.0.0 to 10.255.255.255 (10/8 prefix, class A), (ii) 172.16.0.0 to 172.31.255.255 (172.16/12 prefix), and (iii) 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.

3.2 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 (see Figure 6) is given to the lower layer (e.g., the Ethernet card device driver). Except for the IP Options field, all other fields are fixed in length as shown. Minimally (i.e., without options), the IP header is 20 bytes in length. With IP options, an IP header can be as long as 60 bytes. The maximum length of an IP datagram is 65535 bytes. (IP over Ethernet limits this to 1500.)

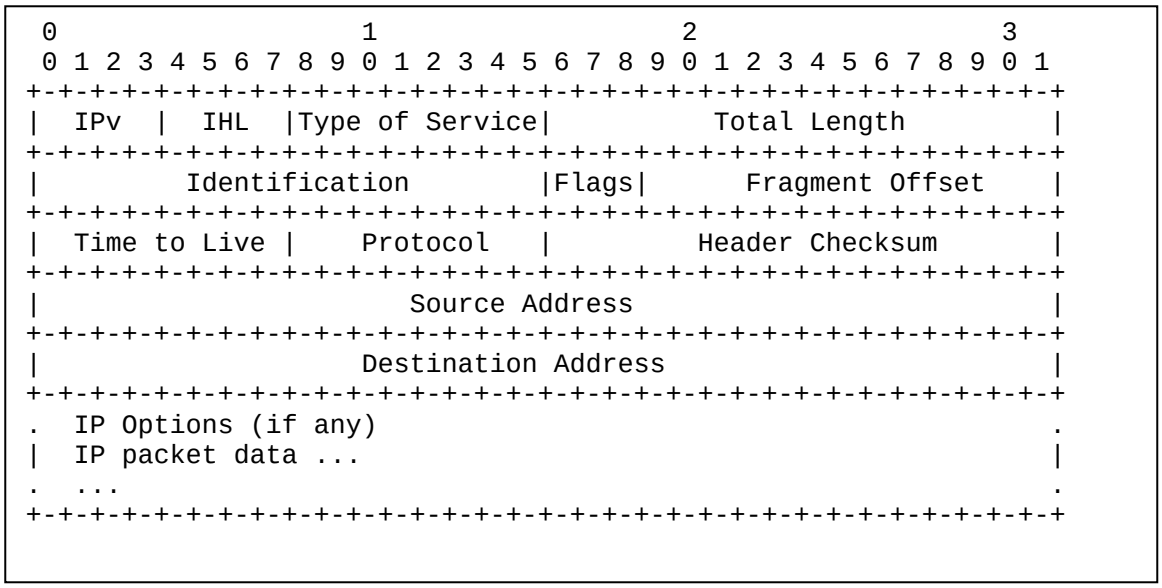

Figure 6: IPv4 Datagram

IPv is the version number of the protocol; currently it is 4. 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 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.

3.3 IP Fragments

When datagrams are too large to be sent in a single IP packet, they are 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, usually determined by interface hardware limitations. All such fragments will have the same identification field value (see Figure 6). 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 into one IP datagram.

3.4 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 agent, sends a registration request to the home address. 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.

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.

Each mobile node and home agent compute an un-forgeable 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.

3.5 IP Issues

Several aspects of IP design have become troubling issues as the mischievous elements began to use Internet.

The typical IP layer of an OS does not check for anomalies in the header. E.g., 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.

Covert channels can be setup using the ID field of IP packets, IP checksums, etc.

3.5.1 IP Spoofing

The IP layer of the typical OS simply trusts that the IP source address 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, a spoofer must circumvent the IP layer and talk directly to the raw network 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 another 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 from sending further packets to B by sending a spoofed packet announcing a window size of zero to A as though it originated from B.

3.5.2 IP Fragment Attacks

A well-behaving set of IP fragments is non-overlapping. Malicious fragmentation involves fragments that have illegal offsets. For example, the fragments may be so crafted that the receiving host in its attempts to reassemble calculates a negative length for the second fragment. This value is passed to a function (such as `memcpy`) that copies from/to memory, which takes the 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 reassemble. Many firewalls do not perform packet reassembly 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 destination 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 reassembly 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 were/are bugs and naïve assumptions in the reassembly code.

4 ICMP

Internet Control Message Protocol (RFC 792, 1981; RFC 950, 1985) is a required protocol that manages and controls the IP layer. 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. The popular network utilities `ping` and `traceroute` use ICMP.

ICMP is in the network layer. But, an ICMP message is encapsulated as an IP datagram. These are treated like any other IP datagrams.

4.1 Message Format

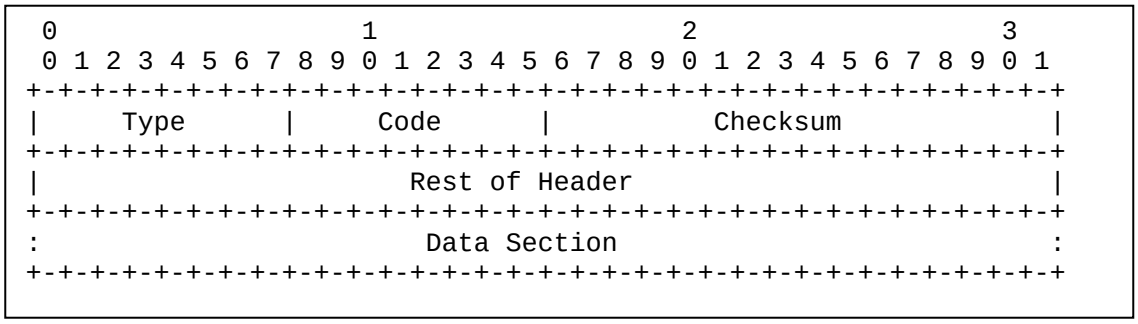

Figure 7: ICMP packet

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 checksum. ICMP messages are either error or query/reply messages.

4.2 Error Messages

ICMP is designed to report network errors, such as a host or entire portion of the network being unreachable or a packet being directed at a closed port. Error messages are reported to the original source on problems encountered by a router or the destination

host in processing an IP datagram. However, ICMP is not designed to correct errors. To avoid infinite regress, no ICMP messages are sent about ICMP messages. Also ICMP messages are not sent about errors in handling (i) other than fragment zero of fragmented datagrams, (ii) multicast address, (iii) addresses such as 127.0.0.0, or 0.0.0.0.

In an ICMP error message, the 4 bytes of Rest of Header is set to 0, and the data section is the first 8 bytes of the IP datagram causing the error.

A Destination Unreachable (Type = 3) error message is sent to the original source host when a datagram cannot reach its destination. The code field indicates the cause.

A Source Quench Message (Type = 4) is sent (i) if the gateway does not have the buffering capacity to forward a datagram, or (ii) if datagrams arrive too fast to be processed

A router R1 sends a Redirect Message (Type = 5) to a source host S if datagrams to a destination D through R1 will make the next hop to another router R2 and S can send directly to router R2.

The Time-Exceeded (Type = 11) ICMP error messages are delivered when the TTL of a datagram is 0. The `traceroute` utility constructs IP datagrams with well-chosen TTL values and collects the time exceeded messages to map a route from the source to a destination IP address.

The Parameter Problem (Type = 12) message is sent if a router finds a problem with any of the parameters in an IP header that causes it to drop the datagram.

4.3 Request/Reply Messages

The Echo Request (Type = 8) and Echo Reply (Type = 0) query messages can identify network problems. The well-known `ping` (packet Internet groper) command sends several echo requests, captures their echo replies, and displays statistics about speed and datagram loss.

The Router Advertisement (Type = 9) and Router Solicitation (Type = 10) messages are periodically multicast by a router announcing its multicast addresses. Hosts can discover the addresses of their neighboring routers simply by listening to the advertisements. When a host starts up, it may multicast a Router Solicitation to ask for immediate advertisements, rather than waiting for the next periodic ones to arrive. Note that the se router discovery messages do not indicate which router is best to reach a particular destination. A host choosing a poor first-hop router for a particular destination, it should receive an ICMP Redirect from that router.

A Timestamp Request (Type = 13) includes the timestamp, a 32-bit unsigned integer counting milliseconds since midnight UT, of the sender. A Timestamp Reply (Type = 14)

Message includes the originate timestamp and the timestamp of the receiver. These can be used to estimate the round-trip time of a datagram or to synchronize clocks.

The Address Mask Request (Type = 17) and Address Mask Reply (Type = 18) messages enable a host to discover its address mask.

4.3.1 ICMP Issues

ICMP has been one of the easiest among the protocols to exploit. For example, the ICMP redirect message, intended to improve routing performance, has often been used maliciously.

The attack tool of 1997, called *smurf*, sends ICMP ping messages. There are three hosts in smurfing: the attacker, the intermediary router, and the victim. The attacker sends to an intermediary an 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 echo reply packet to the source address. The victim is subjected to network congestion that could potentially make it unusable.

The so-called Ping of Death attack of 1996 sent an echo request packet that was larger than the maximum permissible length (2^16 - 1).

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.

In addition to the exploits described above, ICMP has enabled several other exploits via reconnaissance and scanning.

5 USER DATAGRAM PROTOCOL

UDP (RFC 768, 1980) is a connectionless transport protocol belonging to OSI layer 4. It is a thin protocol on top of IP, providing high speed but low functionality. UDP does not guarantee the delivery of datagrams. Messages can be delivered out of order, delayed, or even lost. Datagrams may get duplicated without being detected. 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), NFS, and DNS.

5.1 Port Numbers

Port numbers are used by the transport layer to multiplex communication between several pairs of processes. To each message, this layer adds addresses, called port numbers. The port numbers are assigned by the OS. IP delivers host-to-host. The UDP layer of an OS

picks up the IP datagrams, examines the UDP payload, extracts the destination port number, and delivers the UDP data to the process that registered with the destination port.

The port numbers appearing in the UDP header are similar to the TCP port numbers (see next section), but the OS support required by UDP ports is much simpler and less resource consuming than that of TCP ports.

The ports 0-1023 are reserved for specific well-known services provided by privileged processes. These numbers are assigned by IANA. These are often called *well-known* ports. For example, HTTP officially uses port 80, telnet officially uses port 23, and DNS officially uses port 53. The ports in the range 1024-49151 are called *registered ports* because they are listed by the IANA for specific services, but on most systems these ports can be used by unprivileged processes. The *dynamic* or *private* ports range from 49152 to 65535. Client processes and nonstandard services are assigned port numbers by the OS at run time. On most computer systems, there is a list of these port numbers and service names in a file named `etc/services`. Almost always, the same port assignments are used with both UDP and TCP.

5.2 User Datagrams

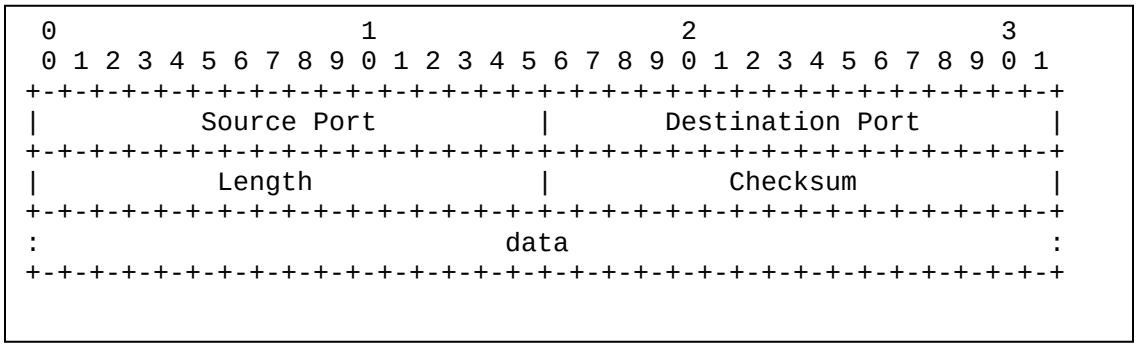

Figure 8: UDP packet.

UDP packets are called user datagrams. 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.

5.3 Connectionless Service

Each user datagram sent from a process to another is independent from others. UDP header contains no information that specifies the order of individual datagrams, or that

some were sent before. As a result, the sending and receiving processes have no way of deducing if datagrams have been lost or that they arrived out of order.

When a receiver detects an error using the checksum, there is no provision for requesting a retransmit. The datagram in error is simply discarded without informing any entity.

There is no flow control. A prolific sender process can overwhelm a receiver making its buffers overflow.

6 TCP

Transmission Control Protocol (RFC 793, RFC 3168) offers a client process a connection to a server process. 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.

6.1 Port Numbers

The port numbers are classified as in UDP. Both UDP and TCP ports co-exist and, to the extent possible, the same port assignments are used with both UDP and TCP. However, many protocols, such as HTTP (port number 80), use TCP exclusively.

6.2 TCP Header

TCP messages, called *segments*, are sent as one or more IP datagrams. A segment consists of a TCP header (Figure 9) followed by the data payload.

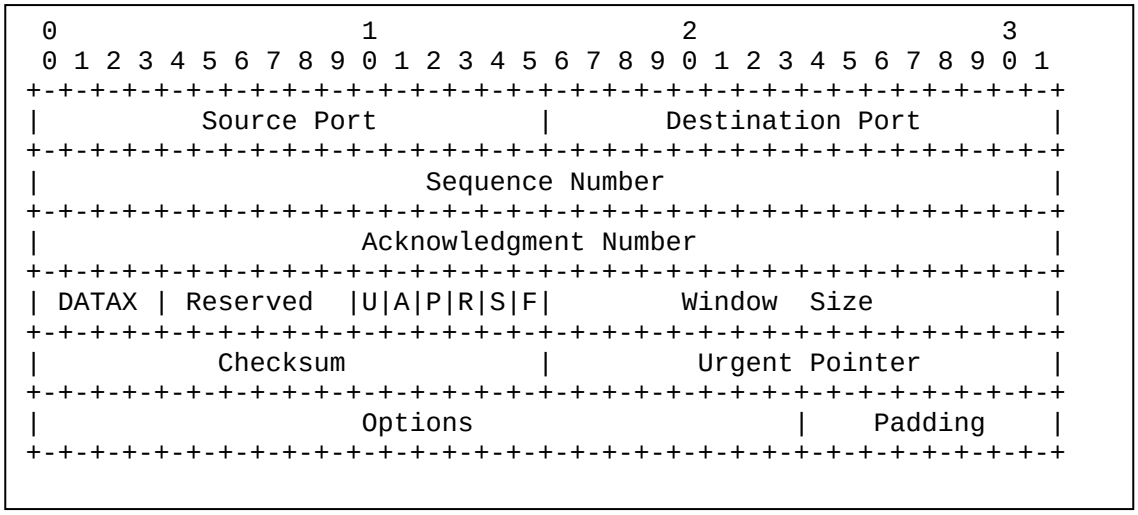

Figure 9: TCP header.

The Source Port and Destination Port are the port numbers of the sender and receiver. The sequence number and the acknowledgment number are strongly coupled together, and are explained in the next section. The 4-bit DATAX number multiplied by 4 specifies the number of bytes in the TCP header. This indicates where the data payload begins. Window size is described under Congestion Control section. Urgent pointer is valid when URG is 1. Its value is an 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.

TCP segments have a number of flags that have, collectively, a strong influence on how the segment is processed. The letters |U|A|P|R|S|F| in the fourth row of the segment are abbreviated names for control bit flags. When set to 1, their meanings are as follows: URG, urgent pointer field is significant; ACK, acknowledgment number is valid; PSH, push function is active; RST reset the connection; SYN synchronize sequence numbers; and FIN, sender is finished with this connection. 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 set to 1 are all illegal combinations. Past implementations have accounted only for valid combinations, ignoring the invalid combinations as "will not happen."

6.3 State Diagram

A TCP server process starts its life by passively opening a port and listening 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 10. We describe how to read this diagram in the next section via two important stages: one that establishes a connection and the other that closes a connection.

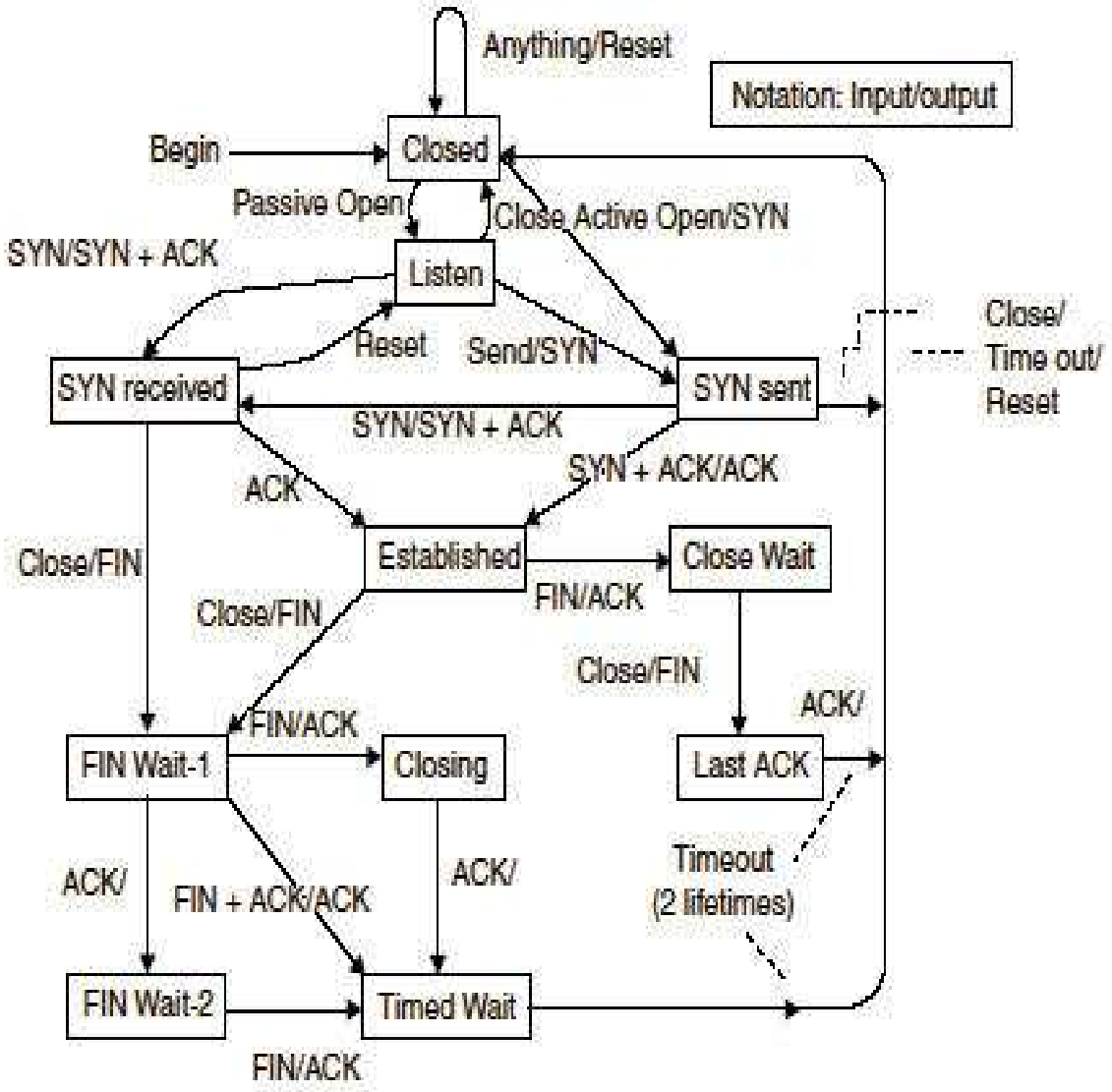

Figure 10: TCP State Diagram

6.4 Connections

A connection needs to be established as needed and kept alive between two processes, initiated by one and accepted by the other, for them to communicate. The connection keeps track of the messages between the two processes through the sequence and acknowledgment numbers.

6.4.1 Three-Way Handshake

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

1. A: "I would like to talk to you, B." A sends to B a segment with ACK = 0, SYN = 1, and the initial sequence number (ISN) chosen by A. Recall that the acknowledgment number is valid only when the ACK bit is set. This segment includes an initial window advertisement. A was in the Listen state, and now enters SYN-sent state. B was in the Listen state, and on receiving this SYN-segment enters the SYN-received state.
2. B: "OK, let's talk." B replies with a SYN¨+ACK segment (i.e., SYN = 1 with sequence number set to the ISN of B, and ACK = 1, acknowledgment number = ISN of A + 1). At this point, B remains in the SYN-received state. The condition of the connection is known as *half-open*. If B was unwilling to talk, it responds with a RST = 1 segment refusing the request for service, and moves into the Listen state.
3. A: "Thanks for agreeing!" Having received the SYN+ACK segment from B, A sends a segment with ACK = 1, acknowledgment number = ISN of B + 1, SYN = 0, sequence number = previous sequence number + 1. At this point, A is in the Established state. When B receives this segment, B also enters the Established state and the connection is fully open and functional: A and B can continue to exchange messages. In these messages, ACK = 1 and SYN = 0.

The ISN can be a zero, but that is not secure, so the ISN is randomly chosen.

Here is an example where the client is on port 1037 establishing a connection with a service on port 80.

Table 1: An Example Three-way Handshake

SYN	ACK	Source Port	Destination Port	Sequence number	Acknowledgement number
1	0	1037	80	102723769	0
1	1	80	1037	1527857206	102723770
0	1	1037	80	102723770	1527857207

6.4.2 Four-Way Handshake

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

1. A: “No more data from me.” A sends to B a segment with FIN = 1 and of course ACK = 1. The FIN flag is used when closing a connection down the normal way. Host A enters the FIN-Wait-1 state. The receiving host B enters the CLOSE WAIT state and starts the procedure of gracefully closing the connection. Each end of the connection sends a segment with the FIN = 1. The receiver is expected to acknowledge a received FIN segment by sending a FIN = 1 segment.
2. B: “I acknowledge your FIN.” B sends to A a segment with FIN = 0, ACK = 1, acknowledgment number = sequence number of A’s FIN segment + 1. On receiving the ACK segment from B, host A enters the FIN-WAIT2 state.

3. B: “No more data from me either.” B sends to host A another segment, but now with FIN = 1. B enters the Last-ACK state, and after two time-out periods, enters the CLOSED state.
4. A: “I acknowledge your FIN too.” On receiving the FIN segment from B, host A sends to B a segment with FIN = 0 and ACK = 1. Host A enters the Timed-Wait state, and after two time-out periods, enters the CLOSED state.
5. Further segments cannot be exchanged.

So, four packets are used to close a TCP connection in the normal situation. This is a teardown of two *half-closes*. From the state diagram, it can be seen that the FIN and ACK packets from B can arrive in the reverse order to the above or even FIN+ACK together in one packet.

6.5 Timers

TCP depends on many timers that the host must maintain per (attempted) connection as it follows the state diagram (see Figure 10).

The connection establishment timer is started on receiving the first packet of the three-way handshake. A typical value of this timer is 75 seconds. 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 minutes. If a packet with FIN = 1 is received, the timer is canceled. On expiration of the 10 minutes, the timer is restarted with a value of 75 seconds. The connection is dropped if no FIN packet arrives within this period.

TIMED-WAIT timer is started when the connection enters the TIMED-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 minutes. 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 hours. After the expiration of the timer, probes are sent to the remote end. The connection is dropped if the remote does not respond.

6.6 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 S does not receive an acknowledgment from the destination D within the time-out period, it retransmits the segment. The time out period is continually adapted by TCP per connection by measuring the round trip delay. Acknowledgments are piggybacked on reply data. The window size (see Figure 9) 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. The sequence number, together with the acknowledgment number, serves as a ruler for the sliding window protocol.

6.7 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 smaller 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.

6.8 TCP Issues

TCP has been in long use. Many of its implementations made improvements that go beyond the prescriptions of its RFC. Nevertheless, there are several old issues with no clear solutions and new vulnerabilities are being discovered. Covert channels can be setup using the TCP initial sequence numbers, TCP timestamps, etc. The 2004 TCP reset attack leading to severe concerns in the routing protocol BGP used in large routers, and the 2003 Shrew denial of service attack exploiting the congestion control algorithms are example attacks that exploit omissions of important details in the design.

This section briefly describes several TCP weaknesses.

6.8.1 TCP Sequence Number Prediction

TCP exploits are typically based on IP spoofing and sequence number prediction. In establishing a TCP connection, both the server and the client generate an initial sequence number (ISN) from which they will start counting the segments transmitted. Host Y accepts the segments 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 predict them.

6.8.2 Connection Closing

The 4-way handshake described above is not the only way to closing a connection.

Connections can be closed by sending a TCP segment with its 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 segment is not acknowledged. A host H sends a connection resetting RST segment 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 sever) wishes to close the connection. Resetting is unilateral. Any new incoming segments for that connection will be dropped.

Connections can be closed by FIN also. The attacker constructs a spoofed FIN segment. It will have the correct SEQ numbers so that it is accepted by the targeted host. This host would believe the (spoofed) sender had no data left. Any segments that may follow from the legitimate sender would be ignored as bogus. The rest of the four-way handshake is also supplied by the attacker.

6.8.3 Connection Hijacking

Suppose X has initiated a TCP connection to Y. An attacker Z can take over this connection. Z can send segments 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 segments from X are discarded by Y. The attacker Z is now effectively impersonating X, using "correct" SEQ/ACK numbers from the perspective of Y. As a result, Z has hijacked the connection: host X is confused, whereas Y thinks nothing is wrong as Z sends "correctly synchronized" segments 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 X a segment that it generates of appropriate length, spoofing the sender as Y, using the ACK numbers that X would accept.

6.8.4 Floods and Storms

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."

The SYN flood attack first 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 segments 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 useful at least as a stepping tool to other attacks, such as IP spoofing.

ACK storms are generated in the hijack technique described above. A host Y, when it receives segments from X after a hijack has ended, will find the segments 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.

Legitimate applications or OS services can generate a storm of packets. On many systems, the standard services known as `chargen` that listens typically at port 19 and `echo` that listens typically at port 7 are enabled. `Chargen` sends an unending stream of characters intended to be used as test data for terminals. The `echo` service just echoes what it receives. It is intended to be used for testing reachability, identifying routing problems, and so on. An attacker sends a packet to the port 19 with the source address spoofed to a broadcast address, and the source port spoofed to 7. The `chargen` stream is sent to the broadcast address and hence reaching many machines on port 7. Each of these machines will echo back to the victim's port 19. This ping-pong action generates a storm of packets.

6.8.5 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. Unfortunately, scrubbing may disrupt interoperability because of hidden assumptions made by programs beyond the specifications of the RFCs.

7 Routing

When the source S and the destination D are on the same network, we have direct delivery of an IP datagram that does not involve routing. When the two hosts are not on the same network, there can be multiple paths, in general, between S and D. Because of failures, maintenance, and other reasons, the intermediate nodes, known as routers, may come on or off during the delivery of packets. Note that consecutive packets sent by S to D may have to travel entirely disjoint routes depending on how the network is connected at the moment.

Routing algorithms and protocols discover routes that the datagram can take from S to various routers ultimately arriving at D.

7.1 Routers

Routers are specialized computer systems whose primary function (often their sole function) is to route network traffic. The typical network host has only one NIC and hence is on only one network and sending and receiving of network traffic not intended for it is secondary to its main functionality. Routers have multiple NICs, each on a separate network. A router examines the destination IP address of a packet, consults its routing tables, and sends the packet on the network connected to the final destination or the next router.

Routers are OSI layer 3 devices. Note that a router may run several routing protocols simultaneously.

7.2 Routing Tables

Network hosts (including routers) have routing tables that record information regarding where to deliver a packet next so that definite progress is made in moving the packet closer to its final destination. It can be visualized as a table of just two columns: To send the packet to a destination given in column 1, send the packet to the *next hop* whose IP address is given in column 2. Such a table would have to list all possible destinations. Table 2 shows the actual routing table of a "simple" host that has three network cards named `eth0 ... eth2`.

Table 2: Routing Table of a Simple Host with three NICs

Mask	Destination	Gateway	Flags	Metric	Ref	Use	NIC
255.255.255.0	192.168.17.0	0.0.0.0	U	0	0	0	eth0
255.255.255.0	130.102.12.0	0.0.0.0	U	0	0	0	eth1
0.0.0.0	0.0.0.0	192.168.17.111	UG	0	0	0	eth2

Given the destination IP address `d` of a datagram, the routing module of the IP layer scans the rows of the table as follows. A row numbered `r` is said to *match* if the Destination[r] value of that row equals `d & Mask[r]`, the result of `d` bit-wise-AND-ed with the mask in that row; and, the *next-hop* n is Gateway[r]. If n is zero, the datagram is delivered direct; d and the source address are on the same network. If n is non-zero, the datagram is delivered to the IP address n. The last row has both mask and destination set to zero. This catch-all default row indicates the next hop IP address for any packet whose destination address does not match 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 host.

The Flags (see Table 2) indicate whether the device of the row is Up or not, Gateway or not, etc. The metric column shows a cost number for datagrams going through that row. The metric can quantify delay, throughput, etc.

7.3 Routing Protocols

The routing tables of real routers are considerably more complex than the above. For example, they are structured to permit the use of longest CIDR prefix match which can (i) override routes to large networks with more specific host or network routes, and (ii) aggregate routes to individual destinations into larger network addresses.

The routing table of an ordinary host is tiny and rarely changes from boot-up to shut down. The tables of routers on the Internet, however, are large (tens of thousands of rows) and must be dynamically adjustable to changing conditions, perhaps by the millisecond, of the Internet. Routing protocols keep the routing tables up-to-date. The structure and content of the routing tables depend on the protocol.

Interior gateway protocols (IGP) maintain the routing tables within an *autonomous* system, i.e., a network within the control of a corporation, university, or an ISP. Exterior gateway protocols (EGP) maintain the routing information among autonomous networks.

Globally consistent routing information is achieved by routing *arbiter* database of reachability information that is replicated by route servers at network access points where autonomous systems connect.

7.3.1 Interior Gateway Protocols: RIP and OSPF

Routing information protocol (RIP) and open shortest path first (OSPF) are examples of IGPs.

RIP (RFC 2453, 1998) is based on the *distance vector* routing method which represents the inter-network as a graph G of nodes and edges, and computes shortest paths from the source node to other nodes. Hop count is the number of edges on the shortest path to a destination. A router that uses RIP maintains the routing table with rows for each

destination network it has encountered along with the hop count obtained from G. RIP periodically propagates, as unsolicited responses, the contents of its table to neighboring routers. RIP-based routers receive these responses and update their own tables.

RIP messages are UDP transported.

RIP is easy to implement but does not scale well. Route changes may result in tables not stabilizing, even forming routing loops, and some routers may have incorrect rows at any given time. Also, the number of RIP messages is proportional to the number of networks.

OSPF (RFC 2328, 1998) is based on *link states*. Two routers are said to be *linked* if they can communicate directly. The link is *up* if both are alive and reachable; otherwise the link is *down*. An OSPF-based router periodically broadcasts the status of all its links. Link status messages propagate unchanged. OSPF-based routers listen to these messages, and re-compute shortest paths whenever there is a link status change.

OSPF messages are encapsulated in IP.

Each OSPF-router computes the status of links independently of other routers. Routing table updates will converge. The number of messages depends only on the number of links of a router, not on the number of networks. Thus, OSPF scales better than RIP.

7.3.2 Exterior Gateway Protocol: BGP

BGP4 (RFC 4271, 2006), Border Gateway Protocol version 4, is the most widely deployed of the exterior gateway protocols. Each autonomous system (AS) designates one router, typically at the border, on its behalf. BGP maintains routing information among these routers. BGP runs in two modes: Exterior BGP is run between different AS, and Interior BGP is run between BGP routers in the same AS.

BGP messages are TCP transported.

BGP uses ideas from both distance vector and link status methods and extends them into path vectors described below. BGP routers collect routing information from other BGP routers, and from within their own autonomous systems through RIP and/or OSPF, and pass the combined information to their BGP router peers.

BGP has four message types. An OPEN message establishes a connection to a BGP peer, which also authenticates the sender. UPDATE messages communicate reachability information. KEEP-ALIVE messages actively test peer connectivity. NOTIFICATION messages both control and report on errors.

A BGP UPDATE message contains withdrawals of destinations that are now unreachable, and *path attributes* to newly reachable destinations. The path attributes can specify seven types of information: (i) the origin of this information, (ii) list of autonomous systems that the path goes through, (iii) the next hop, (iv) Multi-exit

discriminator (MED) which suggests a preferred route into the AS that is advertising, (v) a local preference number that indicates preference for an exit point from the local AS that has many exits, (vi) ATOMIC_AGGREGATE, and (vii) AGGREGATOR. To a newly acquired peer, full information is sent once, and later update messages carry only the incremental changes.

A BGP router R has conceptually three routing information tables/bases (RIB). All information received in UPDATE messages is entered into RIB-IN table. RIB-LOCAL holds the routes that R has selected for use by R's own AS. R analyzes the data of RIB-IN in the context of policies set by its administrator and may or may not update RIB-LOCAL. Among the multiple advertisements for the same route that it receives, BGP selects one path as the best route, inserts it into RIB-LOCAL, and propagates the path to its neighbors. Contents of RIB-OUT are extracted from RIB-LOCAL to be disseminated to peers.

The best route is chosen based on path attributes; delay, bandwidth or latency does not matter. A typical selection is as follows. If the path specifies a next hop that is inaccessible, drop the update. Select the path with the largest local preference. If the local preferences are the same, prefer the path that was originated by BGP running on this router. If no route was originated, prefer the route that has the shortest AS path. If all paths have the same AS path length, prefer the path with the lowest origin type (where IGP < BGP < "incomplete"). If the origin codes are the same, prefer the path with the lowest MED. If the paths have the same MED, prefer the external path over the internal path. If the paths are still the same, prefer the path through the closest IGP neighbor. As a final tie breaker, select the path with the lowest IP address, as specified by the BGP router ID.

IGP may converge faster, but an IGP does not scale up. Also, through the path attributes BGP has inherent support for routing policies.

7.4 Route Spoofing

Several routing spoofs have become known.

An attacker sends an ICMP redirect packet (see Section 4.2 above) 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 ineffective. Unlike ARP cache entries, host route entries do not expire. Name servers are obvious targets for this attack.

RIP-based attacks work by broadcasting illegitimate routing information to passive RIP hosts and routers. In both of the above cases, the redirection can be made to any host chosen by the attacker.

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.

A compromised BGP router can modify, drop, or introduce fake updates. Also, there are several assumptions underlying the UPDATE messages, such as (i) each AS along the path is authorized by the preceding AS to advertise the prefixes, (ii) the first AS in the path is authorized to advertise the prefixes by the holder of the prefixes, and (iii) a route is withdrawn only by the neighbor AS that advertised it, that may not be valid making BGP vulnerable to many forms of attack.

8 INFRASTRUCTURE PROTOCOLS

In this section, DNS, ARP and RARP protocols are described. The primary purpose of these protocols is to support other protocols.

8.1 Domain Name Service (DNS)

Because of the mnemonic value, humans prefer to work with host names such as `gamma.cs.wright.edu`, rather than its IP address 130.108.2.22, where `gamma` is the name of the host, and `cs.wright.edu` is the name of the domain the host is in. The primary function of DNS is to map such a name into its IP address.

The DNS name space is a tree hierarchy (RFC 1035). The top-most subtrees are the top level domains such as `.com`, `.edu`, `.net`, and `.org`, and the country code domains such as `.us` and `.in`. Subtrees of these are known as sub-domains. The leaves are the individual hosts. A *fully qualified domain name* is the sequence of labels, separated by a dot, on the path from a node to the root of the tree.

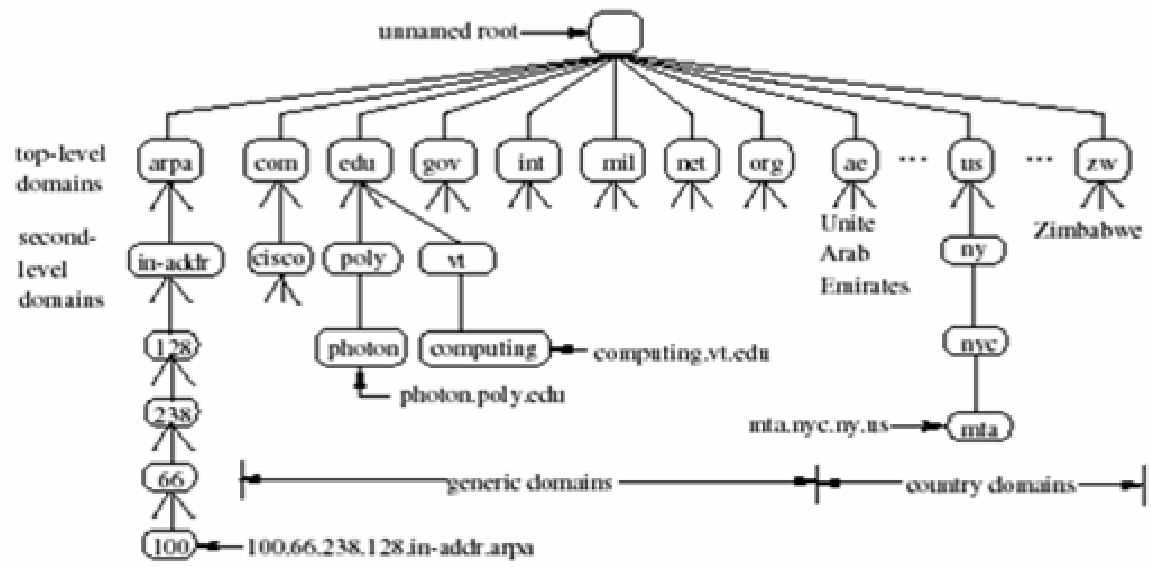

Figure 11: Domain Name Hierarchy

The top-level domains (2006) are: `ac, ad, ae, aero, af, ag, ai, al, am, an, ao, aq, ar, arpa, as, at, au, aw, az, ba, bb, bd be, bf, bg, bh, bi, biz, bj, bm, bn, bo, br, bs, bt, bv, bw, by, bz, ca, cat, cc, cd, cf, cg ch, ci, ck, cl, cm, cn, co, com, coop, cr, cu, cv, cx, cy, cz, de, dj, dk, dm, do, dz, ec edu, ee, eg, er, es, et, eu, fi, fj, fk, fm, fo, fr, ga, gb, gd, ge, gf, gg, gh, gi, gl, gm gn, gov, gp, gq, gr, gs, gt, gu, gw, gy, hk, hm, hn, hr, ht, hu, id, ie, il, im, in, info int, io, iq, ir, is, it, je, jm, jo, jobs, jp, ke, kg, kh, ki, km, kn, kr, kw, ky, kz, la lb, lc, li, lk, lr, ls, lt, lu, lv, ly, ma, mc, md, mg, mh, mil, mk, ml, mm, mn, mo, mobi mp, mq, mr, ms, mt, mu, museum, mv, mw, mx, my, mz, na, name, nc, ne, net, nf, ng, ni, nl no, np, nr, nu, nz, om, org, pa, pe, pf, pg, ph, pk, pl, pm, pn, pr, pro, ps, pt, pw, py, qa re, ro, ru, rw, sa, sb, sc, sd, se, sg, sh, si, sj, sk, sl, sm, sn, so, sr, st, su, sv, sy sz, tc, td, tf, tg, th, tj, tk, tl, tm, tn, to, tp, tr, travel, tt, tv, tw, tz, ua, ug, uk um, us, uy, uz, va, vc, ve, vg, vi, vn, vu, wf, ws, ye, yt, yu, za, zm, zw.`

Of these, the two-letter names are country domains. The top-level domain `arpa` is an *inverse domain* used to map an IP address to its name.

The mapping of names to addresses is not one-to-one. It is possible to associate a name with multiple IP addresses thus providing load distribution. A single IP address can be associated with multiple names, one being a *canonical* name and others perhaps more mnemonic thus providing host aliasing.

8.1.1 DNS Servers

The domain name space is maintained as a database distributed over several *domain name servers*. A server can delegate the maintenance of any sub-domain to another server. A delegated sub-domain in the DNS is called a *zone*. The parent server keeps track of such delegations. Each name server has authoritative information about one or more zones. It may also have cached, but non-authoritative, data about other parts of the database. A name server marks its responses to queries as authoritative or not.

A server whose zone is the entire tree is known as a *root server*. There are 13 root servers (in 2006; visit http://www.root-servers.org) located in the United States and other countries.

8.1.2 Resource Records

The database is a collection of *resource records* (Figure 12), each of which contains a domain name and four attributes: (i) The *record type* identifies what is stored as data value. (ii) The *class* attribute of the record is "IN" for Internet. (iii) The time-to-live (TTL) value indicates how long a non-authoritative name server can cache the record.

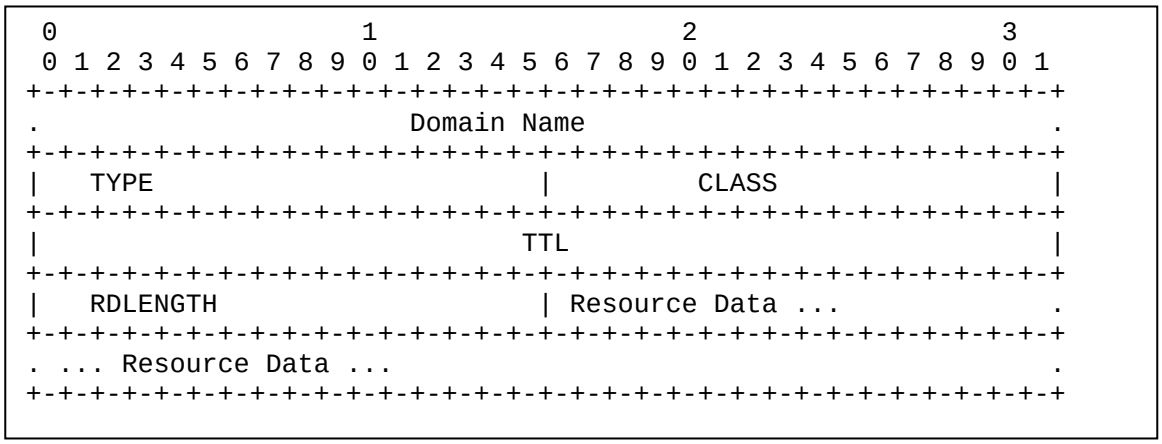

Figure 12: Resource Record

Some record types are: A, AAAA, MX, and PTR. The data of an A record type is an IP address, of an AAAA record is an IPv6 address, of an MX record is the canonical host name of the mail server and its IP address, of a PTR record is a pointer used to map an IP address to a domain name.

8.1.3 Name Resolution

The DNS *resolver* is a piece of software. Every host is configured with at least one local name server N if it is to find hosts not listed in the `etc/hosts` file. Each host maintains

a short cache table that maps fully qualified domain names to IP addresses. When a name is not found in either this file or this cache, the host enquires with N via the DNS protocol using the resolver. Either TCP or UDP can be used for DNS, connecting to server port 53. The protocol is stateless—all the information needed is contained in a single message.

The primary function of DNS is to answer a query to translate a fully qualified domain name into its IP address. This is done by retrieving the A record. A reverse look-up (also called an inverse query) is to find the host name given the IP address. This is done by retrieving the PTR record. Some network services use this to verify the identity of the client 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 the address or other DNS data 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.

8.1.4 Security of DNS

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.

DNS zone transfers help map the targeted network during the reconnaissance stage of an attack.

The DNS protocol is improved in the DNSSEC (DNS Security Extensions), which is expected to be deployed widely. DNSSEC provides (i) origin authentication of DNS data, (ii) data integrity, and (iii) authenticated denial of existence. The answers in DNSSEC are digitally signed. Note that DNSSEC does not provide confidentiality of data.

8.2 Address Resolution Protocol (ARP)

ARP (RFC 826, 1982) is typically used to determine the Ethernet MAC address of a device whose IP address is known. 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 to discover its own IP address by broadcasting the Ethernet address and expecting a server to reply with the IP address.

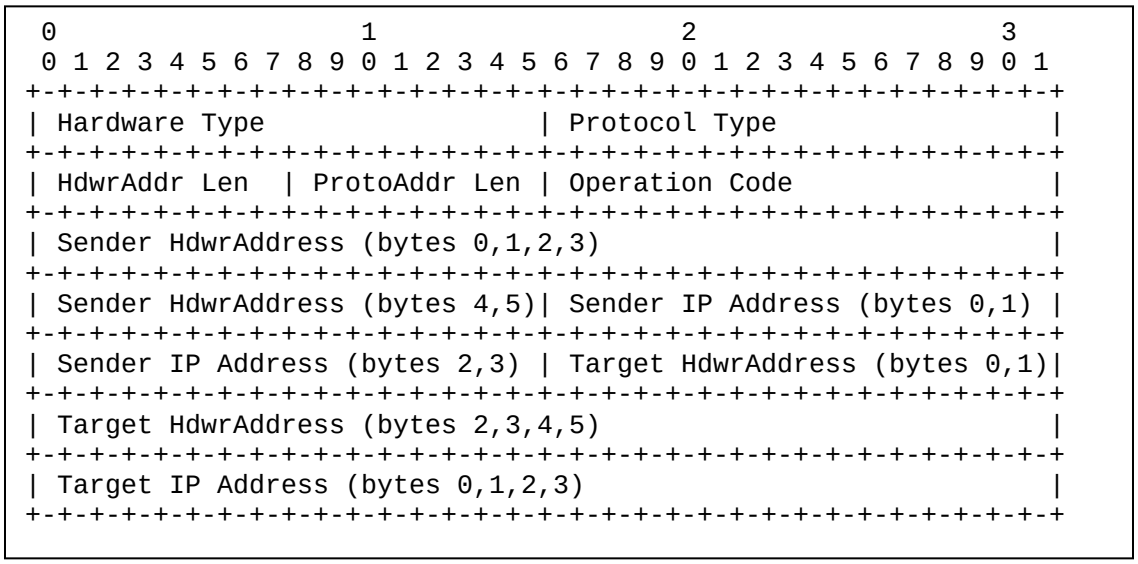

Figure 13: An ARP request/response packet

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 13. 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.

8.2.1 ARP Cache

An OS maintains a table called the ARP Cache (see Table 3). The cache accumulates as the host continues to network. 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 your Ethernet address” is broadcast. Once the table is updated because of receiving a response, all the queued IP packets can be sent.

Table 3: A Small Portion of an ARP Cache

IP address	Ethernet address

130.108.2.23	08-00-69-05-28-99
130.108.2.1	00-10-2f-fe-c4-00
130.108.2.27	08-00-69-0d-99-12
130.108.2.20	08-00-69-11-cf-b9
130.108.2.10	00-60-cf-21-2c-4b
192.168.17.221	00-50-ba-5f-85-56
192.168.17.112	00-A0-C5-E5-7C-6E

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.

8.2.2 ARP Poisoning

ARP poisoning is an attack technique that corrupts the ARP cache 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, 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.

9 APPLICATIONS

There are numerous applications based on TCP/IP. In this section, we describe briefly several protocols and applications that belong in the OSI layer 7. Several chapters in this Handbook are devoted to applications, e.g., Chapter 18 Streaming Multimedia, Chapter 181 Virtual Private Networks, Chapter 186 Computer and Network Authentication, Chapter 187 Password Authentication, Chapter 194 Computer Network Management, and Chapter 195 E-mail and Instant Messaging.

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.

9.1 Dynamic Host Configuration Protocol

A typical host will invoke a DHCP (RFC 2131, 1997) client program soon after booting into the OS to configure its network. A DHCP server delivers host-specific configuration parameters, such as an IP address, a subnet mask, a list of default routers, TTL, and MTU. An OS utility then associates the IP address with a host name.

DHCP assumes that the IP layer software will pass the packets delivered to the NIC of the host even though it has not been assigned an IP address yet. A DHCP client broadcasts (i.e., the IP destination address is 255.255.255.255) a request in a UDP packet containing its own MAC address. A DHCP server process listens to such requests, and IP-broadcasts or hardware-unicasts a reply that contains the configuration parameters.

DHCP has three mechanisms for IP address allocation. In automatic allocation, DHCP assigns a permanent IP address to a client. In dynamic allocation, DHCP leases an IP address to a client for a limited period (or until the client explicitly relinquishes the address). In manual allocation, the IP address is assigned manually but is conveyed to the client via DHCP. Dynamic allocation is the only one of the three mechanisms that allows automatic reuse by a different host of an address that is no longer needed by the client to which it was assigned.

9.2 Simple Network Management Protocol

SNMP (RFC 3411) allows network managers to see how the network is functioning and control it. A management agent (MA) runs as a server process on each managed device. A network administrator invokes a management client (MC) program on his console. The MC can send queries as IP datagrams to the MA obtaining status information, or commands to alter the conditions in the managed device. Simple network devices that do not run an agent are known as unmanaged devices.

All SNMP operations can be cast as a sequence of one or more of (i) retrieve the current value of a management information base (MIB) variable, or (ii) store a value for a variable. E.g., an immediate reboot of a managed device can be caused by storing a zero in the MIB variable that gives the time until the next reboot.

9.3 Authentication protocols

Authentication is the process of verifying the credentials of a user, a host (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.

9.3.1 PAP and CHAP

PPP (see Section 2.4.5) uses PAP or CHAP for authentication. PAP [Password Authentication Protocol, (RFC 1334)] is a two-way handshake protocol. It sends the user

name and password in plain text, obviously vulnerable to sniffing. CHAP [Challenge Handshake Authentication Protocol (RFC 1944)] 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 results, the expected response, with the response received from the client. If they are the same, the client is assumed authentic.

9.3.2 RADIUS

Remote Authentication Dial in User Service (RFC 2865) is a widely-used protocol that provides authentication, authorization and accounting services for managing access to resources, in particular wireless or dial-up Internet access. The protocol itself is quite simple, and has been implemented many times. A RADIUS client running on a network access server machine collects the users name and password and sends them as an access request to a server. The RADIUS server has a database of user names associated with their passwords through which it validates the user. The client and the RADIUS server have a shared secret that is used to encrypt portions of their messages.

9.4 Virtual Private Networks (VPN)

Virtual private networks (VPN) enable secure communication through public networks using cryptographic channels.

Point-to-point tunneling protocol (PPTP) is used in providing virtual private networks (VPN). PPTP encapsulates PPP packets. After the initial PPP connection to a PPTP server, a PPTP tunnel and a PPTP control connection are created. Tunneling is the process of sending packets of a certain protocol embedded in the packets of another protocol. PPTP uses an enhanced generic routing encapsulation (GRE) mechanism to provide a flow- and congestion-controlled encapsulated datagram service for carrying PPP packets.

See Chapter 181 for details on Virtual Private Networks.

9.5 SSL/TLS

Transport Layer Security (RFC 2246, 1999) provides privacy and data integrity. Privacy refers to a third party being unable to get unencrypted versions of messages between two parties. Integrity refers to the receiver being able to rely that the messages have not been tampered with in transit. TLS was referred to as secure socket layer (SSL) before becoming a standard.

TLS is implemented as a layer above the TCP. Higher level protocols can layer on top of the TLS transparently. Programs that used TCP can be readily re-written to use TLS instead. HTTPS and modern implementations of such protocols as FTP, Telnet, POP3S and SMTP are based on TLS. The TLS standard, however, does not specify how other protocols add security with TLS; the decisions on how to initiate TLS handshaking and how to interpret the authentication certificates exchanged are left up to the judgment of the designers and implementers of protocols which run on top of TLS.

TLS is composed of Record and Handshake protocols. The Record Protocol provides connections that are private by encrypting data using symmetric cryptography. The encryption keys are generated for each connection based on a secret negotiated by TLS Handshake Protocol. The connection is tamper-proof. Messages are integrity checked using a keyed message authentication code based on secure hash functions such as SHA, and MD5.

The TLS Handshake Protocol provides (i) authentication of the peer's identity using public key cryptography, such as RSA, and DSS, (ii) secure negotiation of a shared secret even if an attacker is present in the middle of the connection, and (iii) reliable negotiation messages that no attacker can modify without being detected by the parties to the communication.

9.6 Network File System

NFS protocol (RFC 3530, 2003) makes the file volumes located on a file server available, using UDP or TCP, to client machines.

File volume mounting, file open, read, write, and close operations are standard system calls in OSs. The mount operation can detect an attempt at mounting a remote directory via NFS. Such a remote mount operation sends client machine information and the path name of a remote directory being requested to the NFS server process. The server returns a mount handle, assuming the file volume export permissions are positive for this client. File open, read, write, and close operations on the client machine specify path names relative to the mount point as if they are local.

9.7 File Transfer Protocol (FTP)

The primary function of FTP (RFC 959, 1985) is to place a copy of a local file on a remote machine (called PUT, popularly known as uploading) or bring a copy of a remote file (called GET, popularly known as downloading). Additionally, there are a few directory maintenance commands.

FTP uses two TCP connections, one called the *control* connection and the other the *data* connection. The FTP client opens a control connection to port 21 of the FTP server machine. On the control connection, the client can issue commands that change various settings of the FTP session. The GET command requests for the transfer of the contents

that the server has and the PUT command requests the server to receive and store the contents that the client is about to send. All content transfer occurs on the data connection. This connection persists the entire session.

The data connection can be opened in one of 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 on the control connection 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 client initiates a data connection. Passive FTP is useful in firewall setups that forbid initiation of a connection from an external host (the FTP server) to the internal network (the FTP client).

FTP is an insecure protocol. The messages between the client and the server over both the connections are in the clear text. In particular, the user name, password, and all file data are transmitted in the unencrypted form. This comment does not apply to SSL based FTP.

9.8 Telnet

Telnet belongs to a class of programs called remote shells which provide a virtual terminal to a command line shell running on a remote machine.

Telnet (RFC 854; 1983) establishes a TCP connection with a telnet server on the reserved port 23 and passes the keystrokes of the telnet client to the server and accepts the output of the server as characters to be displayed on the client. The server presents these keystrokes as input received from a pseudo terminal to the OS hosting the telnet server.

Telnet defines a network virtual terminal (NVT) format as that which permits interoperability with machines that use different characters for common operations such as terminating a line and interrupting a run-away process. The telnet client typically maps the signal-generating keys of the keyboard to invoke the corresponding control functions of the NVT. The control functions are encoded as escape sequences of 2 bytes, the IAC (255), followed by the 1-byte code of the control function. Telnet uses the URGENT DATA mechanism of TCP to send control functions so that the telnet server can respond appropriately.

Telnet is an insecure protocol. User name, password, and all data in both directions are transmitted in the unencrypted form. This comment does not apply to SSL based telnet.

9.9 rlogin

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 in both directions are transmitted in the unencrypted form.

9.10 Secure Shell

SSH (RFC 4251; 2006) provides the functionality of FTP, telnet and rlogin but with greater security. There was a version called SSH-1; the version as defined in RFC 4251 is called SSH-2.

FTP, telnet and rlogin 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.

There are three primary advantages in using `ssh`. (i) Telnet and rlogin do not authenticate the remote machine; SSH does. (ii)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. (iii) 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. (i) Encryption and decryption consumes computing and elapsed time. (ii) 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. (iii) SSH is susceptible to man-in-the-middle attack.

9.10.1 SSH Architecture

SSH-2 protocol is dependent on SSH Authentication Protocol (RFC 4252), SSH Transport Layer Protocol (RFC 4253), and SSH Connection Protocol (RFC 4254).

The transport protocol (SSH-TRANS) handles the cryptographic initial key exchange and authenticates the server, sets up encryption, and compression. The transport layer also arranges for key re-exchange during a long session. The functionality of the transport layer is comparable to TLS.

The authentication protocol (SSH-USERAUTH) authenticates the user to the SSH server process. It provides for a number of authentication methods. It runs over SSH-TRANS.

The connection protocol (SSH-CONNECT) provides interactive login sessions, remote execution of commands, forwarded TCP/IP connections, and forwarded X11 connections. SSH-CONNECT defines the concept of channels. A channel transfers data in both

directions. A single SSH connection can simultaneously host multiple channels multiplexed into a single encrypted tunnel. Some channel types are: (i) “shell” for terminal shells, `sftp`, `scp`, and execution of other commands, (ii) “direct-tcpip” for client-to-server forwarded connections, and (iii) “tcpip-forward" for server-to-client forwarded connections. SSH-CONNECT runs over SSH-TRANS and SSH-USERAUTH

9.10.1.1 Host Authentication

During the first few steps of installing an OS on a machine, an authentication key pair for the host is generated using cryptographic algorithms. The pair consists of two rather large numbers that are coupled in the sense that any message encoded by one can be decoded by the other. Care should be taken to re-use this key pair whenever an OS is re-installed. The public key of the host is widely published. The private key is safe guarded.

The SSH client program verifies that it is talking to the real server by sending a message after encrypting using the public key of the SSH server machine. The server would decrypt using the private key it should have to obtain the original message.

The public key of the SSH server machine is obtained in one of two ways. (i) The client maintains a database of SSH server names and their public 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. (ii) The host name-to-key association is certified by a trusted certification authority (CA).

9.10.1.2 User Authentication

User authentication is client-driven. Several authentication methods are in wide use. (i) Password: the user types a password that is sent to the server through an encrypted channel. (ii) Public Key supporting at least DSA or RSA key pairs. (iii) Keyboard-interactive method (RFC 4256) where the server sends one or more prompts to enter information and the client displays them and sends back responses keyed-in by the user. This is used in one-time password authentication, such as S/Key or SecurID.

9.10.1.3 Tunnels

The SSH-CONNECT protocol provides encrypted channels that can be used for setting up forwarding ("tunneling") arbitrary TCP/IP ports and X11 connections. Suppose we have an SSH session for user U going between the client at host at C and the server at host S. SSH can establish a tunnel connecting a TCP port i at C with a port j at H; that is, traffic sent to C:i will be delivered to H:j and vice versa. All traffic on this tunnel is encrypted.

9.11 Mail Protocols

Electronic mail existed “before Internet” but has become as common as the telephone due to it.

Email client programs such as MS Outlook or Mozilla Thunderbird are made up of two components: a user agent (UA) and a mail transfer agent (MTA). A UA program accepts mail memos to be sent from a user. The mail is delivered to a server by an MTA program. Mail is received using the protocols POP and IMAP.

9.11.1 Mail Message Format

An email consists of a message body, header lines and an envelope. The message body consists of text and attachments. The mail headers adhere to the standard format of Internet messages (RFC 2822). The headers are plain text lines, each line made up of a keyword, a colon, and a value. Certain keywords must appear among the headers. E.g., a TO address header is required. This specifies the email address of the recipient in the form of `mbx@dnm`, where `mbx` is the name of a mail box on the local machine, and `dnm` is the domain name of the destination. An optional header line is REPLY-TO. An envelope included the FROM and TO headers.

9.11.2 SMTP

Simple Mail Transfer Protocol (RFC 2821; 2001) defines the commands and replies among mail transfer agent (MTA) clients and servers. An MTA client transfers the mail spooled on the local machine across the Internet to an MTA server. The transfer can occur in a single connection between the original mail-sender and the final mail-recipient, or may go through intermediary systems known as MTA relays.

SMTP is specified to require only a reliable ordered data stream channel. Modern email clients use SSL/TLS as the transport.

9.11.3 POP

Post Office Protocol (POP) is used to send/receive email from a server. POP2 requires SMTP to send messages. POP3 can be used with or without SMTP. The POP protocol downloads entire message bodies, and can optionally keep the messages on the server. A user reading email from multiple machines may not have a consistent view of his messages.

9.11.4 IMAP

Internet Message Access Protocol (IMAP) is a protocol for accessing email while it is sill located on the server. IMAP4 supports encrypted login mechanisms and SSL for the transport.

9.11.5 MIME

Multipurpose Internet Mail Extensions (MIME) defines the format of messages to allow for: (i) Textual message bodies in character sets other than US-ASCII, (ii) An extensible set of different formats for non-textual message bodies, (iii) Multi-part message bodies, and (iv) Textual header information in character sets other than US-ASCII.

9.12 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 commands displayed by a Web browser are the so-called links that associate a URL (universal resource locator) with a (visible piece of) text or graphic. URLs have the following syntax:

```
scheme://[userName[:password@]]serverMachineName[:port]/[pa
th][/resource][?parm1=parma&parm2= parmb]
```

A simple example of the above is:
`http://www.cs.wright.edu/~pmateti/InternetSecurity`

where the scheme chosen is `http`, and the port defaults to 80. A click on such a link generates a request message from the browser to the Web server process running on the remote machine `www.cs.wright.edu`, whose name is obtained from the link clicked. The web server maps the path `~pmateti/InternetSecurity` to a local file according to the rules of the web server. This may be a pre-existing file, or generated on the fly. The server transmits a copy of this local file. The browser then displays the page it receives from the server.

9.12.1 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. The GET method requests the content of a web page. 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: HTTP/version-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, which the server sends.

9.12.2 Cookies

HTTP is stateless in that the HTTP server does not act differently to a specific request based on previous requests. Occasionally, a Web service wishes to maintain a minor amount of historical record of previous requests. Cookies (RFC 2965) 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 browser then creates a cookie stored on the browser's storage. In subsequent requests sent to the same server, the browser includes the header line "Cookie: value." Depending on the browser, cookies are stored in a database or as small files of text. The value of a cookie is not interpreted by the browser in any way. Cookie values often store user-specific information, such as a saved shopping cart, previous sites visited, user-name and password, and previous advertisements shown.

9.12.3 Authentication

Web servers requiring user authentication send a `WWWAuthenticate`: header. The browser prompts the user for a username and password, and sends this information in each of the subsequent request messages to the server.

10 Next Generation TCP/IP

The TCP/IP that is in wide use now (2006) is version 4. Version number 5 does not exist. Version 6 is available now but may take a few years to be widely deployed in place of v4. All U.S. federal agencies must deploy IPv6 by 2008. However, IPv4 is not expected to die even by 2010.

In TCP/IP v6 suite, IPv4 is replaced by IPv6, and ICMPv4 is replaced by ICMPv6. There is no TCP v6. The same UDP, TCP and infrastructure protocols are implemented over IPv6.

10.1 IPv6

IPv6 (RFC 2460) has several improvements over IPv4, in areas such as IP address size, authentication and security, header format, flow labeling, extensions and options, auto

configuration, and multicast routing. IPv6 was previously known as IPng (IP Next Generation).

10.1.1 IPv6 Header

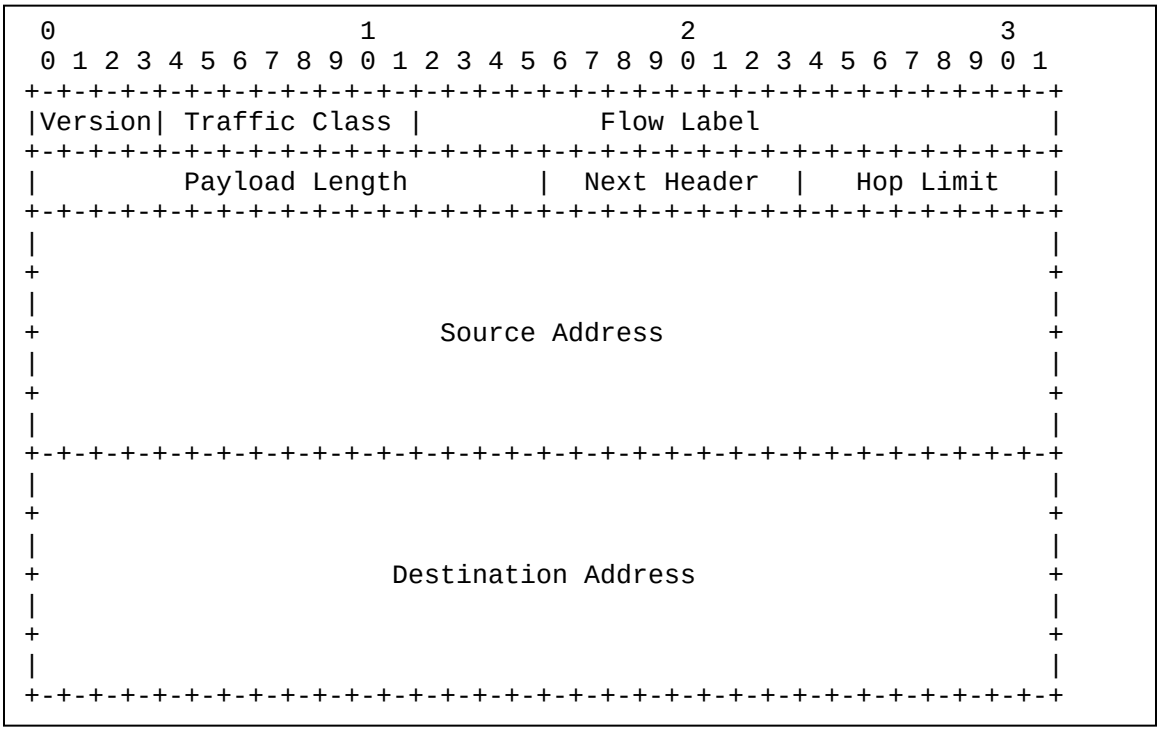

Figure 14: IPv6 Header

The value of the Version field (Figure 14) is 6. The 20-bit Flow Label specifies special router handling from source to destination(s) for a sequence of packets. A flow is a sequence of packets that a specific source sends to a specific destination. The Payload Length is a 16-bit unsigned integer specifying the number of bytes of data that follow the IPv6 header. The 1-byte Next Header identifies the type of header that follows the IPv6 header. The 1-byte Hop Limit (cf. TTL of IPv4) is an unsigned integer. It is decremented by one by each node that forwards the packet. The packet is discarded if Hop Limit becomes zero.

10.1.2 IPv6 Addresses

IPv4 addresses are only 4 bytes long, and the address space is nearly exhausted. IPv6 addresses are 6-byte long. This is sufficiently large that every cell phone and electronic device can be assigned a unique address.

When data is to be sent to a number of hosts simultaneously a *multicast* address destination is used. In IPv6, each multicast address can have a different scope, allowing a

transmission to be targeted to wide or narrow range of hosts. With a so-called *anycast* destination, data is sent to any one in a group of hosts.

10.1.3 Extensions

IPv6 options are placed in separate extension headers located between the IPv6 header and the transport-layer header in a packet. Many of these IPv6 extension headers are skipped by intermediate routers until the packet arrives at its destination. This is a major improvement in router performance. In IPv4, the router examines all options. Current IPv6 extensions are: Extended routing, Fragmentation and reassembly, Integrity authentication, and security, Encapsulation and Confidentiality, Hop-by-Hop, and Destination Options to be examined by the destination node.

10.1.4 Encrypted Security Payload

The Encrypted Security Payload (ESP) is an extension for data confidentiality and the prevention of easy decodability of information obtained through eavesdropping. The IPv4 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 IPv6 includes IPsec. The cryptographic protocols of IPsec provide authentication, encrypt data, and guarantee message integrity. It also provides for a key exchange protocol, such as the IKE (Internet Key Exchange) protocol.

10.1.5 Other Improvements over IPv4

Note that the following fields of IPv4 header (Figure 6) are dropped: IHL, Type of Service, Identification, Flag, Offset, and Header Checksum. The simpler header of IPv6 improves routing efficiency, performance, and forwarding rate.

An IPv6 host that wishes to auto configure sends a link-local request for its configuration parameters to which a router should respond with a router advertisement packet. A host can also use DHCPv6 or be configured manually.

10.2 ICMP6

ICMPv6 (RFC 2463, 1998) is used by IPv6 to report errors encountered in processing packets, and to perform other network-layer functions. ICMPv6 is a required companion to IPv6. IGMP and ARP are merged into ICMPv6 and RARP is omitted.

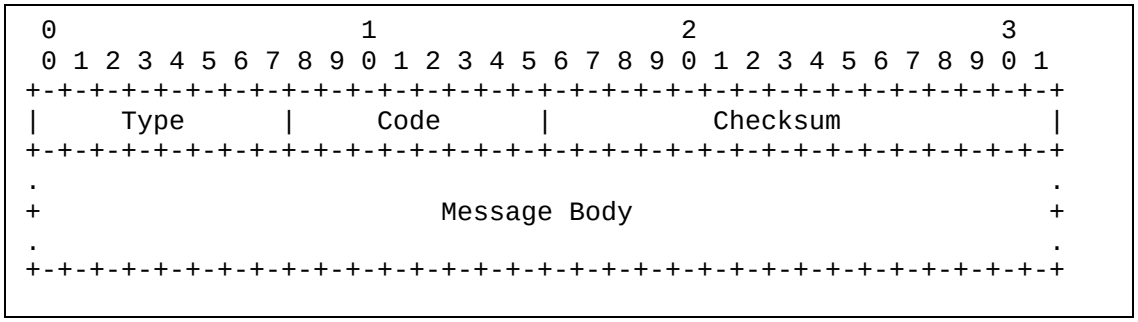

Figure 15: ICMPv6 Packet

Every ICMPv6 message (Figure 15) is preceded by an IPv6 header and zero or more IPv6 extension headers. ICMPv6 messages are either error messages or request/reply informational messages.

10.2.1 Error Messages

There are error messages in ICMPv6 corresponding to those in ICMPv4, except a Source Quench. The source-quench message is no longer needed because the `Traffic Class` Priority and `Flow Label` fields of IPv6 (Figure 14) are sufficient for congestion control purposes. Additionally, there is a new "Packet Too Big" message sent by a router when it receives an IPv6 datagram that is longer than the MTU of the outgoing link.

10.2.2 Request/Reply Messages

The Echo Request, Echo Reply, Router Advertisement, and Router Solicitation query messages are essentially as in ICMPv4. The ICMPv4 Timestamp Request, Address Mask Request messages are dropped.

10.2.3 Neighbor Discovery (ND)

ICMPv6 subsumes the functionality of ARP through the Neighbor Discovery (RFC 2461) protocol. ND uses ICMPv6 messages named Router Solicitation, Router Advertisement, Neighbor Solicitation, Neighbor Advertisement, and Redirect. Through these messages routers, next-hops, address prefixes, and parameters are discovered; IPv6 addresses are auto configured; un-reachability of neighbors and duplicate addresses are detected.

11 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.

12 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. See also *octet*.

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.

Node A synonym for host.

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 pre-agreed 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 is a fake address. In DNS spoofing, a query receives fake response.

Tunneling Sending packets of a certain protocol embedded in the packets of another protocol is called tunneling.

13 REFERENCES

Arkin, O. (2001). ICMP usage in scanning: the complete know-how. Retrieved December 2003 from http://www.sys-security.com/html/projects/icmp.html .

Butler, K., Toni Farley, Patrick McDaniel, Jennifer Rexford, "A Survey of BGP Security", 2005. Retrieved Mar 2006 from http://www.patrickmcdaniel.org/pubs/td-5ugj33.pdf .

Comer, D. (2000a). *Internetworking with TCP/IP: Vol. 1. Principles, protocols, and architecture* (4th ed.). Englewood Cliffs, NJ: Prentice Hall.

Comer, D. (2000b). *The Internet book: Everything you need to know about computer networking and how the Internet works* (3rd ed.). Englewood Cliffs, NJ: Prentice Hall.

Denning, D. E., & Denning, P. J. (1998). *Internet besieged: Countering cyberspace scofflaws*. Reading, MA: Addison Wesley. ISBN: 0201308207

Doyle, Jeff (2005). *Routing TCP/IP*. Cisco Press. Volume I: ISBN 1587052024, Volume II: ISBN 1578700892.

Behrouz A Forouzan (2005). *TCP/IP Protocol Suite* (3rd ed.). Boston, MA: McGraw-Hill. ISBN: 0072967722.

Garfinkel, S., Spafford, G., & Schwartz, A. (2003). *Practical UNIX and Internet security* (3rd ed.). Sebastapol, CA: O'Reilly.

Gourley, D., & Totty, B. (2002). *HTTP: The definitive guide*. Sebastapol, CA: O'Reilly. ISBN 1565925092.

Halabi, B. (2000). *Internet routing architectures*. Indianapolis, IN: Cisco Press.

Iren, S., Amer, P. D., & Conrad, P. T. (1999). *The transport layer: Tutorial and survey*. ACM Computing Surveys, 31, 360 - 404.

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

Kurose, J. F., & Ross, K. W. (2003). *Computer networking: A top-down approach featuring the Internet* (2nd ed.). Reading, MA: Addison Wesley.

Mateti, P. (2006). Internet security class notes. Retrieved Feb 2006 from www.cs.wright.edu/~pmateti/InternetSecurity

Rowland, C. H. (1997). Covert channels in the TCP/IP protocol suite. *First Monday*. Retrieved December 2003 from www.firstmonday.dk/

Stallings, W. (2003). *Cryptography and network security: Principles and practice* (3rd ed.). Englewood Cliffs, NJ: Prentice Hall.

Stevens, W. R. (1993). *TCP/IP illustrated: Vol. 1. The protocols.* Reading, MA: Addison Wesley.

Stewart, J. W., III. (1999). *BGP4: Inter-domain routing in the Internet*. Reading, MA: Addison Wesley. ISBN: 0201379511.

Tanenbaum, A. S. (2003). *Computer networks* (4th ed.). Englewood Cliffs, NJ: Prentice Hall.

14 FURTHER READING

TCP/IP details are part of many university 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 Forouzan, Tanenbaum, 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 Kurose and Ross, Halabi, and Doyle cover this topic extensively. BGP security is surveyed in the article by Butler et al.

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 (http://www.cs.wright.edu/~pmateti/InternetSecurity) that has lab experiments and readings online. The book by Garfinkel and Spafford explores security from a practical UNIX systems view.

The Web site `www.cert.org` issues timely and authoritative alerts regarding computer exploits and has a comprehensive collection of guides on security.

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.