SECURITY ISSUES WITH TCP/IP

Renqi Li **E. A. Unger**
Kansas State University

Abstract

An introduction to network security , basic definitions and aa brief discussion of the architecture of TCP/IP as well as the Open System Interconnection(OSI) Reference Model open the paper. The relationship between TCP/IP and of some OSI layers is described. An indepth look is provided to the major protocols in TCP/IP suite and the security features and problems in this suite of protocols. The secutiy problems are discussed in the context of the protocol services .

Keywords:

Unix, security, TCP/IP, network security

Introduction

Computer and network security is one of the major problems in the computer technology today. Security creates, in general, a trade-off with convenience and cost in decision makers' minds. No network that requires the highest degrees of openness, flexibility, easy access, cooperation, and interdependence can be very secure. Usually, most are not willing to forgo convenience remote access via network to their computers. Inevitably, organizations that accept this premise suffer from some loss of security [1], [3].

Networks are risky for at least three major reasons. First and most obvious, more points now exist from which an attack can be launched. A second reason is that the physical perimeter of the computer system has been extended. Messages received may be of uncertain origin; messages sent are often exposed to other systems on the net. The third reason is more subtle, and deals with an essential distinction between an ordinary dial-up modem and a network. Modems, in general, offer one service, typically the ability to log in; there are vulnerabilities in the login service, but it is a single service, and a comparatively simple one. Networked computers, on the other hand, offer many services: login, file transfer, disk access, remote execution, etc. Services that are more complex and difficult to protect. Some basic definition appear in Figure 1 [2], [5].

TCP/IP Suite Overview

The Open System Interconnection (OSI) Reference Model is the theoretical model that guides the development of protocols for computing systems. The OSI paradigm divides network issues into functions or layers. Each layer in the OSI Reference Model defines a level of function with greater abstraction at the higher levels. Compatibility of equipment can be defined within a layer, or lower-level implementations can be hidden to achieve compatibility at some higher level. The dual purposes of the model are to insure information flow among systems and simultaneous permit variation in basic communications technology. Moreover, network architecture flexibility is possible, i.e., one network may take care of the lower levels of the protocol and another network the higher levels, using gateways among the networks. These seven layers are given in Figure 2.

Connection-oriented operations: A user and a network set up a logical connection before the transfer of data occurs; additionally it is common that some type of relationship is maintained between the data units being transferred through out the duration of the user/network connection.

Connectionless-mode operations: No logical connection between the user and the network is established prior to the data transmission. The data units are transmitted as independent units.

Router: A system used to connect separate LANs (i.e., Local Area Network) and WANs (i.e., Wide Area Network) into an internet, and to route traffic between the constituent networks.

internet: a set of networks connected by routers.

Gateway: An IP router. Many documents use the term gateway rather than router.

Network Address: The 32-bit IP address of a system.

Port: A 2-octet binary number identifying an upper level user of TCP.

Hosts: The sources or destinations for information.

Packet: Originally meant a unit of data sent across a packet-switching machine. Currently, the term refers to a protocol data unit at any level.

Protocol Data Unit (PDU): A generic term for the protocol unit (e.g., a header and data) used at any layer.

Protocol: A set of functions governing the operation of a communication functions. For example, IP consists a set of rules for routing data, and TCP includes rules for reliable, in-sequence delivery of data.

Protocol Stack: A layered set of protocol functions that work together to provide communication between applications. For example, TCP, IP, and Ethernet make up a protocol stack.

Protocol Suite: A family of protocols that work together in an orchestrated and consistent fashion.

Segment: A protocol data unit consisting of a TCP header and optionally, some data.

Datagram: A data unit transported by IP protocol mechanisms.

Point-to-Point Protocol (PPP): A protocol for data transfer across serial links; supports extensive link configuration capabilities, and allows traffic for several protocols to be multiplexed across the link.

Figure 1. Basic Definitions

Layer 7: Application Provides paradigms for end-user and end-application functions, such as file transfer (FTAM), virtual terminal service(VTS), and electronic mail to interact with layer 6.

Layer 6: Presentation Translates data for use by layer 7, such as protocol conversion, data unpacking, encryption, and expansion of graphics commands.

Layer 5: Session Provides for the establishing of a session connection between two presentation entities, to support orderly data exchange.

Layer 4: Transport Relieves the session layer of concerns for data reliability and integrity by defining transparent transfer of data between session entities.

Layer 3: Network Contributes the means to establish, maintain, and terminate network connections among open systems, particularly routing functions across multiple network.

Layer 2: Data Link Defines the access strategy for sharing the physical medium, including data link and media access issues.

Layer 1: Physical Defines of the electrical and mechanical characteristics of the network.

Figure 2: ISO-OSI Protocol Layers

TCP/IP protocols

The specific layering used for the TCP/IP protocols was dictated by requirements that evolved in the academic and defense communities. IP provides functionality to allow different types of networks on an internet. TCP provides reliable data transfer. OSI Layering subsequently was strongly influenced by the way that TCP/IP was organized. The two lowest TCP/IP layers deal with the world of device drivers, media access controls, physical attachments, and physical signals. They package data into units, i.e., frames or packets, and send the data from an interface on the local system to a destination interface attached to the same physical network. Local area networks(LANs) and wide area network(WANs) provide these lower layer functions. The boundary between IP and the lower layers is an important one. When a vendor implements this boundary a new type of network interface and medium for IP can be added to computer without undue effort. IP can share a network interface and medium with other protocols. For example, both TCP/IP and DECnet traffic might share a single Ethernet interface.

The IP layer corresponds to OSI's network layer (layer 3). It routes data between hosts. The data may traverse a single network or may be relayed across several networks in an internet. Data is carried in datagrams. The IP layer is connectionless because every datagram is routed independently and IP does not guarantee reliable or in-sequence delivery of datagrams. IP routes its traffic without concern for to which application-to-application interaction a particular datagram belongs. In fact, OSI provides a connectionless network service that closely resembles IP. OSI also defines a connection oriented service at this layer.

TCP provides reliable data connection services to applications. TCP contains the mechanisms used to guarantee that data is error free, complete, and in sequence. TCP sends segments by passing them to IP, which routes them to the destination. TCP accepts incoming segments from IP, determines which application is the recipient, and passes data to that application in the order in which it was sent. The connection-oriented OSI transport layer also guarantees reliable data transmission.

User Datagram Protocol (Layer 4) are invoked send isolated message. UDP packages data into User Datagrams and passes them to IP for routing to a destination. UDP is a connectionless communication service that often is used for simple database lookups. OSI has defined a specification for connectionless transport, but this has not yet been implemented in products.

The higher OSI layers provide some interesting functions for instance, the session layer sets up and terminates application-to-application communications. Applications use the presentation layer to negotiate a data transmission format that both partners can understand. The TCP/IP protocol suite includes a set of standard application services including program-to-program communications, File Transfer Protocol(FTP), Simple Mail Transfer Protocol(SMTP), Telnet terminal access, and Domain Name System(DNS) directory services. Most TCP/IP product families support Network File System(NFS) file servers and file clients, as well as a set of network functions that have been built using a facility which is Remote Procedure Call(RPC). File transfer(FTP), mail(SMTP), and terminal access(Telnet) modules communicate with their peers via reliable TCP connections. Most NFS file serves exchange UDP messages with their clients, although there are a few NFS implementations that are built on top of TCP. The Domain Name System(DNS) protocols provide TCP/IP network directory services. DNS servers carry out most of their transactions by means of UDP messages, but
occasionally switch to TCP when larger amounts of data need to be moved.

Security Overview:TCP/IP Protocols

The Internet Protocol(IP) is a network layer protocol that routes data across an internet provideing the mechanisms needed to transport datagrams. A datagram is made up of an IP header and a unit of data to be delivered. IP is a packet multiplexer. Messages from higher level protocols have an IP header prepended.

IP is an example of a connectionless service. There is no notion of a virtual circuit or "phone call" at the IP level: every packet stands alone. It permits the exchange of traffic between two host computers without any prior call setup. Since IP is connectionless, it is possible that the datagrams could be lost between the two end users' stations because of the limitation of resources which are transmission bandwidth buffer memory and CPU processing. For example, the IP gateway enforces a maximum queue length size, and it this queue length is violated, the buffers will overflow. In this situation, the additional datagrams are discarded in the network. So IP is an unreliable datagram service. No guarantees are made that packets will be delivered, delivered only once, or delivered in any particular order. Nor is there any check for packet correctness. The checksum in the IP header covers only the header. Checks are not performed on the user data stream.

In fact, there is no guarantee that a packet was actually sent from the given source address. In theory, any host can transmit a packet with any source address. Authentication, and security in general, must use mechanisms in higher layers of the protocol. IP supports fragmentation operations. A packet traveling a long distance will travel through many hops. Each hop terminates in a host or router, which forwards the packet to the next hop based on routing information. During these travels a packet may be fragmented into smaller pieces if it is too long for a hop. A router may drop packets if it is too congested. Packets may arrive out of order, or even duplicated, at the far end. There is usually no notice of these actions: higher protocol layer(i.e., TCP) are supposed to deal with these problems and provide a reliable circuit to the application. If packet is too large for the next hop, it is fragmented.No reassembly is done at intermediate hops.

IP Routing Operations

Internet Protocol software runs in hosts and in IP routers. In general, a computer's IP software will allow it to act as an IP host, an IP router, or both. There is no need for TCP in a dedicated router, since application connections do not originate or terminate at the router. A router is, of course, connected to at least two networks. Modern router products are equipped with multiple interface hardware slots that can be configured with the combination of attachments that the customer needs: Ethernet, Token Ring, point-to point synchronous, fiber optic,etc.

TCP/IP networks use a 32-bit address to identify a host computer and the network to which the host is attached. The structure of the IP address is depicted in Figure 3.5. Its format is IP Address = Network Address + Host Address. The IP address just identify a host's connection to its network. If a host machine is moved to another network, its address space must be changed.

The IP address space can take the following forms:

	Network address space value
A	0 - 127 (Number 0 and 127 are reserved)
B	128 -191
C	192 - 223
D	224 - 254

The maximum network and host addresses that are available for the class A, B, and C addresses are as follows:

	Max. net. numbers	Max. host numbers
A	126	16,777,124
B	16,384	65,534
C	2,097,152	254

For convenience, the Internet addresses are depicted with decimal notations. As an example, a Class B internet address of binary 10000000 00000011 00001001 00000001 is written as 128.3.9.1. This address translates to network ID = 128.3 and host ID = 9.1. Therefore, the decimal notations for the IP address space can take the following forms:

A	network.host.host.host
B	network.network.host.host
C	network.network.network.host.

IP datagram has a number of service optional fields that may be invoked by a communication application, but they are not commonly used. The options consist of Strict Source Route, Loose Source Route, Record Route, Timestamp, Department of Defense Basic Security, Department of Defense Extended Security, No Operation, and End of Option List (Padding). From the security point of view, the important ones are strict and loose source routing and security labels. A Strict Source Route (SSR) consists a sequence of IP addresses of routers to be visited on the way to the destination. This service might be used as part of a security program to improve data security. Naturally, the traffic flowing back from the destination to the source should follow the same route; i.e., should visit the same set of routers in reverse order. The Loose Source Route option provides landmark destinations along the way. The datagram will visit the landmarks, but there may be intermediate hops on the way to each landmark. That is, traffic can pass through additional intermediate routers. So the loose source route can be viewed as a set of landmarks that help to find the way to a target host. Just as for Strict Source Routes, return traffic is expected to follow a corresponding Loose Source Route back. As before, incoming router addresses are replaced by the out-going router addresses that make sense to the destination node.

IP Security Labels

The IP security option is currently used primarily by military sites, although there is movement toward defining a commercial variant. Each packet is labeled with the sensitivity of the information it contains. The labels include both a hierarchical component (Secret, Top Secret, etc.) and an optional category: nuclear weapons, cryptography, hammer procurement, and so on. The labels indicate the security level of the ultimate sending and receiving processes. A process may not write to a medium with a lower security level, because that would allow the disclosure of confidential information. On the other hand, it may not read from a medium containing information more highly classified. The combination of these two restrictions will usually dictate that the processes on either end of a connection be at the exact same level. Some systems, such as Unix System V/MLS, maintain security labels for each process. They can thus attach the appropriate option field to each packet. For more conventional computers, the router can attach the option to all packets received on a given wire. Within the network itself, the primary purpose of security labels is to constrain routing decisions. A packet marked "Top Secret" may not be transmitted over an insecure link cleared only for "Bottom Secret" traffic. A secondary use is to control cryptographic equipment; that selfsame packet may indeed be routed over an insecure circuit if properly encrypted with an algorithm and key rated for "Top Secret" message.

Address Resolution Protocol (ARP)

The IP stack provides a protocol for resolving addresses. The Address Resolution Protocol(ARP) is used to take care of the translation of IP addresses to physical addresses and hide these physical addresses from the upper layers. There are some static or algorithmic mappings between these two types of addresses, a table lookup is usually required. The table provides the mapping between an IP address and a physical address. In a LAN (like Ethernet or an IEEE 802 network), ARP takes the target IP address and searches for a corresponding physical address in a mapping table. If it finds the address, it returns the 48-bit address back to the requester, such as a device driver or server on a LAN. However, if the needed address is not found in the ARP cache, the ARP module broadcasts a request onto the network. The ARP request contains an IP target address. Consequently if one of the machines receiving the broadcast recognizes its IP address in the ARP request, it will return an ARP reply back to the inquiring host. This packet contains the physical hardware address of the queried host. Upon receiving this packet, the inquiring host places this address into the ARP cache. Thereafter, datagrams sent to this particular IP address can be translated into the physical address by accessing the cache. The ARP system thus allows an inquiring host to find thephysical address of another host by using the IP address.

The Internet defines a mapping table to be used with the address mapping protocols. The mapping table consists of a row entry for each IP address on each host gateway (machine). Four columns are provided for each row entry. The ifIndex column contains the interface (physical port) for the specific interface for this address. The physical address entry contains the media dependent address. The IP address entry contains the IP address, which corresponds to the physical address. The mapping type entry is set to one of four values: other =1 (none of the following), invalid = 2 (the mapping for this row entry is no longer valid), dynamic = 3 (the mapping does not change), static = 4 (entry may change).

There is some risk, security holes in ARP, if untrusted nodes have write-access to the local net. Such a machine could emit phony ARP messages and divert all traffic to itself; it could then either impersonate some machines or simply modify the data streams. On special security networks, there is a way to solve this problem; the ARP machine is usually automatic, so one can make the ARP mappings statically hard-wired, and the suppress automatic protocol to prevent interference.

TCP Connection Operations and A Sequence Number Attack

TCP provides reliable virtual circuits user processes. Lost or damaged packets are retransmitted; incoming packets are shuffled around, if necessary, to match the original order of transmission. The order is maintained by sequence numbers in every packet. Each byte sent, as well as the open and close requests, are numbered individually. All packets except for the very first TCP packet sent during a conversation contain an acknowledgment number; it gives the sequence number of the last sequential byte success-fully received. The initial packet, with the SYN bit set, transmits the initial sequence number for its side of the connection. The initial sequence umbers are random. All subsequent packets have the ACK ("acknowledge") bit set.

Servers are processes that wish to provide some service via TCP - listen on particular ports. By convention server ports are low-numbered. The port numbers for all of the standard services are assumed to be known to the caller. When a connection request packet arrives, the other fields are filled in. If appropriate, the local operating system will close the listening connection so that further requests for the same port may be honored as well. Clients use the offered services. Client processes rarely ask for specific port numbers on their local host, although they are allowed to do so. They normally receive whatever port number their local operation system chooses to assign to them. The sequence numbers have another function. Because the initial sequence number for new connections changes constantly, it is possible for TCP to detect stale packets from previous incarnations of the same circuit (i.e., from previous uses of the same circuit: <local host, local port; remotehost, remoteport>). There is also a modest security benefit: a connection can not be fully established until both sides have acknowledged the other's initial sequence number. There is a threat lurking here. If an attacker can predict the target's choice of starting points - and it was showed ten years ago that this was indeed possible under certain circumstances - then it is possible for the attacker to trick the target into believing that it is talking to a trusted machine. In that case, protocols that depend on the IP source address for authentication can be exploited to penetrate the target system. This is known as a sequence number attack.

Two further points are worth noting. First the sequence number attack depended in part on being able to create a legitimate connection to the target machine. If those are blocked, the attack would not succeed. Conversely, a gateway machine that extends too much trust to inside machines may be vulnerable, depending on the exact configuration involved. Second, the concept of a sequence number attack can be generalized. Many protocols other than TCP are vulnerable. In fact, TCP's three-way handshake at connection establishment time provides more protection than some other protocols.

User Datagram Protocol (UDP)

The User Datagram Protocol(UDP) extends to application programs the same level of service by IP. UDP is classified as a connectionless protocol. Delivery is on a best-effort basis; there is no error correction, retransmission, or lost, duplicated, or re-ordered packet detection. Even error detection is optional with UDP. To compensate for these disadvantages, thereis much less overhead. In particular, there is no connection setup. This makes UDP well suited to query/response applications, where the number of messages exchanged is small compared to the connection setup/tear-down costs incurred by TCP. It is sometimes used in place of TCP in situations where the full services of TCP are not needed. For example, the Trivial File Transfer Protocol (TFTP) and the Remote Procedure Call (RPC) use UDP. An application that wants to send data via UDP passes a block of data to UDP. UDP adds a header to the block, forming a User Datagram. The User Datagram is then passed to IP and packaged in an IP Datagram. When UDP is used for large transmissions it tends to behave badly on a network. The protocol itself lacks flow control features, so it can swamp hosts and routers and cause extensive packet loss. UDP uses the same port number and server conventions as does TCP, but in a separate address space. Similarly, packets usually inhabit low-numbered ports. There is no notion of a circuit. All packets destined for a given port number are sent to the same process, regardless of the source address or port number.

It is much easier to spoof UDP packets than TCP packets, since there are no hand-shakes or sequence numbers. Extreme caution is therefore indicated when using the source address from any such packet. Applications that care must make their own arrangements for authentication.

Routers and Routing Protocols

Routing protocols are mechanisms for the dynamic discovery of the proper paths through the Internet. They are fundamental to the operation of TCP/IP. The most used routing protocols currently are Routing Information Protocol (RIP) and Open Shortest Path First (OSPF).

The Internet Gateway Protocol in widest use today is RIP. It's popularity is based on its simplicity and availability. RIP computes routes using a simple distance vector algorithm. Every hop in the network is assigned a cost. The total metric for a path is the sum of the hop costs. RIP tries to choose the next hop so that the datagrams will follow a best cost path. Like all automated routing protocols, RIP has to send routing updates, receive routing updates, and recompute routes. A RIP router sends information to its neighbor routers every thirty seconds. If the neighbors are on a LAN, the update is broadcast.

In the late eighties, the Internet Engineering Task Force started work on a new protocol to replace RIP. The result is the Open Shorties Path First (OSPF) Interior Gateway Protocol, a link state protocol intended for use within Autonomous Systems of all sizes. OSPF is designed to scale well and to spread accurate routing information through an internet quickly. In addition, OSPF supports, traffic splitting across multiple paths, routing based on Type of Service, and authentication for routing update messages. A node that is initializing obtains a copy of the current routing database from a neighbor. After that, only changes need to be communicated. Changes are known quickly because OSPT uses an efficient flooding algorithm to spread update information through the area (i.e., a set of networks): after receiving an update, a node sends the update to a selected set of neighbors.

Possible attacks can arise in variuos ways. Routing information establishes two paths: from the calling machine to the destination and back. From a security perspective, it is the return path that is often more important. When a target machine is attached, what path do the reverse-following packets take to the attacking host? If the enemy can somehow subvert the routing mechanisms, then the target can be fooled into believing that the enemy's machine is really a trusted machine. If that happens, authentication mechanisms that rely on source address verification will fail.

There are a number of ways to attack the standard routing facilities. One path attackers can take is to play games with the routing protocols themselves. It is relatively easy to inject bogus Routing Information Protocol (RIP) packets into a network. Hosts and other routers will generally believe them. If the attacking machine is close to the target than the real source machine, it is easy to divert traffic. Many implementations of RIP will even accept host-specific routes, which are much harder to detect.Some routing protocols, such as RIP and OSPF, provide for an authentication field. These are of limited utility for three reasons. First, the only authentication mechanisms currently defined are simple passwords. Anyone who has the ability to play games with routing protocols is also capable of collection passwords wandering by on the local Ethernet cable. Second, if a legitimate speaker in the routing dialog has been subverted, then its messages - correctly and legitimately signed by the proper source - cannot be trusted. Finally, in most routing protocols each machine speaks only to its neighbors, and they will repeat what they are told, often uncritically. Deception thus spreads.

Not all routing protocols suffer from these defects. Those that involve dialogs between pairs of hosts are harder to subvert, although sequence number attacks, similar to those described earlier, may still succeed. A stronger defense, though, is topological. Routers can and should be configured so that they know what routes can legally appear on a given wire.

Another attack is to employ the IP loose source route option. With it, the person initiating a TCP connection can specify an explicit path to the destination, overriding the usual route selection process. According to RFC 1122, in some networks, there is a rule that the destination machine must use the inverse of that path as the return route, whether or not it makes any sense, which in turn means that an attacker can impersonate any machine that the target trusts.

The easiest way to defend against source routing problems is to reject packets containing the option. Many routers provide this facility. Source routing is rarely used for legitimate reasons, although those do exist. For example, it can be used for debugging cer-tain network problems. You will do yourself little harm by disabling it. Alternatively, some versions of rlogind and rshd will reject connections with source routing present. This option is inferior because there maybe other protocols with the same weakness, but without the same protection.

Domain Name System (DNS)

To set up communication with a host, its name must be translated to a numerical address. Originally, each TCP/IP host kept a complete list of all of the names and addresses of all hosts on its network. It is now impossible to keep these lists current on a huge dynamically growing network. The Domain Name System (DNS) was invented to solve this problem. DNS consists of a database of hosts name and addresses distributed across a set of servers. DNS protocols enable users to submit database queries and receive database responses. In its normal mode of operation, hosts send UDP queries to DNS servers. Servers reply with either the proper answer or with information about smarter servers. Queries can also be made via TCP, but TCP operation is usually reserved for zone transfers. Zone transfers are used by backup servers to obtain a full copy of their portion of the name space. They are also used by hackers to obtain a list of targets quickly. Different records are stored by the DNS includin , address of a particular host, start of authority which denotes start of subtree; contains cache and configuration parameters, and gives the address of the person responsi-ble for the zone, Mail exchange, host type and operating system information.

The DNS name space is tree structured. For ease of operation, subtrees can be delegated to other servers. Two logically distinct trees are used. The first maps host names such as "depot.cis.ksu.edu" to addresses "129.130.10.5". Other information may optionally be included, such as HINFO or MX records. The second tree is for inverse queries, and contains PTR records (pointers to the host machines); in this case, it would map "3.160.130.129.in-addr.arpa" to "depot.cis.ksu.edu". There is no enforced relationship between the two trees, though some sites have attempted the mandate such a link for some services. This disconnection can lead to trouble. A hacker who controls a portion of the inverse mapping tree can make it lie. That is, the inverse record could falsely contain the name of a machine your machine trusts. The attacker then attempts an rlogin to your machine, believing the phoney record, will accept the call. Most newer systems are now immune to this attack. After retrieving the putative host name via the DNS, they use that name to obtain its set of IP addresses. If the actual address used for the connection is not in this list, the call is bounced and a security violation logged.

There is a more damaging variant of this attack. In this version, the attacker contaminates the target's cache of DNS responses prior to initiating the call. When the target does the cross-check, it appears to succeed, and the intruder gains access. A variation on this attack involves flooding the target's DNS server with phony responses, thereby confusing it. Although the very latest implementations of the DNS software are immune to this, it seems imprudent to assume that there are no more holes. It is strongly recommended that exposed machines not rely on name-based authentication. Address-based authentication, though weak, is far better.

There is also a danger in a feature available in many implementations of DNS resolvers. They allow users to omit trailing levels if the desired name and the user's name have components in common. For example, suppose someone on "foo.dept.big.edu" tries to connect to some destination "bar.com". The resolver would try "bar.com.dept.big.edu","bar.com.big.edu", and "bar.com.edu" before trying the correct "bar.com". There lies a risk. If someone were to create a domain "com.edu", they could intercept traffic intended for anything under ".com". And if they had any wild card DNS records, the situation would be even worse.

Authentication problems aside, the DNS is problematic for other reasons. It contains a wealth of information about a site: machine names and addresses, organizational structure, etc. Keeping this information from the overly curious is hard. Restricting zone transfers to the authorized secondary servers is a good start, but clever attackers can exhaustively search your network address space via DNS inverse queries, giving them a list of host names. From there, they can do forward lookups and retrieve other useful information.

Simple Mail Transfer Protocol (SMTP)

The SMTP standard is one of the most widely used upper layer protocols in the internet protocol stack. As its name implies, it is a protocol that defines how to transmit messages (mail) between two users.SMTP uses the concept of spooling. The idea of spooling is to allow mail to be sent from a local application to the SMTP application, which stores the mail in some device or memory. Typically, once the mail has arrived at the spool, it has been queued. A server checks to see if any messages are available and then attempts to deliver them. If the user is not available for delivery, the server amy try later. Eventually, if the mail cannot be delivered, it will be discarded or perhaps returned to the sender. This is known as an end-to-end delivery system, because the server is attempting to contact the destination to deliver, and it will keep the mail for a period of time until it has been delivered.

There are a number of possible attacks to SMTP. If the caller specified a return address in the "MAIL FROM" command. At this level there is no reliable way for the local machine to verify the return address; one must use some higher level mechanism if you need trust or privacy.

An organization needs at least one mail guru. It helps to concentrate the mailer expertise at a gateway (router), even if the inside networks are fully connected to the internet. Then administrators on the inside need only get their mail to the gateway mailer. The gateway can ensure that outgoing mail headers conform to standards. The organization becomes a better network citizen when there is a single, knowledgeable contact for reporting mailer problems.

From a security standpoint, the basic SMTP by itself is fairly innocuous. It could, however, be the source of a denial-of-service attack that is aimed at preventing legitimate use of the machine. Suppose it is arranged to have 50 machines each mail you 10001-MB mail messages. Can your system handle it? Can they handle the load? Is the spool directory large enough?

The mail aliases can provide the hacker with some useful information. Commands such as VRFY <postmaster> and VRFY <root> often translate the mail alias to the actual logic name. This can give clues about who the system administrator is and which accounts might be most profitable if successfully attacked. It's a matter of policy whether this information is sensitive or not.

The EXPN subcommand expands a mailing list alias; this is problematic because it can lead to a loss of confidentiality. A useful technique is to have the alias on the well-known machine point to an inside machine, not reachable from outside so that the expansion can be done there without risk.

The most common implementation of SMTP is contained in sendmail. Sendmail is a security nightmare. It contains of tens of thousands of lines of C and often runs as root. It is not surprising that this violation of the principle of minimal trust has a long and infamous history of intentional and unintended security holes. It contained one of the holes used by the Internet Worm and was mentioned in a New York Times article. Privileged programs should be as small and modular as possible and the SMTP daemon does not need to run as root.

TELNET

Telnet provides simple terminal access to a machine. The Telnet client exchanges data with the remote Telnet server over a TCP connection. The telnet server interacts with applications by emulating a native terminal. The TELNET data unit is called a command, which has three bytes the first one is the interpret as command (IAC) byte. This is a reserved code in TELNET. It is also an escape character, because it is used by the receiver to detect if the incoming traffic is not data but a TELNET command. The next byte is the command code. This value is used in conjunction with the IAC byte to describe the type of command. The third byte is the option negotiation code. It is used to define a number of options to be used during the session.

There are a number of possible attacks. As a rule, telnet daemons call login to authenticate and initialize the session. The caller supplies an account name usually a password to login. A telnet session can occur between two trusted machines. In this case, a secure telnet can be used to encrypt the entire session, protecting the password and session contents.

Most telnet sessions come from untrusted machines. Neither the calling program, calling operating system, nor the intervening networks can be trusted. The password and the terminal session are available to prying eyes. The local telnet program may be compromised to record user name and password combinations or log the entire session. This is a common hacking trick.

Traditional passwords are not reliable when any part of the communications is tapped. Hackers are are focusing on major network backbones. The use of a one-time password scheme is strongly recommended. A one-time password is one that changes every time it is used. Instead of assigning a static phrase to a user, the system assigns a static mathematical function. The system provides an argument to the function, and the user computes and returns the function value. Such systems are also called challenge-response systems, since the system presents a challenge to the user and judges the authenticity of the user by the user's response. One-time passwords are very secure for authentication, since an intercepted password is useless. The authenticators can secure a login nicely, but they do not protect the rest of a session. For example, wire tappers can read any proprietary information contained in mail read during the session. If the TELNET command has been tampered with, it could insert unwanted commands into your session, or it could retain the connection after you think you have logged off. It is possible to encrypt a TELNET session. But encryption is useless if you cannot trust one of the endpoints. Indeed, it can be worse than useless: the untrusted endpoint must be provided with your key, thus compromising it [6].

Network Time Protocol (NTP)

The Network Time Protocol (NTP) operation is a valuable adjunct to gateway machine. As its name implies, it is used to synchronize a machine's clock with the outside world. It is not a voting protocol; rather, NTP believes in the notion of absolute correct time, as disclosed to the network by machines with atomic clocks or radio clocks tuned to national time synchronization services. The primary time server,

upon receiving clocking information for a master clocking source, then uses the NTP protocol to coordinate clocks at the secondary time servers. Secondary time servers may in turn provide clocking for other secondary time servers. The accuracy of the clocking decreases as NTP messages are propagated through the clocking hierarchy. NTP messages contain several control fields including, Sync distance (to Primary clock), ID of Primary clock, time that local clock is updated, origin of timestamp, receive timestamp, transmit timestamp, and the authenticator. The sync distance is an
estimate of the round trip propagation delay to the primary clock, as seen by the originator of the NTP message. The ID of the primary clock contains the unique identifier of the primary time server. The last field is Authenticator. It is an optional field used for authentication purpose.

Possible attacks include the following. Knowing the correct time allows you to match log files from different machines. The timekeeping ability of NTP is so good (generally to within an accuracy of 10 milliseconds or better) that one can easily use it to determine the relative timings of probes to different machines, even when they occur nearly simultaneously. Such information can be very useful in understanding the attack's technology. An additional use for accurate timestamps is in cryptographic protocols; certain vulnerabilities can be reduced if one can rely on tightly synchronized clocks.

Log files created by the NTP daemon can also provide clues to actual penetrations. Hackers are fond of replacing various system commands and of changing the per-file timestamps to remove evidence to their activities. NTP itself can be the target of various attacks. In general, the point of such an attack is to change the target's idea of correct time. Consider, for example, a time-based authentication device or protocol. If someone can reset a machine's clock to an earlier value, he can replay an old authentication string.

To defend against such attacks, newer versions of NTP provide for cryptographic authentication of message. Although a useful feature, it is some what less valuable than it might seem, because the authentication is done on a hop-by-hop basis. An attacker who cannot speak directly to your NTP daemon may nevertheless confuse your clock by attacking the servers from which your daemon learns of the correct time. In other words, to be secure, you should verify that your time sources also have authenticated connections to their source, and so on up to the root. You should also configure your NTP daemon to ignore trace requests from outsiders.

Finger Protocol

Two standard protocols, finger and whois, are commonly used to look up information about individuals. The former can be quite dangerous. The finger protocol can be used to get information about either an individual user or the users logged on to a system. The amount and quality of the information returned can be cause for concern. To be sure, the most important output of finger - the mapping between a human-readable name and an electronic mail address - is very important. Finger is called by some people "one of the most dangerous services, because it is so useful for investigation a potential target." It provides personal information, which is useful for password guessers, when the account was last used (seldom or never used accounts are much more likely to have bad passwords), when the user last connected from (and hence a likely target for an indirect attack), and more.

Remote Procedure Call (RPC)

Remote Procedure Call protocol is a widely used software module developed by Sun Microsystems, Inc. It is used on almost all Unix-based systems. It greatly facilitates the distribution of applications to multiple machines. RPC is a remote subroutine call program. It allows a caller program, client, to send a message to a server. The caller program then waits for a reply message. RPC can be implemented on either a TCP or a UDP transport layer. Most of the essential characteristics of the transport mechanisms show through. Thus, a subsystem that uses RPC over UDP must still handle lost messages, duplicates, out-of-order messages, etc. Therefore, it will be up to the end user application to provide for retransmissions, timeouts, and other transport layer mechanisms.

The RPC message contains only three fields: remote program number, remote program version number, and remote procedure number. The purpose of the remote program number is to identify a group of procedures such as a database system, a file system, etc. The procedures are really macros or catalogued procedures such as a read or write. The procedures are identified with remote procedure number. In addition to reporting the results of successful calls, the server must be able to let the client know when its requests are rejected and why. A call may be rejected for reasons such as mismatched versions or a client authentication failure. The server needs to report errors caused by a bad parameter, or a failure such as "unable to find file". [7]

Every RPC call and reply is authenticated, when the application on the client machine issues an RPC request, it carries information about the user executing it. When the server sends results, they carry authentication information as well.Authentication information contains two parts: credentials and verifier. The credentials identify the user sending the message, and the verifer proves that user's identity. The application on the client machine and the procedure on the server machine must agree on a "flavor" of authentication. The RPC facility provides several flavors of authentication:

None: the credentials are empty and the verifier is empty. This means the message contains no information about the user that sent it.

Unix: the credentials contain the machine name, the effective user ID of the user sending the message, and the effective group ID of the user sending the message. However, the verifier field is empty. That means the sender does not prove his/her identity.

Secure: uses cryptography for verification of the caller's identity. The credentials contain the sender's identity, and the verifier contains an encrypted token that proves the sender's identity. So, not only does the application on the server machine know the identity of the uesr on the client machine, it is sure that the identity is correct.

Vendor-defined: lets you create your own authentication flavor. (For example, researchers at MIT have developed an authentication scheme named Kerberos[8]. Kerberos verifies a user's identity by using a three-party handshake scheme.

Usually, a null authentication variant can be used for anonymous services. For more serious services, Unix authentication field is included. Great care must taken here! The machine name should never be trusted, and neither the user-id nor the group-id are worth anything unless the message is from a privileged port. Indeed, even then they are worth little with UDP-based RPC; forging a source address is trivial in that case. Never take any serious action based on such a message. RPC does support cryptographic using DES, the Data Encrypting Standard.

(This is sometimes called Secure RPC.) Unfortunately, DES-authentication RPC is not well integrated into most system. NFS is the only standard protocol that uses it. Furthemore, the key distrubution mechanisms are very awkward, and do not scale well for use outside of local area networks.

RPC Portmapper has a security issue. The RPC protocol specificaton introduced a method for dynamically finding out what port a service is using. At each server host, a single RPC program, portmapper, acts as a clearinghouse for information about the ports that other RPC program use. With an exception of NFS, RPC-based servers do not normally use fixed port numbers. They accept whatever port number of the operating system assigns them. The portmapper maintains a list of the local active RPC programs, their version numbers, transport protocols, and the ports at which they are operating. The portmapper - which itself uses the RPC protocol for communication - acts as an intermediary between RPC clients and servers. To contact a server, the client first asks the portmapper on the server's host for the port number and protocol (UDP or TCP) of the service. This information is then used for the actual RPC call.

The portmapper has other abilities that are even less benign. It is always will inform anyone on the network what services you are running; this is extremely useful when developing attacks. The most serious problem with the portmapper, though, is its ability to issue indirect calls. To avoid the overhead of the extra routing necessary to determine the real port number, a client can ask that the portmapper forward the RPC call to the actual server. But the forwarded message, of necessity, carries the portmapper's own return address. It is thus impossible for the applications to distinguish the message from a genuinely local request, and thus to access the level of trust that should be accorded to the call.

Summary

The use of an operating system like Unix which uses the facitities provided by the TCP/IP protocol suite opens the environment to a number of security problems. Some of these are avoidable but most require significant policy decisions, encryption, additional security products for authentication of users and for temporal checks on the state of the system. The protocols as they are used to create products are being improved. At the current time it is important for the system manager to be aware of the security holes in the suite. The references in this paper provide considerable insight into the problems.

References

[1]. William R. Cheswick, Steven M. Bellovin, Firewalls and Internet Security: Repelling the Wily Hacker, Addison-Wesley Publishing Company, 1994.

[2]. Sidnie Feit, TCP/IP: Architecture, Protocols, and Implementation, McGraw-Hall, Inc., 1993.

[3]. Uyless Black, TCP/IP and Related Protocols, McGraw-Hall, Inc., 1992.

[4]. Craig Hunt, TCP/IP Network Administration, O'Reilly & Associates, Inc., 1992.

[5]. Thomas W. Madron, Network Security in the '90s: Issues and Solutions for Managers, John Wiley & Sons, Inc., 1992.

[6]. Steven L. Shaffer, Alan R. Simon, Network Security, AP Professional, 1994.

[7]. Charles P. Pfleeger, Security in Computing, Prentice Hall, Inc., 1989.

[8]. Michael Padovano, Networking Applications on UNIX System V Release 4, Prentice Hall, Inc., 1993.