A NETWORK PERIMETER WITH SECURE EXTERNAL ACCESS

Frederick M. Avolio

Marcus J. Ranum

Trusted Information Systems, Incorporated
Glenwood, MD

ABSTRACT

A private network that carries sensitive data between local computers requires proper security measures to protect the privacy and integrity of the traffic. When such a network is connected to other networks, or when telephone access is allowed into that network, the remote terminals, phone lines, and other connections become extensions to that private network and must be protected accordingly. In addition, the private network must be protected from outside attacks that could cause loss of information, breakdowns in network integrity, or breaches in security.

While security is important, security measures that are onerous or cumbersome often end up being circumvented by legitimate users of the network in order to get their work done. Because of this, usability — or "user friendliness" — in security features is also of the utmost importance.

Trusted Information Systems, Inc. (TIS) has built a prototype system that provides for strong user authentication, access control, and integrity protection for unclassified but sensitive data on a private (isolated) network (or collection of networks). Furthermore, the prototype system supports the secure connection of the private network to an external internet, as well as dial-up network connections to the private network, via a firewall and secured links, with strong user authentication and encryption of traffic. TIS used a combination of commercial off-the-shelf (COTS) software[1] and custom software for this project.

This paper summarizes the extended system configuration and functional services, and describes the required security services and specific protection mechanisms used to provide these services.

[1] During the course of this paper, any products or services mentioned by name are mentioned only as examples of existing technologies or systems, and should not be construed as product endorsements.

INTRODUCTION

The goal of this work was to take an established and clearly demarked security perimeter and extend it. Extensions could be from the internal network to homes or hotel rooms over phone lines or to users on other hosts on another, outside network, over an external network. The extension of the security perimeter is done for selected services, under well-understood controls, without compromising security.

The initial installation of this work was prototyped for an extended LAN (referred to as the *campus network*) supporting end users who want to be able to do one or more of the following:

- Use a portable computer outside the office in support of their work.
- Exchange electronic mail (e-mail) with users on external networks, such as the Internet.
- Remotely connect from a portable computer or another network into the campus network for access to their files, reading electronic mail, etc.
- Access Internet services, both commercial and non-commercial, from their desktops or while remotely connected from home, a hotel, an airplane, etc.
- Support strong user authentication and privacy in communications.

As a separate task, we were asked to provide a mechanism for examining and verifying incoming mail and routing of validated mail for special handling.

RISKS AND ASSUMPTIONS

On the basis of the needs expressed by the users, TIS drew up the policies and assumptions that would affect how security was implemented. We did not try to quantify the probability of the risks, but included all that seemed possible. We recognized the following risks and make the following assumptions:[2]

- The data we are protecting, while not classified, is highly sensitive and would do damage to the organization and its mission if disclosed or captured.
- The integrity of the campus network directly affects the ability of the organization to accomplish its mission.
- The campus network is physically secure; the people using the campus network are trustworthy.
- Machines on the campus network are considered to be unsecure. We rely on the physical security of the campus to protect them.
- Whenever possible, staff members who are connected from remote sites should be treated as members of the campus network and have access to as many services as is possible without compromising campus security. The security perimeter shall be extended to include them.
- The Internet is assumed to be unsecure; the people using the Internet are assumed to be untrustworthy.
- Staff members are targets for spying; information they carry or communicate is vulnerable to capture.
- Passwords transmitted over outside connections are vulnerable to capture.
- Any data transmitted over outside connections are vulnerable to capture.
- There is no control over e-mail once it leaves the campus; e-mail can be read, tampered with, and spoofed.
- Any direct connection between a campus system (computer) and one on the outside can possibly be compromised and used for intrusion.
- Software bugs exist and may provide intrusion points from the outside into the campus.
- Password-protected accounts on any campus machine directly reachable from the outside can be compromised and used for intrusion.
- Telephone use by staff members outside the campus is intercepted and recorded. This includes data transmissions (modem connection to the campus).
- Security through obscurity is counter-productive. Easy-to-understand measures are more likely to be sound, and are easier to administer.

POLICIES

On the basis of these assumptions about the environment (both inside and outside) and the risks, we put together a security policy. Its salient points are:

- We are implementing a perimeter defense.
- A tight security perimeter is our main goal. The ability to extend the security perimeter to include staff members at remote sites (at home, traveling, in remote offices) is a close second.
- Security is more important than service; when they cannot be reconciled, security wins.
- The security policy is made to be changed with a change in risks or business (service) needs.
- "That which is not expressly permitted is prohibited. [5]" The security perimeter must be designed to block everything, and services must be enabled on a case-by-case basis only after a careful assessment of need and risk.
- Even if there is a bug in the implementation of a network service, it should not be able to compromise the campus.
- Direct network connections from outside to inside will never be permitted; proxy servers will be used.
- Network services should be implemented with a minimum of features and complexity, allowing thorough and quick review of the source code.
- The completed system should be able to be tested to ensure that it meets security goals.

[2] We purposely use general terms here because these may be applicable to any organization. However, every organization has some different risk concerns and business requirements.

- Dial-in connections will be controlled through strong user authentication and encryption.

- Passwords will never be transmitted “in the clear”; if they must be transmitted in an unsecured fashion, one-time passwords will be used.

- Official communications via e-mail will contain digital signatures for authentication of the sender and non-repudiation.

- An individual user should be able to use the same mechanism for user authentication across all services that require it.

- Data on mobile computers will be encrypted.

SYSTEM OVERVIEW

The prototype system provides protection for the campus network and its Internet and dial-up connections. This work forms a model that can be used for secure extensions of other private networks. A primary goal of this work was to provide secure and easy-to-use remote access to the protected network, providing protection against accidental or malicious modification or disclosure of data, and providing strong user authentication. The system configuration appears in Figure 1.

The prototype system incorporates the following protection mechanisms to support our goal of being able to extend the security perimeter.

- A strong user authentication service to establish user identity

- File integrity and protection on desktop and portable computers, via disk (or file system) encryption with user authentication mechanisms

- E-mail security using Privacy Enhanced Mail [1] software to provide a mechanism for examining and verifying incoming mail, based on valid digital signatures, and for identifying validated mail for special handling

- Encryption of all communication between the campus and remote users, including link encryption for dial-up connections

- An Internet firewall toolkit to enforce the security perimeter and provide secure Internet connection for e-mail, FTP, TELNET, and other services

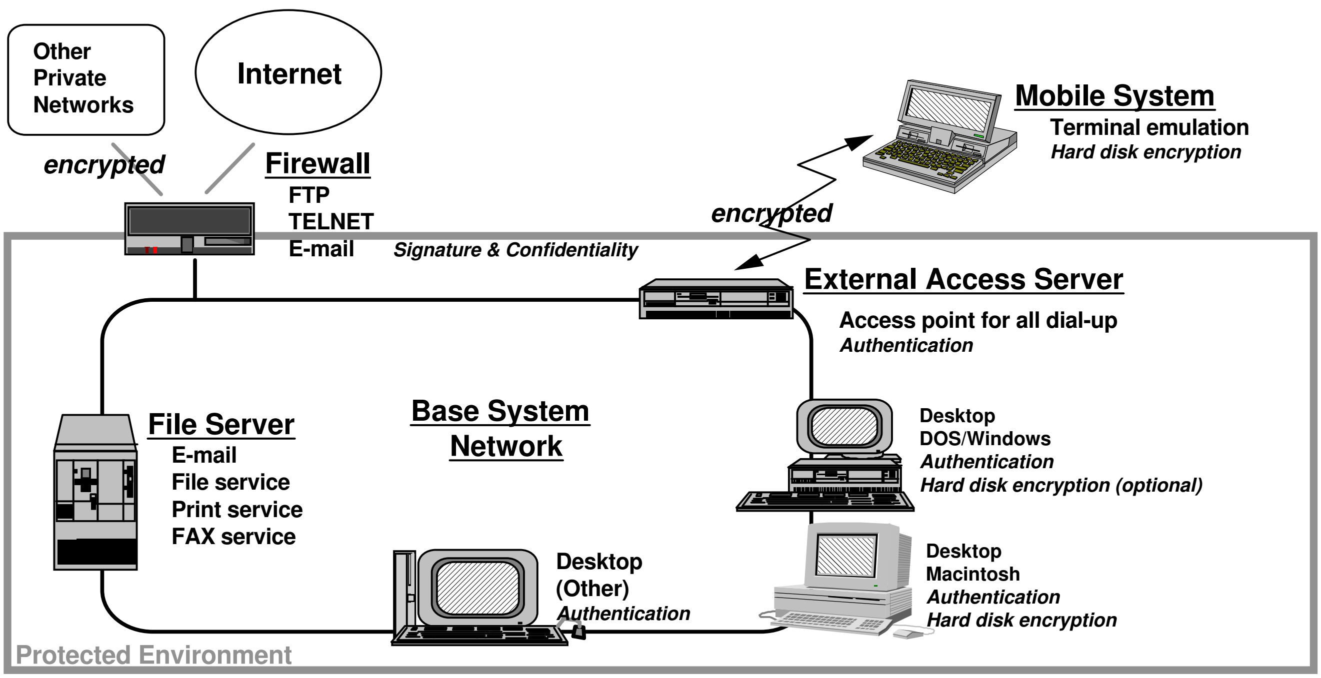

Figure 1: System overview

This paper discusses each of these protection mechanisms as employed in the prototype system.

USER AUTHENTICATION

We use "authentication" as defined by the National Computer Security Center's "Red Book" [2] as "(1) to establish the validity of a claimed identity or (2) to provide protection against fraudulent transactions by establishing the validity of ... the individual ...." Identification of a user is often accomplished on computers through the use of a user name and password pair. The password is kept secret and must be difficult to guess; only the user knows the proper name and password pair to use. In reality, passwords are often weak (guessable). Further, in the case of identifying users over outside communication links, there exist opportunities for capture of the user name and password information (although the password is usually not echoed, it is transmitted over the communications link "in the clear"). Consequently, while it would seem that a user name and password pair constitute good identification criteria, the password is too easily guessed or captured. In the prototype system, authentication of a user is done in such a fashion that we can apply a high degree of trust to the identification. This can be accomplished with one-time passwords, or authentication devices or tokens such as Digital Pathways SecureNet or Security Dynamics SecurID. We use all three mechanisms, as examples, to show different ways that strong user authentication can be done.

One-time passwords are passwords that are used only once. A user in a trusted or protected environment, before leaving for a remote location, will identify himself to the computer and generate a list of passwords or phrases to use as part of a challenge response system. The list generated by the computer gives sets of word pairs or numbers and a set of words associated with those numbers [3].

Challenge	Response
coddle	slugfest
toaster	require
doghouse	skateboard
eagle-eye	strummer

Figure 2: Word pairs

Challenge	Response
22	want dew hurl wavy otter stop
23	wise gal miss be king ball
24	iris a gap lure now red
25	stun otiose tom too oven glow

Figure 3: Number and Word List Combinations

If a user was to login and gets "doghouse" in response to the user name entered, the user would have to enter "skateboard" in response (see Figure 2). This would validate the user to the system. It would not matter if someone captured that data on the communications line because the challenge "doghouse" would never be given again with the expected response of "skateboard." A more elaborate system is illustrated by the second table (Figure 3). A publicly available program called S/Key works in this fashion.[3]

The benefit of a word list method such as S/Key is that it is easy to use, it is flexible, and it is inexpensive to implement. Before a user leaves the campus, a list of challenges and responses can be generated, or the user can rely on mobile computer-based software to generate responses.[4] S/Key generates them based on a secret password the user shares with S/Key on a server on the campus network. No challenge will be issued more than once; even if someone was to capture the remote user's challenge and response, it would be unusable. There is a risk of compromise, however, if someone gets hold of the list, knows what it is for, and knows the associated login name.

We recommend the mobile PC-based approach to response generation, if using this method, in which the response is generated only after the user types in his password to the program. While this method is not as secure as the next two we will discuss, for many organizations it will be "good enough" based on their security policies.

[3] A one-time password program using word pairs is in use at the Massachusetts Institute of Technology. S/Key, a publicly available package from Bellcore, implements this second example of a numeric challenge and a word list response.

[4] Mac and DOS software exist.

User authentication with systems like the Digital Pathways requires the user to have a hand-held device about the size of a pocket calculator. The user identifies himself to the card using a personal identification number known only to the user. The user logs in using his login name and is given a numeric challenge. The user then keys the challenge into his encryption device and reads the response his device displays. This is then typed in as the response to the challenge. The software on the host knows the seed used for encryption on the handheld device issued to the user. The host does the same calculations as the device does. Since the challenge is random, the response given is of no use to anyone eavesdropping on the connection. Practically speaking, that particular numeric challenge will never be given to that particular user again. If the "password" is compromised, it is not a security threat, since the "password" is never reused.

SecurID, from Security Dynamics, operates in a similar manner. Each SecurID card has a unique seed used for encryption. Every 30 or 60 seconds (depending on the model used), the card shows a different numeric value, based on the seed and the date and time. The server software applies the same algorithm, based on the login name used (and a table indicating which card is issued to that user). This removes the challenge step from the authentication process, but otherwise is similar to other methods of user authentication. A personal identification number is used on this device also. While SecurID has versions of their device that do not require entering a PIN, we recommend user identification to the card to avoid the risk mentioned with S/Key (above). The vulnerability with SecurID is that there is a fixed time period when the same response would work. This is easily fixed in software, allowing only 1 login per period. We used both Digital Pathways cards and SecurID cards in our implementation. Other products could be used as well.

Access to the outside from the inside for TELNET or FTP can require user authentication. Since the user is already authenticated in this manner on the inside network, we made a policy decision not to re-authenticate if the connection originates on the inside. All access to the internal network by users on the outside — via TELNET, FTP, or modem — requires user authentication. In this way, we can trust that the remote user is who he or she claims to be.

Access for "dial-in users" is provided by an authentication server that supports S/Key, SecurID, and SecureNet authentication of the remote users. Once connected and authenticated, a remote user will be considered a member of the campus network and will have full access to all services provided. The authentication server runs on a general-purpose host holding the user authentication data bases and is available as a network service (as described in the firewall toolkit section, later).

The network authentication server provides a generic authentication service for network applications. The authentication server must run on a secure host, since its database could be a point of attack. It can embed support for multiple forms of authentication systems simultaneously.

FILE INTEGRITY AND PROTECTION

Through the use of encryption and access control based on strong user authentication (the same user authentication described above), computer and file access may be limited and controlled. Desktop computers on the campus can be configured to be privately owned or general-use computers. Any individual with an account on the campus network is able to use any general-purpose desktop computer to access his or her personal files on the network. Users authenticate themselves to the network servers by using authentication technology through an authentication server, and so access to personal files is permitted. Unauthorized access is prohibited. Whether this form of file protection is used varies with the workstation and the office it is in. The assumption that the campus was a protected environment allowed for looser protection on computers within the environment and, so, fits the security policy.

Personal desktop computers can be configured such that only the "owner" has access to the computer. In this case, commercial off the shelf software (such as Watchdog or User-EZ) is used to encrypt the local disk based on a password or authentication device identifier. Only the authorized user of the computer can gain access to the local files, because only with the proper password or response to a challenge (again, as with user authentication above) can the disk be decrypted. As with network file servers, the policies permit us to forgo strong user authentication in this case.

Given the risks, assumptions, and user requirements, portable personal computers were a special security concern. If the notebook computer is used for accessing and working with sensitive data, there must be a way to protect that data while on the portable computer. The portable computer, even when disconnected from the campus network, must still be considered an extension to the network and protected accordingly. It is too easy to leave a notebook PC behind in a taxi or plane, or to have someone examine it if it is left behind in a hotel room.

Therefore, notebook computers are also protected with disk encryption. To gain access to the notebook computer, the user will have to authenticate herself to the unit using a token (e.g., SecurID) or a password. The notebook computer's disk is encrypted and unusable without valid authentication. Risks associated with the use of such software, besides the implementation being dependent on proper use, are the problems of "instant off/on" modes on portable computers (bypassing the initial "login" sequence), and the computer being stolen while powered up. All software we checked had a time-out option, requiring periodic re-authentication of the user if the system is idle.

SECURE ELECTRONIC MAIL

To secure e-mail, additional security measures are desirable, especially for official business. Privacy Enhanced Mail (PEM) [4] provides three essential features necessary for official business:

- Integrity: Any electronic data received via PEM can be shown to be the same data that was sent. In other words, it can be proven that what the reader is reading is what the sender sent; no one changed the data in transit. This doesn't prevent tampering, but it does detect it.

- Authentication and non-repudiation: The sender, using PEM, digitally signs the e-mail with a digital signature only the sender can use. Digitally signed e-mail can be checked for a valid and true signature, just as the messages can be checked for integrity. A digital signature, then, provides for authentication of the sender information on e-mail, as well as non-repudiation by the sender (it can be proven that only the sender — or someone with access to the sender's private encryption key — could have sent the message).

- Privacy: Using any one of various encryption algorithms available, e-mail can be sent from one user to another, such that only the intended reader can decrypt the message.

In addition to using these features on individual mail messages between users, a PEM-based gateway and router is used on a mail gateway or hub.[5] This mail hub software can be configured to selectively sign or encrypt outgoing messages. This provides automation of these services between networks or from the campus network to a set of outside user mailboxes. This is currently in use at TIS but not at EOP.

The mail hub software also can examine incoming e-mail for validly signed electronic mail. If a valid digital signature is found, this signature is checked against a list of signatures for "smart routing" of the mail. In an organization that receives a large number of electronic messages for an individual, such as is the case for e-mail for a marketing department or high level executive, it is desirable for the computer system to make some mail routing decisions before human intervention is needed. Some mail might go to general handlers, such as is the case for the majority of mail received by a marketing department. Mail from certain individuals, however, might need to receive special handling or go directly to an individual recipient. For example, mail addressed to a special recipient with a digital signature indicating that it is from an individual on a list of high-priority senders gets different handling than unsolicited e-mail addressed to the same mailbox. A PEM-based gateway and router is able to do this automatic checking and routing.

Finally, this same mechanism could be used for digitally signing official publications sent by e-mail to recipients or posted to Usenet. Since it is trivial to fake e-mail and Usenet postings, there is a significant benefit to e-mail and Usenet postings that can be checked for integrity and authenticity. Only valid digitally signed publications, as an example, would be released to Usenet where they can be independently verified.

ENCRYPTED REMOTE COMMUNICATIONS

Protection of remote connections to the campus network is provided in four distinct ways:

- TELNET connection from an Internet site: Special encrypting software for an encrypted TELNET session was written. The TELNET session is encrypted from a special TELNET client to a special TELNET server.[6] This allows extending the security perimeter to include the remote user and terminal (although it does not include the remote network). This software will evolve to match Internet standards as they develop.

[5] A *mail gateway* is a computer system that sits between one environment and another and handles the relay of mail. A *mail hub* is a system that acts as a relay site from gateway systems and other mail hubs.

[6] TELNET encryption will use the Internet standard when available.

- Dial-up connection via modem and terminal emulation software: This uses an encrypting terminal emulator and server or encrypting modems. This could also use a cellular telephone connection. The terminal emulator on the remote system (a portable computer, for example) encrypts its end of the session. At the private network side of the connection, a decrypting process runs on an external access server. Because the entire link is encrypted, the security perimeter is extended to include the remote user and terminal. We build this software by modifying a version of Kermit to do encryption.

- Dial-up connection for Serial Line Internet Protocol (SLIP) or Point-to-point Protocol (PPP): This provides an encrypted TCP/IP session, using software or encrypting modems. If the software encryption is employed, this could also be done over a cellular telephone link.[7] There is a risk with allowing this. With this method, there is no way to prevent a user from connecting a whole external network to the campus via SLIP. A user can connect his mobile computer to a network and then, via modem, connect to the campus network. If the mobile computer software allows IP forwarding (allowing the computer to act as a network router), the campus network would be compromised, lowering the level of security to the level of security on the remote, unknown network. Because of this, we recommend caution with allowing this service. Encrypted SLIP was written but will evolve to support any Internet standards developed.

- Point-to-point encryption: Encrypting routers or software allow for encryption of all IP packets between pairs of networks. The benefit of an encrypted link, beyond the obvious observation that no data is ever sent "in the clear" over untrusted paths, is that it allows the joining of network "islands." Because of this, users and hosts on a remote network can be treated as local, trusted users. The security perimeter can be extended to include communication packets over the outside network and the entire remote network. Note that while this is permitted, this should only be allowed if both sites share a common security plan or profile. There are commercial products to support this. We used UUNET Technologies' LAN Guardian in our prototype.

[7]At this writing, we do not know of any commercial modems that support both cellular *and* encrypted communications.

SECURE INTERNET CONNECTION: THE FIREWALL TOOLKIT

The connection to the Internet is through a filtering router in conjunction with a security and applications server. This server runs a version of the UNIX operating system and provides e-mail and DNS service, as well as application support. There are commercial Internet Firewall products that support such a configuration, including those from ANS, Digital Equipment Corporation, and Raptor. As part of this project TIS developed an Internet firewall toolkit consisting of software modules and configuration guidelines, to provide a publicly available base for "industrial strength" firewall security for organizations who desire to build their own firewalls.[8]

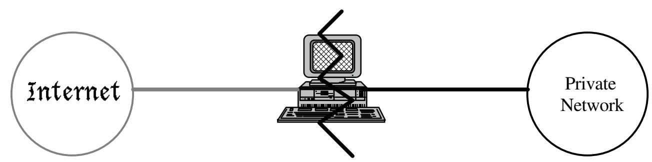

Figure 4. An Internet Firewall

The rationale for installing a firewall is almost always to protect a private network against intrusion. In most cases, the purpose of the firewall is to prevent unauthorized users from accessing computing resources on a private network, and often to prevent unnoticed and unauthorized export of proprietary information. In some cases, export of information is not considered important, but in many cases this is a major, though possibly unwarranted, concern. Many organizations will want to address the problem by not connecting to the Internet at all. This policy can be difficult to enforce. If the private network is loosely administered or decentralized, a single enterprising individual with a high speed dial-up modem can quickly arrange an Internet SLIP connection that can compromise the security of an entire network [5].

The purpose of an Internet firewall is to provide a single point of defense with controlled and audited access to services, both from within and without an organization's private network. Internet firewalls tend to be implemented in one of three ways: either (1) with the security engineered on one or more hosts, (2) with routing between the private network and the Internet (or any two

[8]The TIS Firewall Toolkit is available in source form via anonymous ftp from `ftp.tis.com:/pub/firewall/toolkit/fwtk.tar.Z`.

networks) blocked, or (3) via screening rules in a commercial router, with direct routing between the Internet and the private network. This design decision sets the general stance of the firewall, favoring either a higher degree of service or a higher degree of isolation. In either case, it is sometimes desirable to support proxy forwarders [5] on the firewall, to act as a gateway for specific applications such as FTP or the X Window System. A proxy forwarder for a network protocol is an application that sits on a firewall host and connects specific service requests on one side of the firewall with servers on the other side, in a controlled, auditable, selective, and secure fashion, and often gives the illusion to the software on both sides of a direct point-to-point connection.

The TIS Firewall Toolkit is designed to be used with a host-based security policy, but its components can be used with router-based firewalls. In this paper, we will focus on the former. In a host-based firewall, the security of the host is crucial; once it is compromised the entire network is often open to attack. Still, we believe that a host-based firewall is superior to other designs because of the ease with which it can be maintained, configured, customized and audited. The TIS Firewall Toolkit is designed to be used in conjunction with router-based screening as extra security. To minimize risks, the services that are provided on the external machine ("bastion host") [5] are sharply curtailed and each service is subjected to review. On the "standard" firewall configuration, the only services supported are DNS, SMTP, FTP, NNTP, TELNET (via proxy and forwarding servers), and user authentication. Other proxies such as Treese's X Window System proxy [6] can be added to this architecture.

The firewall toolkit functionality can be broken down into 6 areas: logging, electronic mail, the Domain Name Service, FTP, TELNET, and TCP access control.

Logging

Through the `syslog` facility, all significant security events are logged to a protected host on the internal network. The version of `syslogd` that the toolkit uses is based on the BSD "net2" sources, with some modifications to support pattern-matching and program execution on matched patterns. Many systems administrators have `cron` jobs set up on their systems to alert them of possible security problems by searching the system logs at regular intervals. By permitting the systems manager to easily add regular expressions to the `syslogd` configuration, security-related log messages can be identified instantly, before log files can be tampered with. `Syslogd` contains further modifications that permit an arbitrary command to be invoked with any specified logging rule, so that, for example, vitally important security log events can be delivered to the systems manager's beeper, or delivered immediately by electronic mail.

Electronic Mail

Mailers are one of the favorite points of attack against UNIX systems. The Morris Internet worm exploited a well-known hole in the standard UNIX SMTP server, `sendmail`. Many systems running sendmail, including those with Internet firewalls, were penetrated by the worm. A few that had replaced sendmail with other SMTP servers were not [7]. Typically, the problem with mailers is twofold: they are complex and perform file system activity, and they often require privileges so that they can manipulate users' mailboxes.

To secure mail service, direct network access to sendmail is prevented. A simple program that implements a skeleton of the SMTP protocol is presented on the SMTP port on the mail server. This sendmail-proxy, called *smap*, is small enough to be subjected to a code review for correctness (unlike sendmail) and simply accepts all incoming messages and writes them to disk in a spool area. Rather than running with permissions, the sendmail-proxy runs with a restricted set of permissions and runs "chrooted"[9] to the spool area. A second process is responsible for scanning the spool area and delivering the mail messages to the real sendmail for delivery — a mode of operation in which sendmail does not require permissions for operation. Many Internet firewalls run sendmail and rely on "trustworthy" versions of the software; running the mail software in a reduced-permissions mode is a more general solution to the problem, neatly side-stepping the issue of whether or not a given version of sendmail contains bugs.

While smap answers all valid sendmail SMTP commands sent to it, it does not execute any of them except those directly involved with mail exchange: HELO, FROM, RCPT, DATA, and QUIT. Smap preserves sendmail's functionality, while preventing an arbitrary user on the network from communicating directly with sendmail. Analyzing the sendmail program's 20,000 lines of source code for bugs is a sizable task when compared to analyzing smap's 700 lines.

[9] Chroot is a mechanism in UNIX whereby a process is irrevocably confined to a single branch of a filesystem. Once the chroot is performed on a process, the restricted branch of the filesystem is treated as its root directory. This mechanism makes it easy to prevent access to device files or files such as the password file.

Domain Name Service (DNS)

The name service software available for UNIX implements an in-memory read-only database. As such, it cannot be used to gain unauthorized access to a system. Past attacks on firewalls have used name service spoofing as a technique for impersonating trusted network hosts. In order to remove the threat of name service spoofing, the firewall does not rely on name service for any security related information. The name server software is necessary for high performance large-scale mail systems and is configured so that the only application that relies on name service for addressing is the electronic mail system.

FTP

The FTP application gateway is a single process that mediates FTP connections between two networks. Since it performs no disk access other than reading its configuration file and is a small and relatively uncomplicated program, it can be proven that it is not capable of compromising the security of the system. Just to be certain, the application gateway runs as a non-privileged user, after being "chrooted" to a private directory on the system. To control FTP access, the application gateway reads a configuration file, containing a list of FTP commands that should be logged, and a description of what systems are allowed to engage in FTP traffic. All traffic can be logged and summarized. Optionally, the gateway can permit FTP traffic from the Internet to the campus network for users who first authenticate themselves to the system.

TELNET

The TELNET application gateway is a small, simple application that mediates TELNET traffic. As with the FTP application gateway, the only file accessed is the configuration file that is read at start-up. Immediately after the configuration file is read, the TELNET application gateway is "chrooted" to a restricted directory, where it runs as a non-privileged process. The TELNET gateway's configuration file allows specification of which systems or networks can use it, and what systems or networks it will permit connection to. Initially, it will be configured to permit campus systems to use the gateway to connect to Internet systems, but not vice-versa. Optionally, the TELNET gateway can require strong authentication before permitting use. All connections and their durations are logged.

TCP Access and Use

On BSD-based UNIX systems, most network processes are started up by an initial connection to a general-purpose network listener `inetd`, which establishes a connection between the incoming request and the program to service the request. For example, an incoming request for the TELNET service is "heard" by the running network listener. The program, according to `inetd`'s configuration file and the entry for TELNET, is executed and connected to the incoming request.

`Inetd`, the internet services daemon, performs no function other than to invoke specified processes to manage network services when a system attempts to connect to them. Some vendor implementations permit a systems administrator to specify the user-id that the service should be invoked as, but there is no provision for limiting access based on the source of the request. A variety of implementations of "wrapper" processes is available on the Internet [8] with varying functionality.

The toolkit uses a "wrapper" process called `netacl`, which provides support for all TCP-based services. (If only TCP-based services are supported, UDP services are disabled and are no longer a threat worth worrying about.) `Netacl` has no great advantages over other versions of TCP wrappers, other than its minimal size (240 lines of code, including a large copyright header and comments), its lack of support for UDP (purposely), and its sharing a common configuration mechanism with the other tools in the toolkit. We believe that these differences provide a significant security advantage.

TCP Plug-Board Connection Server

Certain services such as Usenet news are often provided through a firewall. In such a situation, the administrator has the choice of either running the service on the firewall machine itself or installing a proxy server. Since running news, for example, on the firewall exposes the system to any bugs in the news software, it is safer to use a proxy to gateway the service onto a "safe" system on the campus network. Plug-gw is a general purpose proxy that "plugs" two services together transparently. Its primary use is for supporting Usenet news, but it can be employed as a general-purpose proxy if desired. Plug-gw is configurable, as are the other proxy servers. Since it only acts as a data pipe, it performs no local disk I/O and invokes no subshells or processes. Like the other proxy servers, it logs all transactions.

UDP

Since we decided that no direct traffic would be permitted between an outside system and an inside system, and since UDP is connectionless and point-to-point (and so cannot be used through network proxies), UDP services are not allowed.

User Authentication

The network authentication server authd provides a generic authentication service for network applications. Its use is optional, required only if the firewall FTP and TELNET proxies are configured to require authentication. Authd's purpose is to provide a generic interface to multiple forms of authentication. For large organizations, where several forms of authentication challenge/response cards are in use, authd can link them all together to use a single database. A simple administrative shell is included that permits the authentication database to be manipulated over a network, with optional support for encryption of authentication transactions. The authd database supports a basic form of group management; one or more users can be identified as the administrator of a group of users, and can add, delete, enable, or disable users within that group. Authd internally maintains information about the last time a user authenticated to the server and how many failed attempts have been made. It can automatically disable accounts that have multiple failures. Extensive logs are maintained of all authd transactions. Authd is intended to run on a secured host, such as the bastion host, since its database is a possible point of attack.

OBSERVATIONS

Securing a network's perimeter is an interesting exercise in tradeoffs. Since many useful tools (e.g., encrypting terminal emulators and PPP servers that use specific authentication systems) are not available in off-the-shelf form, a certain amount of software development is required. A whole range of interoperability problems is encountered and must be overcome. Hard choices must be made as to whether to solve problems by replacing code, fiddling with configurations, or persuading users to change their habits. Often none of these choices is appealing.

Implementing security-related software designed to resist intrusion requires organized thought if one wishes to have any confidence in the security of the result. During implementation, several design principles emerged:

- Statement of Mission. Most importantly, one should draw up a clear statement of mission before beginning to design one's solution. Assumptions should be stated, and design goals should be identified. This includes formulating a list of risks to defend against, ranging from things that absolutely cannot be permitted to happen, down to things that are interesting but can be ignored. Each identified risk should have a solution proposed for it, even if the solution is "We'll ignore that one." Risk analysis often brings to light issues that are easy to overlook. For each solution that is proposed, a means of verifying the solution's correctness should also be presented; otherwise it's hard to tell if one has protected oneself or simply muddied the waters. Problems must be addressed at a global level. Deciding "What am I trying to protect and protect against?" or "What am I trying to do?" produces a better solution than starting with the components: "How do I secure my Internet link? How do I secure my dial-in lines?"

- Keep It Simple, Stupid (KISS). If the source code for a security-related program is large enough and complex enough that it cannot be checked over in a couple of minutes, it's too complicated to be secure. There are many packages for UNIX systems that control security-related information that are entirely too feature-laden to be maintainable or trustworthy. Also, if you have a reasonable overall goal, it is easier to articulate, implement, and administer.

- Assurance. It should not be comforting to trust your network security to software that has been "hammered on enough that there aren't likely to be any more bugs." Employ configuration practices such that, even if your software has bugs, an attacker cannot use it to compromise the system. An example of this approach is causing the SMTP listener to run "chrooted," without permissions, so that even if someone manages to find a hole in it, they are trapped in an isolated compartment on the system.

- Minimize Risk. If a network service is disabled, it can't hurt you. Your security is better if you shut everything off and turn it on bit by bit than if you try to run around and shut off only the services that are known to be dangerous. This means, in a nutshell, that you must accept that what you don't know *can* hurt you. The other alternative leaves you in an "arms race" against your potential attackers.

- Verify. When you postulate that you have shut down all unnecessary network services, have a procedure in place to verify that, in fact, it is the case. While this may seem like a nebulous task, it is really fairly easy.

In keeping with a simple solution, acceptable failure modes are defined and a means of assuring that those failure modes will be met is developed.

While this project was motivated by the requirements of the Executive Office of the President, the work we have done can be used by any organization that wishes to secure their external computer communications. We have provided security with easy-to-use remote access to and from a protected network, while protecting data from disclosure, in an environment using strong user authentication. Further, many of these methods can be employed in an environment that already has mechanisms in place for addressing some of these concerns. Others, such as the Firewall Toolkit, can be used independently of the other methods discussed in this paper, but should only be done in conjunction with a strong security policy, which has thoroughly examined security threats and concerns and mapped out other counter-measures.

ACKNOWLEDGMENTS

This work was done under a contract from the U. S. Department of Defense, Advanced Research Projects Agency (ARPA), number DABT 63-92-C-0020.

REFERENCES

1. James M. Galvin and David M. Balenson, "Security Aspects of a UNIX PEM Implementation," Proceedings of the 3rd USENIX UNIX Security Symposium, September 1992.

2. National Computer Security Center, "Trusted Network Interpretation of The Trusted Computer System Evaluation Criteria," July 31, 1987.

3. Phil Karn, Neil M. Haller, and John S. Walden, Bellcore, S/Key software kit, available via anonymous ftp from `thumper.bellcore.com: /pub/nmh/skey/*`.

4. Stephen T. Kent, "Internet Privacy Enhanced Mail," Communications of the ACM, August, 1993.

5. Marcus J. Ranum, "Thinking About Firewalls," Proceedings of Second International Conference on Systems and Network Security and Management (SANS-II), April, 1993

6. Winfield Treese and Alec Wolman, "X Through the Firewall, and Other Application Relays," Cambridge Research Lab Technical Report 93/10, Digital Equipment Corporation, May 3, 1993.

7. Bill Cheswick, "The Design Of a Secure Internet Gateway," Proceedings of the 3rd USENIX Security Symposium, September 1992.

8. Wietse Venema, Department of Math and Computing Sciences, Eindhoven University of Technology, The Netherlands. USENET Archives, `comp.sources.misc, Volume 20`, August 1991.