ULTRIX

digital

Security Guide for Administrators

Order Number: AA–PBKTA–TE

ULTRIX

Security Guide for Administrators

Order Number: AA-PBKTA-TE
June 1990

Product: ULTRIX Version 4.0 or higher

digital equipment corporation
maynard, massachusetts

Contents

About This Manual

1 Introduction to Computer Security

2 Providing Physical Security

3 Controlling System Access: Managing User Accounts

4 Tracking Activity on the System: Auditing

5 Protecting the File System

6 Protecting Systems in Networks

7 Starting and Configuring a Secure System

Tables

About This Manual

This guide describes the tasks and considerations associated with managing system security in an ULTRIX environment.

Audience

This guide is written for persons responsible for establishing and implementing system security policies on workstations or hosts with the ULTRIX operating system.

To gain the most from this guide, you should have a working understanding of the basics of ULTRIX system administration. This guide presents information about system security administration; it does not cover the larger topic of ULTRIX system administration.

Before reading this guide, you should be familiar with file permissions, the use of the `chmod` command, and the security relevance of executables that are SUID. For information on these topics and on security features available to the general user, see the *Security Guide for Users*.

Although this guide discusses issues related to physical security and assigning trust to personnel, it is beyond the scope of the guide to describe details of physical security arrangements and related building design; neither does the guide detail methods for security screening of personnel. The subject of disaster planning and recovery is not covered in this guide. These subjects are addressed in publicly available industry and professional publications.

Organization

This guide consists of seven chapters and an index.

Chapter 1	Introduction to Computer Security Describes the role of the security administrator, basic aspects of computer security, and factors to consider when establishing a system security policy. Presents a template for security policy.
Chapter 2	Providing Physical Security Presents guidelines for protecting physical access to your computer system and peripherals.
Chapter 3	Controlling System Access: Managing User Accounts Describes the user accounting system and how you control who can log in to the system and what privileges users have.

Chapter 4	Tracking Activity on the System: Auditing Presents the role of auditing in maintaining security, suggests security-relevant events to audit, describes the tools for performing audits, and tells how to interpret audit results and follow-up on them.
Chapter 5	Protecting the File System Identifies files and directories with special security relevance and recommends appropriate protections for them.
Chapter 6	Protecting Systems in Networks Presents suggestions for protecting systems that are part of larger computer networks, such as DECnet or TCP/IP.
Chapter 7	Starting and Configuring a Secure System Discusses security considerations for system initialization and how to configure system security.

Related Documents

You should have available the hardware documentation for your system and the other documents in the current ULTRIX documentation set.

In particular, you should have the *ULTRIX Security Guide for Users* and the *Guide to System Environment Setup*.

For information about networking and managing user accounts in a distributed environment, the following are helpful:

Guide to Networking
Guide to Kerberos
Guide to BIND/Hesid Service
Guide to Yellow Pages Service

Conventions

The following conventions are used in this guide:

`user input`	This bold typeface is used in interactive examples to indicate typed user input.
`system output`	This typeface is used in interactive examples to indicate system output and also in code examples and other screen displays. In text, this typeface is used to indicate the exact name of a command, option, partition, pathname, directory, or file.
UPPERCASE lowercase	The ULTRIX system differentiates between lowercase and uppercase characters. Literal strings that appear in text, examples, syntax descriptions, and function definitions must be typed exactly as shown.
rlogin	In syntax descriptions and function definitions, this typeface is used to indicate terms that you must type exactly as shown.

filename	In examples, syntax descriptions, and function definitions, italics are used to indicate variable values; and in text, to give references to other documents.
#	A number sign is the default superuser prompt.
[ ]	In syntax descriptions and function definitions, brackets indicate items that are optional.
{ | }	In syntax descriptions and function definitions, braces enclose lists from which one item must be chosen. Vertical bars are used to separate items.
`cat`(1)	Cross-references to the *ULTRIX Reference Pages* include the appropriate section number in parentheses. For example, a reference to `cat`(1) indicates that you can find the material on the `cat` command in Section 1 of the reference pages.
RETURN	This symbol is used in examples to indicate that you must press the named key on the keyboard.

Introduction to Computer Security 1

This chapter describes the basics of computer security and how to go about formulating a security policy for your installation. Topics covered include the following:

- Role of the security administrator
- Purpose of computer security
- Tools of computer security
- Security Policy

1.1 Role of the Security Administrator

The role of security administrator can vary widely from site to site. At many sites the system manager, in addition to other duties, is responsible for implementing system security; at other sites someone will have security administration as his or her sole function. Elsewhere, it might be a system operator, under direction of the system manager, who directly implements security policy. Often management, in consultation with the system manager, formulates security policy, or it might be determined by a corporate entity far removed from the site where the policy is implemented.

This book uses the term "security administrator" as though it refers to a single person who is responsible for developing security policy and implementing it. Defining the term this way is useful because managing system security requires considering a wide range of system functions, regardless of who performs those functions.

Functions this book ascribes to the security administrator include:

- Determining security policy
- Adding new users to the system and assigning them privileges.
- Configuring those aspects of the file system that are relevant to security
- Assigning permissions to security-relevant system files
- Performing and evaluating security audits
- Making arrangements for physical security

Because the security administrator performs such functions, it is necessary that he or she have superuser status.

1.2 The Purpose of Computer Security

In simplest terms, the purpose of computer security is to protect the following:

- The data stored on the system

- The processing power of the system
- Continued access by legitimate users to the data and processing power they are authorized to use

1.3 Tools for Computer Security

The security administrator has the following methods to protect the computer system:

- Physical access controls
- System-enforced access controls
- Auditing

For effective computer security, you must use all three methods. The description of how you use these methods and security guidelines for the user community should appear in a security policy document for your site.

This chapter briefly introduces the security tools: physical access control, system-enforced access controls, and auditing, all of which are discussed in depth later in the book. This chapter describes security policy at length.

1.3.1 Physical Access Controls

Physical access controls provide perhaps the most effective method of protecting a computer system. Controlling physical access entails the following:

- Controlling access to the computer system and supporting hardware, including disk drives, tape drives, and printers.
- Controlling access to hardware that communicates with the system, such as terminals, workstations and remote I/O devices.
- Controlling access to storage media, such as CDROMs, disk platters, floppy disks, and tape reels.

1.3.2 System-enforced Access Controls

System-enforced access controls are those supported by the software and hardware. The ULTRIX operating system offers the following system-enforced controls:

User authentication process

The login process requires the would-be user to identify himself or herself to the system as a legitimate user before further dialog with the system is allowed.

Separation of privileges

Once successfully logged in to the system, the user is granted superuser status or simple user status according to need and trust as determined by the security administrator.

File permissions

The user can selectively control who can read, write, or execute those files owned by the user.

Process inheritance of access privileges

When a user causes a process to execute, that process can inherit the access privileges associated with the user and can pass on those inherited privileges to other processes it initiates.

1.3.3 Auditing

With auditing, you can track what happened on the system, when it happened, and who caused it to happen. You can also know about failed attempts to log in to the system.

With the ULTRIX auditing subsystem, you can choose to audit only those events appropriate to the security of your system. Later, when you examine the audit log, you have available an audit tool that enables you to select audit records according to particular characteristics. For example, you can choose to examine all file opens and creations initiated by a specific process.

With proper auditing you can discourage attempts at illegitimate use of the system, adjust security policy to strengthen system protection, and respond appropriately when security violations occur.

1.4 Security Policy

A security policy comprises the procedures and regulations needed to maintain a desired level of system security. Some policies provide you and the system manager with guidelines for the configuration and day-to-day management of the system. Other guidelines address the user community, suggesting "safe" practices for system usage. Before you can determine an appropriate security policy, you must identify the level of security appropriate for your system.

When determining the level of security required by the system, keep in mind that security has a cost, in terms of the ease with which legitimate users can access data and, in the case of auditing, in system response. This cost must be weighed against the possible cost of a system break-in, should one occur.

To determine security policy for your system, you must

- assess the sensitivity and value of system resources; that is, the sensitivity and value of the data stored on your system and the value of the system's processing power
- assess the level of trust that can be placed in those with access to the system and the level of their access needs

1.4.1 Assessing the Value of System Resources

When determining the value of the data and processing power of your system, the following are questions to consider:

- How damaging would it be if an unauthorized user gained access to data on your system?
- How critical would it be if data were corrupted or erased?
- How expensive would it be if legitimate users were denied use of the system because someone had maliciously brought it down?

1.4.2 Assessing User Trust and Access Needs

While it is beyond the scope of this book to address the subject of screening personnel, it is important for you to determine how much trust to place in the community of system users. The following are factors to consider:

- Are all users carefully screened?
- Are there any users from remote sites whose trustworthiness is largely unknown?
- Is there a high rate of turnover among system users?

If all the users of the system are highly trusted, then a high level of system security can be achieved with a relatively relaxed security policy. On the other hand, if there are many users whose backgrounds are unknown, or if there are many temporary users or users from remote sites, then it is prudent to enforce controls on physical access, perform frequent and detailed audits, carefully manage passwords, and restrict access to superuser status.

1.4.2.1 Assigning Superuser Status – Among the users at your site there are some who might require superuser status. Such users might be operators or system programmers.

Anyone with superuser status has complete access to all files and and resources on the system. A superuser can read, write, or remove any file (although an encrypted file is not likely to reveal any information). Superuser activities are logged, but a superuser can alter or remove log files, so a malicious user with superuser status can cover his or her tracks.

Note

Grant superuser status only to users who are highly trusted and who require the status to do their jobs.

1.4.3 Setting Security Policy

Once you have assessed your system's security needs and the level of trust you place in the users, you can write a security policy. It is useful to divide the security policy into two parts, according to how the policy is upheld:

- Security policy upheld by the system security features
- Security policy upheld by people: the administrators, operators and users of the system

The first part is fairly easy to put into practice. It involves such things as file permissions, which protect sensitive files; the auditing subsystem, which lets you track user activities on the system; and the user authentication file, which controls who can sign on to the system.

Implementation of the second part, security policy upheld by people, is more difficult. You will need to convince the system users that security is important, and educate them as to how they can do their parts. Periodically, you will have to repeat the process.

To this end, it is helpful to distribute a security document to every user and operator on the system. Topics discussed should include the following:

- The need for security
- The benefits of security to all system users
- Required security-relevant behavior (such as logging out at the end of every day)
- Suggested security-relevant behavior (such as choosing safe passwords)

The tone of the document should be positive, with the emphasis on the advantages of a secure system. The document must make clear who a user should notify if the user suspects a security violation has occurred or is being attempted.

1.4.4 Security Policy Checklist

Consider the following items for inclusion in your site policy statement:

- Personnel Policies
  - Policies for users
  - Policies for operators
  - Security training
- Auditing Considerations
  - Frequency of audits
  - Depth of audits
  - User access to audit records
- Physical Security
  - Secure environment for system consoles, disk drives, tape drives, and printers (when used to print sensitive material)
  - Controlled access to the operations center
  - Procedures for visitors and maintenance personnel in the operations center
  - Secure storage of backup media
  - Secure environment for workstations
  - Follow the guidelines presented in Chapter 2
- User Authentication
  - Minimum password length
  - Password expiration
- File System Security: follow the guidelines presented in Chapter 5.

- Network Security
  - No default user accounts for systems on DECnet
  - No anonymous FTP configuration for system on TCP/IP network.
  - Set guidelines for the contents of users' `.rhosts` files
  - Follow the guidelines in Chapter 6

1.4.5 Reevaluating Security Policy

As users and conditions on your system change, reevaluate and adjust security policy. Factors that should alert you to the need to reassess policy include:

- Changes in the make-up of the user population; for example, an increase in the number of users whose backgrounds are unknown or an increase in the number of remote users.
- Changes in the kinds of files on the system, in particular, an increase in files with sensitive data.
- Changes in software that might open new security holes, such as a new utility that runs as root and that allows shell escapes.
- The discovery of attempted system break-ins, or of attempts or successes at other security violations.

Providing Physical Security 2

This chapter discusses the physical protection of system resources. The focus is on protecting those resources from unauthorized access, theft, or deliberate damage by users or vandals. The chapter does not deal with protecting your system against natural disaster nor fire, nor with disaster recovery. Information on those topics can be found in professional journals and industry publications.

The major subjects covered in this chapter are

- Protecting the computer room
- Controlling access to resources outside of the computer room

U.S. Government agencies and those doing business with the government usually must adhere to specific physical security guidelines. For more information, refer to *Guidelines for Automatic Data Processing Physical Security and Risk Management, Federal Information Processing Standards Publication (FIPS PUB) 31*, which is available from the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161.

2.1 Protecting the Computer Room

Often, the central processor units, disk drives, tape drives, optical storage systems, system consoles, and printers are housed in a central location, referred to here as the computer room. The concentration of resources in the computer room makes it an attractive target for those who would violate system security. Fortunately, because these resources are located in one area, it is easier for you to defend them against attack.

2.1.1 Computer Room Contents

Because of their sensitivity to attack, the following should be housed in a secure environment:

- The central processing units
- System consoles
- Disk drives
- Tape drives
- Optical disk systems
- Printers used for printing sensitive material

2.1.2 Controlling Access to the Computer Room

To control access to the computer room, do the following:

1. Permit only authorized personnel regular access to the computer room.
2. Keep an up-to-date list of all the members of the computer room staff.
3. Record computer room entrance and exit times for all regular staff members.
4. Escort those who are not part of the regular computer room staff.
5. Maintain a log of the admittance of all non-staff members, such as visitors and maintenance personnel. Record name, reason for admittance, escort's name, and start and end-time of the visit.
6. Train the computer room staff to challenge the presence of anyone who does not appear to belong in the computer room.

As an aid to identifying those with authorized access to the computer room, it is useful to maintain a locked, wall display case containing photographs of the computer room staff and others authorized to be in the computer room.

2.1.3 Handling Storage Media

The physical bulk of items such as printers and tape drives protects them against easy misappropriation, but storage media are more vulnerable, because they are more portable. To protect the information on storage media, such as tapes and disks, you must prevent unauthorized removal of the media from the secure environment. Archival media, such as backup tapes, should be stored in the the computer room or in a location protected at a level at least equal to that of the computer room.

Take the following precautions with storage media:

1. Compress then encrypt all sensitive files on storage media. Use the `compress` and `crypt` commands for this purpose. Doing this lessens the chances that useful data can be read from a stolen tape or disk. For more information on these commands, see `compress`(1) and `crypt`(1) in the *ULTRIX Reference Pages*.
2. Allow only authorized personnel to handle and transport storage media. Keep a log of all storage media brought in to or removed from the computer room. Information logged should include the date and time in or out, the carrier's name, the origin or destination of the media, and a description of its content.
3. Render unreadable any sensitive information on storage media, such as tapes or floppy disks, before a technician is allowed to handle the media, unless the technician is trusted with the content of the media. You can protect sensitive information by compressing then encrypting the contents of the media, or by scrubbing the data completely.

2.2 Controlling Access to Resources Outside the Computer Room

In general, all printers and terminals should be protected against unauthorized access. Printer output and screen displays are potential sources of information about file names, account names, system activities, new product planning, marketing strategy, and more.

Controlling Access to Terminals

An unguarded terminal is a potential entry point into the system. Even when your site has a policy requiring users to lock or halt sessions when they are away from their terminals, users who think they are stepping out for just a minute or two often will violate the policy. When this happens, only physical security prevents anyone from using the terminal to access the system.

If only trusted users have access to the building or room housing terminals, it might not be necessary to protect each terminal individually; you trust the users not to violate security by using a terminal left in an active session under a fellow employee's account. However, it is safer to protect terminals on a terminal by terminal basis.

Controlling Access to Workstations

Workstations often require special consideration for physical security; not because they are inherently less secure than other systems, but because workstations are typically found in ordinary offices, not in the more easily protected environment of the computer room.

In many cases, anyone who gains access to a workstation can easily get superuser status on that system. One method is simply to boot the system into single-user mode.

The vulnerability of workstations means that to protect them you must provide them with the same level of physical security given the systems in the computer room. The simplest approach might be to place workstations only in offices with locking doors and to require all workstation users to lock their offices whenever they are away from their systems. Alternately, you can ensure that only trusted users have access to buildings or areas where workstations are located.

Workstation users must also protect their tape cartridges and floppy disks. Two steps can be taken to protect files on removable storage media:

- Compress then encrypt files with the `compress` and `crypt` commands. Should an unauthorized person gain access to a disk or tape cartridge, this step makes it difficult for the data to be read.
- Lock up all removable storage media when they are not in use.

Controlling System Access: Managing User Accounts 3

This chapter describes how you manage user accounts, which determine who can login to the system, and various attributes associated with user accounts. The major subjects covered are

- Controlling access to the system
- Overview of the ULTRIX user account system
- Managing accounts with the user account system
- Adding or Modifying Distributed Accounts in UPGRADE or ENHANCED Level
- Suggestions for managing user accounts
- Suggestions for the contents of the user-environment files
- Managing user accounts with the traditional BSD system

For a discussion of establishing a user account system in a network environment and the BIND/Hesiod, Kerberos, Yellow Pages features, see Section 6.5.

3.1 Controlling Access to the System

The user account system is the most important security feature on your system. It controls who can log in to the system and, once logged in, the privilege a user enjoys.

3.1.1 User Identity

The key to controlling login to the system is user identity. A user account system holds the names of those people allowed to log in, defining each user's identity on the computer system. The attributes of user identity are as follows:

user name	The name a user enters in response to the login prompt. Each user name represents a user account on the system.
user ID (UID)	A number that is the "real" user identity as far as the system is concerned. The UID identifies the user for purposes of determining file and process ownership. It is possible (but not recommended) for two different user names to have the same UID. In this case, both users will own exactly the same files.
audit ID	A number unique to the user that is employed to track the user's activity on the system.

group ID (GID) A number that becomes the GID for processes and files created by the user. GIDs establish group-level identity for files and processes. A description of how to use GIDs to allow users file access at a group level appears in Section 5.3.

For more information about UIDs and GIDs, refer to the *Security Guide for Users.*

3.1.2 The Login Procedure

The user logs in to the system in order to establish his or her identity with the system and thereby gain system access. In response to the login prompt, the would-be user enters a user name.

At this point, the question is, "How does the system know that the would-be user really is who he or she claims to be?" This is where the password comes into play. The password is assumed to be known only to the authorized user, and therefore the password functions to verify that the person attempting to login actually is the legitimate user.

After entering a user name in response to the login prompt, the user is prompted for a password, if a password is required for the user. The would-be user gains access to the system only if the user name corresponds to an account in the user account system and if the password matches the password for that account.

While a valid user name is necessary to access the system, the user name is not considered protected information. In the work place, the user name often acts as the public identity of the user. For example, user names are often displayed on printer listings to identify who owns which listing. Because the user name can be widely known, it has little or no security value. It is the password that verifies the user's identity, and so it is vital that the password remain secret.

3.1.2.1 Adding a Warning to the Login Prompt – A message can be added to the login prompt that requests the user name at the start of the login procedure. It is recommended that you use this feature to display a message strongly discouraging unauthorized access to the system.

The message can be added to the login prompt by altering the file `/etc/gettytab`, a file containing information about terminal configuration. A message such as "UNAUTHORIZED ACCESS PROHIBITED" is recommended. This can be done by adding the following to the `/etc/gettytab` file.

```
default:\

     :ap:fd#1000:\
     :im=\r\n\r\n(%h)\r\n\
UNAUTHORIZED ACCESS PROHIBITED \
\r\n\r\r\n\r:\
     :sp#1200:
```

For more information about the `/etc/gettytab` file, see `gettytab`(5) in the *ULTRIX Reference Pages.*

3.1.2.2 Restricting Superuser Login – The `/etc/ttys` file contains information used by the system to initialize terminal special files and control their use. Each line in the file describes a terminal. Superuser access at a terminal is fully supported only if the `secure` flag appears in the terminal description.

If the system is in UPGRADE or ENHANCED level, then a user can neither login as root nor gain superuser status through the `su` command at a terminal unless that terminal is marked with the `secure` flag. If the system is in BSD level, then a user cannot log in as root at a terminal unless the `secure` flag is set. However the user can use the `su` command to gain superuser status. Because superuser status is necessary to manage a system, make sure that at least one terminal always allows superuser access.

If the system console is physically secure, as should be the case, it is permissible to set the `secure` flag for the console port. No other terminals should have the `secure` flag set. For a workstation, only the line for the X Window System server, `:0` (and, if the system has two displays, `:1`), should have the flag set.

For more information about the `/etc/ttys` file, see `ttys`(5) in the *ULTRIX Reference Pages.*

3.1.2.3 Providing Users with Trusted Path – In a secure environment, it is important that the user have full confidence that when a prompt on the terminal requests highly sensitive information, the prompt is genuine and not the result of an illicit program designed to gather sensitive information. The classic example of this is the password prompt displayed when a user logs in. How can a user know whether the prompt is genuine or part of a fake login program attempting to learn the user's password?

The ULTRIX operating system offers a feature to assure the user of a trusted path between the user's terminal and the legitimate login process. By means of the secure attention key, the user can establish this trusted path.

The BREAK key functions as the secure attention key. To invoke trusted path, a user simply presses the BREAK key then RETURN. The trusted path mechanism then kills all processes running on the terminal from which it was invoked, and presents the user with the login prompt.

The trusted path feature is available only if you configure your system for it. Trusted path is not supported for pseudo ttys. If you enable trusted path, then the BREAK key does not provide a means of escaping to the LAT terminal server. However, there is a way a user on a system with trusted path can work around the limitation. For example, a user can define CTRL/A as the new terminal server attention key by entering the following commands at the prompt for the local terminal server:

```
set port <BREAK> remote
set port local <CTRL/A>
```

Then to escape to the LAT terminal server, that user need only type CTRL/A

To enable trusted path, select the feature from the security setup script. The security setup script, `/usr/etc/sec/secsetup`, is an interactive script used to configure system security. For more information about this script and selecting a security configuration, see Section 7.2.

3.1.2.4 Disabling Logins – By creating the file `/etc/nologin`. you deny the ability to log in to all users except the root account. Any text in `/etc/nologin` is displayed as a message to users who attempt to log in.

Note

If you create `/etc/nologin`, make sure that at least one terminal allows superuser login (is marked `secure` in the `/etc/ttys` file). Otherwise you might find yourself without superuser access to the system. This could force you to reboot the system to single-user mode in order to regain superuser access.

To prevent users from logging in right before a system shutdown, `/etc/nologin` is created automatically by the `shutdown` command, five minutes before the actual shutdown takes place.

For more information about the file `/etc/nologin`, see `shutdown`(8) in the *ULTRIX Reference Pages*.

3.2 Overview of the ULTRIX User Account System

In traditional BSD UNIX,† the mechanism for managing user accounts is the `/etc/passwd` file. Each line of the file defines a user account, including the user name, the encrypted password, and other user information. The security administrator employs the `adduser`, `removeuser`, and `vipw` commands to keep the accounts in the file up to date.

There are a number of areas where the security provided by the traditional BSD user account system could be improved.

- There is no enforced minimum length for passwords. Although a user of the `passwd` command who trys to change a password to one shorter than six characters is asked to "Please use a longer password," if the user persists and re-enters the short password, the system accepts it on the third entry.
- The maximum length for a password is eight characters. With the `passwd` command a user can enter a new password that is longer than eight characters, but the system records only the first eight characters.
- There is no software mechanism to ensure that users regularly change their passwords.
- There is no software mechanism to ensure that users choose "safe" passwords. A user can choose any password, including one that is easily guessed, such as the first name of the user or the first name of the user's spouse.
- Although `/etc/passwd` is writable only by a user with superuser status, the file must be readable to all users and processes. Because of this, there is always the threat that a password will be decrypted, allowing an unauthorized user to log in on the system.

The ULTRIX operating system offers an improved user account system, with additional account information stored in an authentication data base. The ULTRIX system provides the following improvements over the BSD system:

- A minimum length for passwords can be designated.
- Passwords up to sixteen characters long are supported.
- A password aging feature is available that causes passwords to expire after a designated period. To prevent users from replacing old passwords with new

† UNIX is a registered trademark of AT&T in the USA and other countries.

ones and then immediately changing back to their old, familiar passwords, a minimum life for passwords can also be designated.

- A password generator is available that generates passwords meeting guidelines set by the National Computer Security Center of the U.S. Government. At the security administrator's discretion, users can be required to use system-generated passwords.
- The user authentication data base, which stores passwords, is readable and writable only by those with superuser status and members of the group `authread`. The passwords in this data base are encrypted by a method providing a high degree of security. There is no facility for decrypting the passwords.

The ULTRIX operating system offers the optional use of these ENHANCED security features and supports them by means of the user authentication data base, which is used in addition to `/etc/passwd` to store information about user accounts

3.2.1 The Traditional Password File, /etc/passwd

The file `/etc/passwd` provides the traditional BSD mechanism for managing user accounts.

A typical `/etc/passwd` file might have entries like the following:

```
root:KM6Ft1UP9NX1w:0:1:System PRIVILEGED account,,,:/:/bin/csh
operator:Gqd2AWV0WAXfE:999:28:Ops,649,ms 14C,:/operator:/ops/choose.dump
smith:Cpt4vdFVZdoEA:274:15:Smith,122,ms 2D,487-9223:/usr//smith:/usr/bin/csh
doe:2g2sa8FA.ToQ.:1001:15:Doe,374,ms 11G,463-6558:/usr/users/doe:/bin/tcsh
```

Each line represents a user account. Within each line, colons separate fields containing user attributes. A single line has the fields arranged like this:

```
name:password field:UID:GID:user info:initial working directory:shell program
```

The attributes of the user account are as follows:

- Name

  The user login name.

- password field

  An encrypted password.

- UID

  A number uniquely identifying the account.

- GID

  A number identifying the login group of the user.

- User info

  Text intended to associate the user name with a real person. At a minimum, the text should be the user's actual name. If the text lists the user's name, office number, extension, and home phone number each separated by a comma from the next, then commands such as `finger` can retrieve that information.

- Initial working directory

  The name of the working directory the user is in immediately after login.

- Shell program

  A program, usually a standard UNIX shell, that is executed at the time the user logs in

Note

Make certain that `/etc/passwd` is owned by root and has permissions 644.

In order to function correctly, certain utilities require read access to `/etc/passwd`. Because passwords in `/etc/passwd` are encrypted, the practice of giving read access to group and other is considered safe.

3.2.2 The User Authentication Data Base

In order to provide enhanced account security features, the ULTRIX operating system offers an option that stores an encrypted password and other account security information in an authentication data base. The authentication data base works in conjunction with `/etc/passwd` to support the ULTRIX user account system. This data base is readable only by those with superuser status and members of the group `authread`, unlike `/etc/passwd`, which is readable by group and other. The passwords in the authentication data base can be as much as sixteen characters long, and the data base also stores information needed to support such features as password aging and auditing on a per-user basis.

The authentication data base comprises two files, `/etc/auth.pag`, which stores user accounts, and `/etc/auth.dir`, which is a directory into the data base entries. A user account record can be retrieved and displayed in ASCII format with either the `getauth` or `edauth` commands. When one of these commands is used, the user account is displayed with the entries separated from each other by colons. The ASCII format of a typical user account in the authentication data base appears as follows:

```
1005:EHS95wCWlyCNo7aXOwKzMu3U:630875971:0:604800:07:0:1005:00:00:00
```

A description of the fields in the account record follows. For a description of how to assign values to the fields, see Section 3.3.1.

1. UID

   The UID corresponds to the UID in `/etc/passwd` and serves to uniquely link the user entry in `/etc/auth.pag` with the user entry in `/etc/passwd`.

2. Encrypted password

   The user's actual password can be up to 16 characters long. This password is encrypted, and there is no facility for decrypting the passwords.

3. Password creation/modification time

   The value in this field supports the password expiration feature. The value indicates the number of seconds between January 1, 1970, and the time when the current password was chosen.

4. Minimum password lifetime

   The value in this field determines the minimum amount of time a password must be in use before a user is allowed to choose a new password. The value is the minimum password lifetime in seconds.

5. Maximum password lifetime

   The value in this field determines the maximum amount of time a password can be in use before it expires and can no longer be used to log in to the system. The maximum password lifetime is listed in seconds.

6. Account mask

   This field lets you control aspects of the user's login privileges. The value in the field grants the user one or more of the following abilities:

   - the ability to login
   - the ability to change passwords
   - the ability to choose a password, rather than having to use a system-generated password

7. Login failure count

   This field holds a record of the number of consecutive failed attempts to log in. Whenever a user successfully logs in, the system displays the number of failed login attempts since the last session. The value in this field is reset to zero whenever the user successfully logs in. If five consecutive login attempts fail, a warning is recorded in the file `/usr/spool/mqueue/syslog`. Refer to Section 4.9.1 for information about this file.

8. Audit ID

   This field supports the ULTRIX auditing system. It stores a number unique to the user. The number is used to track the user's activity on the system.

9. Audit control flag

   This field supports the per-user auditing feature of the ULTRIX audit system. It contains a flag indicating how the user audit mask is applied to the system audit mask to determine the events logged for the user in the `auditlog` file.

   Events designated in the user audit mask can be logged in addition to those events designated in the system audit mask, or they can be deleted from the list of events designated in the system audit mask, or they can be logged only if the same events also appear in the system audit mask. These logging variations apply only to the individual user.

10. User audit mask

    This field is used in conjunction with the user audit mask field to support per-user auditing. It stores a value that designates logging specific events for the user. The affect this value has on logging for the user is determined by the value in the field for the audit control flag.

11. This field is not currently used but is reserved for future use.

Note

Make certain that `/etc/auth.pag` and `/etc/auth.dir` are owned by root, have permissions 640, and belong to the group `authread`.

These permissions protect the contents of the authentication data base while allowing read-access to the data base by those utilities that require access.

3.2.2.1 More about Expired Passwords – If you are using the password expiration feature, you can specify a grace period during which the user can log in with an expired password. When the user logs in during this grace period, the system requires the user to change passwords before allowing him or her to continue the session.

At the end of the grace period, if the user has not changed to a new password, the user can no longer log in to the system, and someone with superuser status must reactivate the user's account by invoking the `password` command along with the user name of the frozen account.

Starting five days before a user's password expires, every time the user logs in, the system displays the following message:

```
Password expires in n days
```

In place of *n*, the system displays the actual number of days left until the password expires. If the password will expire in fewer than 24 hours, then the system displays the following message:

```
The password will expire very soon
```

For a description of setting the grace period for expired passwords, see Section 3.3.4.

3.2.3 Security Level

You can choose one of three security levels for your system:

- BSD level, the traditional UNIX user-authentication features.

  User accounts are stored entirely in `/etc/passwd`. The file is maintained with the standard commands.

- UPGRADE level

  Information for each user account is stored in the authentication data base, but passwords can still be maintained in `/etc/passwd`. When a new account is created, it is entered into the authentication data base. All the authentication data base features are available, such as expiring passwords and password generation.

  This level serves to ease your move from the traditional BSD UNIX user account system to the ULTRIX user authentication system. It allows you to continue using `/etc/passwd` as the location of passwords for existing user accounts and gradually build up account entries in the authentication data base.

  If your system is part of a distributed environment, you can use BIND/Hesiod with Kerberos to help you securely manage user accounts across the network. However, UPGRADE level does not support Yellow Pages.

  On a system in UPGRADE level, the entry in the password field in `/etc/passwd` influences the location of the user's password as indicated in the following table:

Table 3-1: Effect of Password Entries When in UPGRADE Level

Content of Password Field in /etc/passwd	Effect on Location of User's Password
Text other than an asterisk (*)	The password resides in `/etc/passwd`.
An asterisk	The password resides in `/etc/auth.pag` of the user authentication data base.
The field is empty	The password resides in `/etc/passwd`. Never allow this situation to occur. When the password field is empty, anyone can log in under the user name without having to enter a password and then choose a password for this account.

- ENHANCED level

  All passwords are stored in the `/etc/auth.pag` file of the user authentication data base. The full range of features of the user authentication system, including expiring passwords and password generation, are available for all user accounts. If your system is part of a distributed environment, you can use BIND/Hesiod with Kerberos to help you securely manage user accounts across the network. However, ENHANCED level does not support Yellow Pages.

You determine the security level when you configure the security features on your system with the security setup script. A description of this process appears in Section 7.2.

3.3 Managing Accounts with the ULTRIX ENHANCED User Account System

As security administrator, you perform the following tasks associated with user accounts:

- add new accounts
- update existing accounts
- remove old accounts
- assign a mandatory minimum length for passwords and assign a grace period for expired passwords

The first three tasks must be performed on a continuing basis, as new users come to the system, other users move off the system, and changes in the system environment need to be reflected by changes in some user accounts.

The final task is performed infrequently, perhaps only once every several years.

Your primary tools for maintaining the user account system are the following commands:

- `adduser`
- `removeuser`
- `edauth` (for systems in UPGRADE or ENHANCED level)
- `vipw` (for systems in BSD level not running Yellow Pages)

3.3.1 Adding New User Accounts

Use the `adduser` command to add new user accounts on the system. The command adds users to both the `/etc/passwd` file and, if appropriate, the user authentication data base. To invoke the command, enter the following:

```
# /etc/adduser
```

The command runs interactively, and enables you to assign values to all the relevant entries for the user account, such as a group name and password expiration attributes. The command also sets up an initial working directory for the new user, with the files `.login`, `.profile`, and `.cshrc`, and with the `bin` directory. For users of `csh`, the `.login` file executes when the user first logs in and configures aspects of the user's environment. For users of `sh`, `sh5`, `ksh`, the file `.profile` executes when the user first logs in. The file `.cshrc` contains commands executed whenever `/bin/csh`, the C shell, is invoked.

The `adduser` command interface is the same regardless of the security level of your system. If you configured the system for UPGRADE level or ENHANCED security level, the account becomes part of the authentication data base. The `adduser` command automatically makes the appropriate entries in both the `/etc/passwd` file and the authentication data base. If you configured the system for traditional BSD UNIX level, the command acts only on `/etc/passwd`.

When you execute the `adduser` command, you are presented with a list of the user attributes that make up a user account. The command enables you to assign values to most of these attributes and automatically assigns values to some others. If your system is configured to use the traditional BSD UNIX account system, only those entries appropriate for `/etc/passwd` are displayed.

The `adduser` command asks you the following:

- Enter login name for new user (initials, first or last name):

  Enter the user name, a string up to eight characters long. A good user name can be formed by concatenating the user's first and middle initials with the user's last name. For example, the name Jane Francis Wilson becomes the user name `jfwilson`. If the resulting string is more than eight characters long, drop the necessary number of characters from the end of the user's last name.

- Enter uid for new user [ ]:

  The area between the square brackets is pre-filled with a default UID calculated by incrementing by 1 the value of the largest current UID. If you choose to, you can overwrite the default UID and enter the user ID of your choice. If you enter an invalid UID, you are warned of that fact and instructed to enter a different value. If you enter a UID that already exits, you are warned of that fact and given an opportunity to choose a new UID or to confirm that you want duplicate UIDs.

- Enter full name for new user:

  Enter the user's full name and other identifying information about the user. Typically this includes the office number, work and home phone number; any information that might be useful to your work with the user and the account. If the text lists the user's name, office number, extension, and home phone number each separated by a comma from the next, then commands such as `finger` can retrieve that information.

- What login group should this user go into [users]?

  Enter the name that determines the default GID for processes and files created by the user. The default group name is "users." Files and processes that originate with this user will have that GID associated with them. The GID is used in determining group ownership of files and processes. A description of GIDs appears in the *Security Guide for Users*. A description of groups and how you create them appears in Section 5.3.

  If the group name you enter does not exist, you are informed of that fact, told the available group names, and asked whether you want to create the new group. You are also given the opportunity to add the user to more than one group.

- Enter parent directory for the user [/usr/users]:

  Enter the name of the user's parent directory. The `adduser` command creates a new directory consisting of the parent directory plus the user's login name. This directory of parent plus login name is the user's working directory when he or she first logs in to the system. The default parent directory is `/usr/users`. For example, J.F. Wilson's initial working directory might be `/usr/users/jfwilson`.

  If the parent directory does not already exist, you are asked whether you want the `adduser` command to create the directory. System management can be simplified by giving all users the same parent directory.

- Enter the user's login shell name [/bin/csh]:

  The system displays a list of suggested login shells based on the contents of the file `/etc/shells`. You are then asked to enter the name of the login shell for this user. You can enter one of the displayed shells or any available shell. The default login shell is `/bin/csh`. If you wish to, you can specify a program that supports a restricted environment. For more information about restricted environments, see Section 3.5.1.

If your system is running with the UPGRADE or ENHANCED security level, then you will also be asked to fill in the following information:

- Enter maximum password lifetime in days [60]:

  Enter the maximum number of days the user is allowed to have a password before being forced to change to a new password. The default is sixty days.

  A value of 0 means the password never expires.

- Enter minimum password lifetime in days [0]:

  Enter the minimum number of days the user must have a password before being allowed to change to a new password. The default is zero.

  A value of 0 allows the user to change the password at any time. If you specify a minimum lifetime greater than the maximum lifetime, then the password can

be changed only by someone with superuser status.

- Will machine generated passwords be required [no]?

  Enter **Y** to require this user to choose a password from a list of machine-generated passwords. The default is that generated passwords are not required.

At this point, you are asked whether you want to edit the `/etc/auth` entry for the user.

- Do you wish to edit the auth file entry for this user [no]?

  If you answer **Y** , then the `edauth` command is automatically invoked and you can modify the user's account attributes in `/etc/auth`. For information about using the `edauth` command, see `edauth`(8) in the *ULTRIX Reference Pages* and Section 3.3.2.1.

After you have completed editing the user entry, you are returned to the `adduser` program. You are informed you that until the password is set for the user, the user cannot log in. The `adduser` command then terminates and automatically invokes the `passwd` command. You now enter a password for the new account. If you do not enter a password at this opportunity, the user's account remains unusable until you invoke `passwd` and choose a password for the account.

For more information about the, `adduser` command, see `adduser`(8) in the *ULTRIX Reference Pages.*

When the `adduser` command is invoked, it automatically enters the following information in the authentication files:

UID

The `adduser` command automatically assigns a unique UID for the user. ULTRIX software is shipped with a `root` account with a UID of zero already defined in `/etc/passwd`.

If the system is in UPGRADE or ENHANCED security level, then root is the only account permitted to have superuser status (a UID of `0`).

If the system is in BSD level, then you can assign multiple accounts a UID of `0`. Remember that for new accounts added while a system is in UPGRADE or ENHANCED level, entries are made in the authentication data base, in the file `/etc/auth.pag`, and the passwords are stored in that file

Audit ID

For users on systems at UPGRADE or ENHANCED level, the `adduser` command automatically assigns audit ID the same value as the UID.

3.3.2 Updating User Accounts

You use one of two commands to modify existing accounts, `edauth` or `vipw`. The command you use depends on which aspect of a user account you want to modify.

Additionally, you can use the `chsh` command to change the user's login shell and the `chfn` command to change the user identity information in `/etc/passwd`. See `chsh`(1) and `chfn`(1) for more information.

3.3.2.1 Updating User Information in the Authentication Data Base – The `edauth` command provides an easy-to-use interface to the authentication data base. The command invokes the editor designated by your `EDITOR` environment variable. If no editor is designated, the command invokes the `ed` editor. The syntax for the command is as follows:

/usr/etc/sec/edauth *user_name*

If you configured the system for traditional BSD level, then use the `vipw` command, not `edauth`, to modify all aspects of existing user accounts.

Use `edauth` to modify entries in the `/etc/auth.pag` file of the authentication data base. These entries are:

- User ID (UID) (Note: if you change the UID in the data base, use the `vipw` command to change the corresponding UID in the `/etc/passwd` file.)
- Minimum password lifetime (`passlifemin`)
- Maximum password lifetime (`passlifemax`)
- Account mask (`authmask`)
- Audit ID (`audit_id`)
- Audit control flag (`audit_control`)
- User audit mask (`audit_syscalls, audit_tevents`)

When you invoke the `edauth` command, a list of the account attributes for the user is presented to you for editing. The list will look similar to the following:

```
uid = 1027
password = SDt7cPT2d/xWsaugseBiC58A
passlifemin = 4 hours
passlifemax = 60 days
passmod = 6/27/90 - 11:15:46
authmask = login,change_password,enter_password
fail_count = 0
audit_id = 1027
audit_control = or
audit_syscalls =
audit_tevents = auth_event,login:0:1
```

Typically you rarely, if ever, use the `edauth` command to change the values for `uid`, `passmod`, `fail_count`, and `audit_ID`. The `audit_ID` and `uid` are assigned for you by the `adduser` command when the account is created. The value of `passmod` is updated automatically whenever the password is changed. The value of `fail_count` is automatically incremented with each successive login failure by the user. To change the password, use the `passwd` command, not the `edauth` command.

Note that the notation for designating minimum and maximum password lifetime is different with `edauth` than it was with `adduser`.

- `passlifemin` (minimum password lifetime)

  Enter the minimum amount of time the user must have a password before being allowed to change to a new password.

  Your entry can specify any combination of days, hours, minutes and seconds in any order. Only the first letter of a time unit need be supplied.

For example, to specify 2 days, 8 hours, 30 minutes, 25 seconds, you can enter

```
passlifemin = 2 d 8 h 30 m 25 s
```

To specify 1 day, 30 minutes, you can enter

```
passlifemin = 1 day 30 minutes
```

If you enter a number with no day, hour, minute, or second label, then seconds is the assumed unit of time.

The default is 0 (zero). A value of 0 allows the user to change the password at any time. If you specify a minimum lifetime greater than the maximum lifetime, then the password can be changed only by someone with superuser status.

- `passlifemax` (maximum password lifetime)

  Enter the maximum amount of time before the password expires. As with the minimum password lifetime, you specify any combination of days, hours, minutes and seconds in any order, and only the first letter of a time unit need be supplied. For example, to specify the password expires in 45 days, you can enter

  ```
  passlifemax = 45 d
  ```

  If you enter a number with no day, hour, minute, or second label, then seconds is the assumed unit of time.

  The default is 60 days. Enter `0` (zero) if you do not want the password to expire.

- `authmask` (account authorization mask)

  Enter one or more of the following values to grant the user the indicated privilege.

login	Allow user to log in.
change_password	Allow user to change password.
enter_password	Allow user to choose own password, rather than requiring use of a system-generated password.

  If more than one entry is made, use a comma to separate each entry. For example `login,change_password`

  If the user is required to have a password generated by the system, then whenever the user changes passwords (with the `passwd` command), the system presents a list of generated passwords, and the user selects one. Even when you do not require the user to have a generated password, the user has the discretion to use one. The `-a` option of the `passwd` command will supply the user with a list of generated passwords.

Two entries you do not have an opportunity to set with the `adduser` command are the audit control flag and the user audit mask. This is because it is anticipated that for most user accounts you will not exercise the feature supported by these entries: per-user specification of events to be logged. If you want this feature, use the `edauth` command to set the audit control flag and user audit mask.

- `audit_control` (audit control flag)

  Enter one of the following to indicate how the user audit mask is applied to the system audit mask to determine the events logged for the user in the `auditlog` file.

  or Designates that the events logged for the user are those specified by either the system audit mask or the user audit mask. Suppose the system mask specifies logging chmod, chown, exit, open, and close, and the user mask specifies logging chown, open, mount, kill, and rename then all the events, chmod, chown, exit, open, close, mount, kill, and rename are logged for the user.

  and Designates that events logged for the user are only those events that are specified by both the system audit mask and the user audit mask. So in the case of the preceding example, only chown and open are logged for the user.

  off Designates no logging is done for this user.

- `audit_syscall, audit_tevent` (user audit mask)

  Designate user-specific event logging using the same syntax you used with the `auditmask` command to designate system-wide event logging. Events entered in a user's individual audit mask are combined with the system audit mask to determine which events are logged for that user.

  A list of auditable system calls and trusted events appears in Section 4.7. For a description of selecting audit events for the system audit mask, see `auditmask`(8) in the *ULTRIX Reference Pages*.

  The syntax for specifying the user audit mask is

  event:*success*:*failure*

  In place of *event,* enter the name of the event. For the audit_syscalls field, enter an auditable system call. For the audit_tevents field, enter a trusted event.

  In place of *success,* enter

  1 to designate logging the event when it succeeds.

  0 to designate no logging of the event when it succeeds.

  Similarly, in place of *failure,* enter

  1 to designate logging the event when it fails.

  0 to designate no logging of the event when it fails.

  Entering just the event with no success or failure specification is the same as specifying both success and failure. That is `login` has the same effect as `login:1:1`.

  You can specify multiple events by listing individual event specifications separated by commas. For example, to specify failed login attempts, successful and failed file opens, and successful file closes, you would enter the following:

```
login:0:1,open,close:1:0
```

  Note that the default for *success* or *failure,* is not 1. There is no default for these individual specifications. That is open::0 is not the same as `open:1:0`.

Your options are to enter an event with no specification for *success* or *failure,* for example, `chown`, in which case both successes and failures are logged, or to specify what you want for both *success* and *failure,* for example `exit:1:0`.

For more information about the `edauth` command, see `edauth`(8) in the *ULTRIX Reference Pages.*

3.3.2.2 Updating User Information in /etc/passwd – Regardless of the security level of your system, always use the `/etc/vipw` command to update the following entries in the `/etc/passwd` file:

- User name
- User ID (UID) (Note: if you change the UID in `/etc/passwd`, and you have user account information in the authentication data base, use the `edauth` command to change the corresponding UID in the data base.)
- Group ID (GID)
- User information text
- The name of the user's working directory when the user first logs in
- The name of the program executed when the user logs in

The `vipw` command command invokes whatever editor is specified by your `EDITOR` environment variable. If no editor is specified, the command invokes the vi editor.

You must use `vipw` rather than some other editor for three reasons:

- During the editing session, the command locks the file to prevent any possible conflicts from outside attempts to alter or read it. Any processing delayed by the locks resumes when the editing session terminates.
- The command checks the consistency of the root account, and if errors are found, prevents the flawed `/etc/passwd` file from being installed.
- Because of the nature of `/etc/passwd` and its relationship with the authentication data base, unexpected results can occur if you use an editor other than `vipw` to modify the `/etc/passwd` file.

For more information, see `vipw`(8) in the *ULTRIX Reference Pages.*

3.3.3 Removing User Accounts

Use the `removeuser` command to delete user accounts. To invoke the command, enter the following:

```
# /etc/removeuser
```

The command is interactive, enabling you to select the particular user account for removal. The account is removed from both `/etc/passwd` and from the authentication data base, if the account has an entry in the data base.

When you remove an account, you are given the option of deleting the user's home directory and files, as well.

For more information, see `removeuser`(8) in the *ULTRIX Reference Pages.*

3.3.4 Assigning Minimum Password Length and Expired Password Grace Period

The file `/etc/svc.conf` contains system configuration information, including fields with security relevance. Fields that affect passwords directly are

PASSLENMIN The value after this field determines the minimum length allowed for passwords. The value shipped with the software is 6.

PASSLENMAX The value after this field determines the maximum length allowed for passwords. The value shipped with the software is 16. To maintain compatibility with password handling on some networks, you might want to restrict users to passwords no longer than 8 characters.

SOFTEXP The value after this field determines the length of the grace period during which a user with an expired password can continue to log in. After this grace period, the user is locked out of the system and will have to come to you to reactivate the account. If the password expiration feature is not enabled (either you are not using the ULTRIX user account system or the value is `0` for the maximum password lifetime in the user's account in `/etc/auth.pag`) then this field has no effect.

The length of the grace period is in seconds. The value shipped with the software is 604800 seconds, the equivalent of 7 days.

To assign a minimum length for passwords or to assign a login grace period for users with expired passwords, edit `/etc/svc.conf` and change the values for `PASSLENMIN` or `SOFTEXP`.

One other field in the file also affects security, `SECLEVEL`. For information about `SECLEVEL`, and how to change it, see Section 7.2. Because of the security relevance of `/etc/svc.conf`, the file should be owned by root and have permissions 644.

If the system freezes a user's account because the password expired and the user failed to choose a new password during the grace period, then to reactivate the account, use the `passwd` command with the user name of the frozen account.

3.3.5 Supporting Password Expiration on Workstations

In order to support password expiration on workstations, the entry for the workstation in `/etc/ttys` needs to include the `/usr/bin/login` command with the `-e` option. If more than one option is designated for `/usr/bin/login` the `-e` option must be the last option specified.

When you configure system security with the security setup script, password expiration is automatically enabled for those workstations listed in `/etc/ttys` at the time the setup script is executed. If you later add workstations to `/etc/ttys`, you will need to manually include the `/usr/bin/login` command with the `-e` option. For information about the security setup script, see Section 7.2.

3.4 Adding or Modifying Distributed Accounts in UPGRADE or ENHANCED Level

If the security level of systems in a network is UPGRADE or ENHANCED, use the following procedure to add or modify a user account that is supported by the

distributed system services (DSS) provided by BIND/Hesiod. This procedure must be carried out on the BIND/Hesiod primary server.

1. Add or modify the account in `/var/dss/namedb/src/passwd`. The format of each user account in this file is the same as the format in `/etc/passwd`:

   ```
   name:password field:UID:GID:user info:initial directory:shell program
   ```

   Because the system is at UPGRADE or ENHANCED security level, you can enter an asterisk (*) in the password field for the user account.

2. Add or modify the account in `/var/dss/namedb/src/auth`. This is an ASCII file with entries in the following order:

   - UID
   - encrypted password
   - password modification date; the number of seconds between January 1, 1970, and the current time when the password is chosen
   - minimum password lifetime (in seconds)
   - maximum password lifetime (in seconds)
   - authentication mask
   - login failure count
   - audit ID
   - audit control flag
   - user audit mask
   - currently unused; reserved for future use

   A colon separates each of these attributes, and the format of a typical entry appears as follows:

   ```
   1005:EHS95wCWlyCNo7aXOwKzMu3U:630875971:0:604800:07:0:1005:00:00:00
   ```

   This format is exactly the same as the format returned for an entry in `/etc/auth` by the `getauth` command. If the user account you want to add to `/var/dss/namedb/src/auth` exists in `/etc/auth` and the user's distributed account is to have the same account attributes as the local account, then you can add the user to `/var/dss/namedb/src/auth` by directing the results returned by the getauth command to the file `/var/dss/namedb/src/auth`.

   For example, if user smith has an account in `/etc/auth` and you want smith to have a distributed account with the same attributes as her local account, you can use the following command:

   ```
   # getauth smith >> /var/dss/namedb/src/auth
   ```

   A different strategy is to create a template entry in `/var/dss/namedb/src/auth`. For example,

   ```
   ${UID}:Nologin:${PASSMOD}:{MINPASSLIFE}:${MAXPASSLIFE}:${AUTHMASK}:0 \
   :${UID}:00:00:00
   ```

The variables can be set to defaults and the template can be copied each time a new user is added to /var/dss/namedb/src/auth. For example:

UID	The UID of the new user.
PASSMOD	The number of seconds between January 1, 1970, and the time current time when the password is chosen.
MINPASSLIFE	86400 (1 day).
MAXPASSLIFE	5184000 (60 days).
AUTHMASK	07 (allow user to login, to change password, and to choose own password rather than forcing user to have a system-generated password).

If you want to assign values to the audit control mask and user audit mask, do the following:

- Create the account locally in /etc/auth with the adduser command.
- Assign the local account an audit control mask and user audit mask with the edauth command.
- Use the getauth command to supply the user account information for the user's distribute account in /var/dss/namedb/src/auth.
- If you want the user served by the distributed account and not by the local account, remove the local account you just created with the command.

3. Rebuild the BIND/Hesiod passwd and auth databases as follows:

```
# cd /var/dss/namedb
# make NOPUSH=yes passwd
# make auth
```

4. If a new account was added, use the passwd command to assign it a password.

For information about getauth command, see getauth(8) in the *ULTRIX Reference Pages*. For information about user account attributes, see Section 3.2.2. For information about using the adduser and edauth commands to create and modify local user accounts, see Sections 3.3.1 and 3.3.2. For information about BIND/Hesiod, see the *Guide to BIND/Hesiod Service*.

3.5 Suggestions for Managing User Accounts

The following are recommended procedures for managing user accounts. Some recommendations assume you are using the ULTRIX user-authentication system.

- When a user will cease to need access to the system:
  - Before a user ceases to have an account on the system, request that the user clean up his or her files and directories.
  - Ensure there is a backup of the user's files.
  - Promptly remove the obsolete user account with the removeuser command.
  - Remove any references to the user in the /etc/group file.

- Inform the user community of the impending removal of the files associated with the defunct account, and allow users adequate time to copy needed files and make any necessary adjustments to absolute path names.
- Remove all the files from the user's directory. This can be accomplished with the following command:

  # **rm -rf** *user's_home_directory*

- Check for any files owned by the user but present in directories other than the user's. The following command will place the names of any such files in the file `check_file`. You can then review the contents of `check_file` and determine which files to remove and which files should have their ownership changed. Enter the following command to do this:

  # **find / -user** *user_name* **-print > check_files**

- Require generated passwords for all accounts.
- Give passwords lifetimes between 60 and 120 days long. Consider shorter lifetimes for the password of the root account.
- For those users requiring only limited access to the system, arrange for startup programs that create a restricted environment.
- Never leave the password field in `/etc/passwd` empty if your system is configured for traditional BSD UNIX level or UPGRADE level.
- Do not create a default user account, also known as a guest account. Typically `guest` is the user name of the account, and the word `nologin` appears in the password field. A local user cannot log in with such an account; however, the account allows any remote DECnet user to access your system. For more information about DECnet security, refer to Section 6.2.

3.5.1 Startup Programs

A startup shell, typically `csh`, is entered by `adduser` in the `/etc/passwd` file. The shell is then automatically executed whenever the user first logs in.

However, you might prefer to have the startup program be a special environment tailored for only that user, rather than a standard shell program.

For example, if you have a remote user whose only legitimate use of the system is to check on the latest information in a particular file, a script can be written with the following properties:

- When the script is executed, it places the user in a read-only editing session with the file to which the user requires access.
- The editor supports only movement through the file; access from the editor to any other files is not supported.
- The editor does not allow any shell escapes.
- When the editing session is terminated, the session is automatically terminated and the user is logged off.

You can designate this script as the remote user's startup program. This allows you to protect any other system resources against access by the user.

Startup programs that establish a restricted environment require careful design. These programs should:

- Allow no shell escapes.
- Allow no programs that allow shell escapes.
- Trap and ignore signals: SIGINT, SIGQUIT, and SIGHUP.

If you want to assign a user a startup program different from the one assigned by the adduser command, you can use the chsh command or the vipw command to change the specification for the login shell, which appears in the last field of the user account record in /etc/passwd. See chsh(1) and vipw(8) for more information.

3.5.2 Using a Password Data Base

By creating a password data base, you improve the speed with which processes can access information in the /etc/passwd file.

The mkpasswd command is designed to create a password data base from the contents of /etc/passwd. To use the command, enter the following:

```
# /etc/mkpasswd /etc/passwd
```

The result is the creation of the password data base, which consists of two new files, /etc/passwd.pag, which contains information from /etc/passwd, and /etc/passwd.dir, which is a directory into /etc/passwd.pag. The command does not affect the contents of /etc/passwd.

If the password data base exists, then a process needing user account information retrieves the information from the data base, rather than from /etc/passwd. Because of the structure of the password data base, processes can retrieve information more quickly from the data base than they can from /etc/passwd. The improvement is significant for systems with a large number of user accounts.

Note that use of the password data base is really a system performance issue rather than a security issue. The password data base files, /etc/passwd.dir and /etc/passwd.pag should be owned by root and have permissions 640.

For more information about the mkpasswd command, see mkpasswd(8) in the *ULTRIX Reference Pages.*

3.6 Suggestions for the Contents of User-Environment Files

When you use adduser to create a new user account, the command also creates two files in the user's working directory, .profile and .login. These files determine characteristics about the environment the user operates in. Which of the files is used depends on the user's login shell, but there are certain security considerations common to both files. Because .profile and .login affect the security of the user's environment, the permissions for these files should be 640. If it is necessary for other users to have read access to the files, then the permissions can be 644.

The guidelines presented here will help create a secure environment for the user. Because users can later modify the .profile and .login files you create for them, it is important that you share these guidelines with the user community. To be effective, the suggested entries for .profile and .login must appear before any program names in those files.

- Make the first entry a line setting the PATH shell variable. This shell variable determines the paths searched by the system when the user invokes a command.

  It is important to the security of the users' accounts that users invoke only those commands they intend to invoke. If the system searches an unsecure path for a command, and an illegitimate program has been placed in that path, the security of the user's account is at risk. Because of this threat, it is important you take the following precautions when setting the PATH shell variable:

  - Make all pathnames absolute; explicitly spell out every pathname.
  - Specify only secure paths. Do not specify open or temporary directories such as /tmp or /usr/tmp.
  - Do not specify the working directory (.). Users often use the cd command to change their working directories, sometimes to unsecured paths. If this happens, then their accounts might be at risk.
  - In the list of paths, place the directories for system files first.

  For example, in .profile the PATH variable can be set as follows:

  ```
  PATH=/usr/bin:/usr/ucb:/usr/local/bin:$HOME/bin
  ```

  Note that in .profile, colons separate path specifications. Two consecutive colons or a leading or trailing colon will result in the working directory being specified as a path. You do not want this to occur.

  In the .login file, the same paths are set for the PATH variable as follows:

  ```
  set path=(/usr/bin /usr/ucb /usr/local/bin $HOME/bin)
  ```

  For the convenience of the user, create a directory $HOME/bin for the user, and designate this as one of the directories to search. This allows the user to place private executable programs in the $HOME/bin directory and invoke them by typing the name of the executable.

- Make the second entry in the user-environment file the command

  ```
  mesg -n
  ```

  This prevents others from sending messages to the user's terminal. You want to deny this ability because workstations and intelligent terminals can sometimes be programmed by such messages.

- Make the third entry the command

  ```
  umask 027
  ```

  The result of this line is that each time the user logs in, his or her umask is set to 027. Files created during the session have at most permission 750 (user=rwx, group=rx, other=no access), although more restrictive permissions are still allowed. The user retains the option of later using the chmod command to assign more or less restrictive permissions to files.

- Any alias command names should reference the absolute pathname of the actual command. For example,

  ```
  alias drc /usr/bin/dircmp
  ```

A typical `.profile` should have the following entries (the actual paths designated in the `PATH` equation might differ for your system):

```
PATH=/usr/bin:/usr/ucb:/usr/local/bin:$HOME/bin
mesg -n
umask 027
```

A typical `.login` file looks different because the syntax for setting the `PATH` shell variable is different. Assuming the same paths as in the previous example, the

```
set path=(/usr/bin /usr/ucb /usr/local/bin $HOME/bin)
mesg -n
umask 027
```

3.7 Managing Users Accounts with the Traditional BSD System

Because of the added security features of the ULTRIX user account system, it is recommended that you use that system for managing user accounts. However, because the security administrators at some sites might decide to forego the added security offered by the ULTRIX user account system, the traditional BSD UNIX user account system is also available.

Select `BSD` on the security setup script if you want only the traditional user account system. When you select `BSD`, user accounts, including passwords, are stored in `/etc/passwd`, and you use `adduser`, `removeuser`, and `vipw` to manage the user accounts.

A description of the security setup script appears in Section 7.2.

Tracking Activity on the System: Auditing 4

This chapter describes the purpose of system auditing, how auditing is performed, what activities should be audited, and how to read and respond to audit reports. Topics covered here include the following:

- Overview of auditing
- The audit subsystem in brief
- Selecting events to audit
- Creating log entries
- Activating the audit subsystem
- Filtering audit information
- Suggested audit events
- Reading audits
- Responding to audits
- Traditional auditing tools
- Archiving log files

4.1 Overview of Auditing

Auditing provides you with a powerful tool for monitoring activity on the system. Through auditing, you can accomplish the following:

- Discourage users from attempting to violate security.

  A user who knows that system activities are monitored and that security violations can be tracked to the responsible individual might be dissuaded from attempting "illegal" actions.

- Detect attempts at violations or activities indicative of probing a system for weak points, and prevent their expansion into actual violations.

  If an audit reveals failed attempts to violate system security, you can take countermeasures to lessen the likelihood of later attacks succeeding.

- Assess damage and restore the system if a break-in should occur.

  Careful analysis of an audit trail after a break-in can help you determine what occurred during the security violation and what steps are needed to return the system to its original state. It also allows you to take steps to prevent similar break-ins in the future.

It is important that you inform users of the purpose and, in general terms, the nature of the auditing performed on the system. Represent auditing in a positive light, as a tool to help protect the users' files and their access to system resources. This helps

minimize any resentment: users who are openly told that their system is regularly audited are less likely to feel as though they are being spied upon. For those users who might be tempted to violate security, knowledge that activities are monitored can be a powerful deterrent.

4.1.1 Auditing Tools

The tools for auditing on ULTRIX systems can be divided into two categories:

- Traditional auditing features, such as the system accounting files and the `last` command.
- The audit subsystem, which has powerful features unique to the ULTRIX operating system.

While this chapter devotes a section to the traditional auditing tools, the chapter's primary subject is the audit subsystem, which is better for performing security-relevant auditing.

In general, auditing tools comprise the following parts:

- Log files, which contain records of events that occurred on the system.
- Event selection tools, which allow you to choose the system activities that are recorded in the log.
- Audit reduction tools, which allow you selectively to extract information from the log.

The ULTRIX audit subsystem records activities in the file `/usr/adm/auditlog.`*nnn*, where *nnn* is the generation number of the file, a number between 000 and 999. The subsystem is capable of recording a wide range of events. You can choose a list of events to log for all users and then add or remove events from that list on a per-user basis to tailor the logging to individual users. A special tool enables you to filter the audit log file and produce reports focusing on the information you need.

4.1.2 The Log Files

Information about events on the system is stored in a number of different files. The following table lists those files storing security-relevant information. The first entry is the log file for the audit subsystem. The other entries apply to traditional audit tools. The specific log files present on your system depends on which auditing and accounting features you have enabled.

Table 4-1: Log Files and Security-relevant Contents

File Name	Security-Relevant Information
`/usr/adm/auditlog.###`	log file for audit subsystem
`/usr/adm/acct`	raw system accounting data, including user commands
`/usr/adm/usracct`	reduced system accounting data
`/usr/adm/savacct`	reduced system accounting data
`/usr/adm/wtmp`	successful logins, logouts, shutdowns and reboots
`/usr/adm/eventlog`	DECnet events

Table 4-1: (continued)

File Name	Security-Relevant Information
/usr/adm/shutdownlog	system shutdowns and reboots
/usr/adm/sulog	failed and successful attempts to login as superuser
/usr/spool/mqueue/syslog	failed logins, requests for DECnet file transfers

Because these files are your record of what has happened on the system, their contents should be protected. The files and directories should be owned by the root account and they should not be writeable by group or other. Because some information in these files might be considered confidential, you might also want to restrict read access to the files.

Note that on many systems the directory /usr/adm is linked to /var/adm, and /usr/spool is linked to /var/spool. In such cases, you must protect the files under the /var/adm and /var/spool directories. Refer to Chapter 5 for a list of files to protect and appropriate permissions for those files.

Log files will continue to grow over time and can become quite large. To avoid problems with shrinking free disk space and to maintain auditing on your system, it is important that you periodically archive the various log files. For more information about archiving log files, see Section 4.10

4.2 The Audit Subsystem in Brief

The ULTRIX audit subsystem provides preselection of events to be logged, a high degree of choice in the kinds of events to be logged, flexible data reduction of the audit log, and ease of maintenance. The commands used with the audit subsystem are:

auditmask	Selects events for inclusion in the audit log or displays a list of events currently being recorded in the audit log.
audgen	Provides you with the ability to generate a log record containing a message of your choice. You can do this either from the command line or shell environment.
auditd	Activates auditing and allows you to configure the audit subsystem.
audit_tool	Selectively extracts information from the audit log and presents it in readable form.

Use of the preceding commands is limited to those with superuser status. Although these commands are discussed in greater detail later in this chapter, it is useful here to suggest a procedure to follow with the audit subsystem.

1. Evaluate your auditing needs and decide on a list of events and system calls to be logged. You might have more than one list if you are logging different events at different times.
2. Create one or more files of events to be audited. If you have only one list of events to be logged, then all you have to do is edit the file /etc/sec/audit_events, which is a file containing a list of all auditable events. Edit the file to reflect the logging that is appropriate for your site.

If you plan to regularly change the list of events that are audited, you will want to prepare one audit event file for each type of auditing you plan to perform. You can do this by making copies of `/etc/sec/audit_events`, and editing each copy to specify auditing of the appropriate events.

3. Set the system audit mask by directing to the `auditmask` command your file of events to be logged. For example, if you edited `/etc/sec/audit_events` and want to use that file to determine what events are logged, enter

```
# /etc/sec/auditmask < /etc/sec/audit_events
```

4. Set the user audit mask in the file `/etc/auth.pag` to enable special logging for individuals. The `edauth` command is used to set the user audit mask. See Section 3.3.2.1 and the reference page for `edauth`(8) for more information.
5. Activate the audit subsystem and configure it to your needs with the `auditd` command.
6. Periodically use the `audit_tool` to extract data from the audit log.
7. Evaluate audit reports and respond to them appropriately.

4.3 Selecting Audit Events

Two features enable you to specify logging of events:

1. With the `auditmask` command, you assign values to the system audit mask, which determines the events that are stored in the audit log for all users.
2. With the user audit mask, you can specify logging events for individual users.

The system audit mask and the user audit mask for the particular user determine the events logged for that user. The way the two masks are combined is controlled by the user's audit control flag. The user audit mask and the user audit control flag are part of the user's account and are stored in `/etc/auth.pag`.

The user audit mask and audit control flag enable you to designate events to be logged for the user in addition to those that are logged for all users, or to selectively exclude the user from the logging of certain events. A description of setting the user audit mask and audit control flag appears in Section 3.3.2.1.

A discussion of audit events appears later in this chapter, in Section 4.7. For a complete description of selecting audit events for the system audit mask, see the reference pages for `auditmask`(8).

The syntax of the `auditmask` command is as follows:

/etc/sec/auditmask [*event_specification*] [-f] [-n]

The *event_specification* lets you designate the event to be logged and whether only successful occurrences of the event, only failed occurrences, or both are to be recorded. An *event_specification* has the following form:

event:*success*:*failure*

In place of *event,* enter the name of the event you want audited. In most cases, an audit event corresponds to an ULTRIX system call.

In place of *success* enter

1 if you want the event logged when it succeeds.

0 if you do not want the event logged when it succeeds.

Similarly, in place of *failure,* enter

1 if you want the event logged when it fails.

0 if you do not want the event logged when it fails.

If you don't specify either success or failure logging, then both are logged. That is `login` has the same effect as `login:1:1`.

You can specify multiple events by listing individual event specifications separated by spaces. For example, to specify failed login attempts, successful and failed file opens, and successful file closes, you would enter the following:

```
# /etc/sec/auditmask login:0:1 open close:1:0
```

Note that the default for *success* or *failure,* is not 1. There is no default for these individual specifications. That is `open::0` is not the same as `open:1:0`. Your options are to enter an event with no specification for *success* or *failure,* for example, `chown`, which is the same as specifying both success and failure, or to specify what you want for both *success* and *failure,* for example `exit:1:0`.

Once an event has been designated for logging, logging of the event continues until you explicitly turn it off by specifying

<*event*>:0:0.

Use the `-f` option to enable logging of all auditable events on the system (both successes and failures). Use the `-n` option to disable all logging.

In most cases you will pass the auditmask command a file that specifies the events to be audited. If you are logging different sets of events at different times, you will want to create multiple event files, one for each of those different sets of events. Each of these files can then be passed to the `auditmask` command at the time you want to begin logging that set of events.

Auditing after System Reboots

When you use the security setup script to configure system security, it alters your `rc.local` file to automatically provide auditing whenever the system boots to multiuser mode. The security setup script copies the following lines to your `/etc/rc.local` file:

```
/etc/sec/auditd -z -o b & echo -n ' auditd'
/etc/sec/auditmask < /etc/sec/audit_events
```

The first line invokes the audit daemon. The `-z` option cleans up after any previous audit daemons. The `-o b` option specifies that the system is to be shut down if an audit log overflow condition occurs. Refer to Section 4.5 for more information about the `auditd` command and its options.

The second line modifies the system audit mask according to the contents of the file `/etc/sec/audit_events`. When `/etc/sec/audit_events` is first shipped to you, it specifies auditing of all events. You can edit `/etc/sec/audit_events` to reflect the audit specifications you feel are appropriate to your site.

4.4 Creating Your Own Log Entries

By using the `audgen` command, you can include a message of your choosing in the audit log. The command builds an audit record consisting of user-supplied text and the audit ID, UID, pid, IP address, and timestamp. The syntax for the command is as follows:

/usr/etc/sec/audgen *text_for_auditlog*

For example, if you wanted to annotate the log file, you might want to create an entry like the following:

```
# /usr/etc/sec/audgen "Begin recording file opens by J. Doe."
```

If you do not enclose the audit log text in quotation marks, then each word in the text will appear on a separate line when the text is later retrieved by the audit tool.

To log the event, `audgen8` must be one of the events designated for auditing with the `auditmask`. To extract records logged with the `audgen` command, include `audgen` in the list of events to be retrieved by the `audit_tool` command.

4.5 Activating and Configuring the Audit Subsystem

To activate the audit subsystem, use the `/etc/sec/auditd` command. In addition to activating the audit subsystem daemon, this commands lets you configure aspects of the subsystem that relate to the audit log.

The syntax for the command is as follows:

/etc/sec/auditd [*options*]

Some of the features and the options used to invoke them are described here. Refer to the reference pages for `auditd`(8) for a complete description of the command and its options.

Display help information

Use the `-h` option to display a menu of brief help information.

Display information about the audit subsystem

There are two options to inform you about the audit subsystem.

- To learn the location of the `/etc/auditlog` file, use the `-q` option.
- To learn the current options specified for the audit daemon, use the `-?` option.

Designate the location of the `/usr/adm/auditlog` file

To designate the local pathname to which audit data is written, use the `-l` option followed by the full pathname you want for the audit log. The file name should be of the form `auditlog.`*nnn*, where *nnn* is a number between 000 and 999, inclusive. The audit daemon normally appends the number automatically, incrementing the number by one each time a new audit log is generated. If you specify a generation number outside of the range 000 and 999, the audit daemon treats that number as part of the file name and appends a generation number after it.

With the `-i` option, you can specify the name of a remote host running the audit daemon. The audit data of the local system is then passed to the daemon running on the remote host.

Designate a fall-back destination for logging information

Because audit information is central to maintaining security, it is important to ensure an uninterrupted flow of information to the audit log. The `-b` option followed by a pathname designates the pathname to which audit data is written if the current location becomes unavailable. This could occur if the current location is a remote host that becomes unavailable, or if the file system of the current location begins to fill up. It is prudent to designate a fall-back destination that is a different file system than the one for the current audit log.

Designate a destination for audit log status reports

To provide you with up-to-date information on the status of the audit log, the audit subsystem sends messages about such things as log overflow conditions and the roll-over of the current log file to a new file. Use the `-c` option followed by a pathname to specify a device or local file as the destination of those messages.

Protect against audit log overflow

The audit system enables you to protect against the possibility of lost audit information due to the log file filling all the available space of a disk partition. The `-o` option followed by a letter lets you specify an action to be taken by the system in the event of an overflow condition. Choices are

a Use the alternate audit log specified with the `-b` option.

b Shutdown the system.

c Switch to the filesystem owned by root with the most free space.

d Suspend auditing until space is made available.

e Overwrite the current log file. Use the `-f` option followed by a number between one and 100 to set the percentage at which an overflow condition is triggered. The number you specify is the percent of free space remaining on the current partition.

Establishing Auditing across a Network

If you have computers linked in a TCP/IP network, you can run the audit daemon on multiple systems and feed the information logged to a single system, the audit hub, for storage and analysis. This can be done as follows:

1. On the host that is to be the central collecting point for audit information, the audit hub, create the file `/etc/auditd_clients`. Each line in this file must have the name of a remote host that will be feeding audit data to the local audit daemon.

2. On the audit hub, enter the following command:

   ```
   # /etc/sec/auditd -s
   ```

   to enable the audit hub to receive audit data from audit daemons on remote hosts.

3. On each remote host, direct the audit data to the system that is the audit hub with the following command:

   ```
   # /etc/sec/auditd -i audit_hub_name
   ```

4. From the audit hub you can designate options for an audit daemon on a remote host by using the `auditd` command as usual to specify the options and by including the `-p` option to designate the ID of the remote audit daemon to receive the options. To learn the ID of a remote audit daemon, enter

   ```
   # /etc/sec/auditd -?
   ```

   It is suggested you use the `-a` option when auditing is done across a network. The `-a` option enables Kerberos authentication to verify the identities of the communicating audit daemons. This feature is available on networks running Kerberos. To avoid authentication problems, systems in such networks must have synchronized time clocks. For more information, refer to the *Guide to Kerberos*.

Note

When feeding audit data from remote hosts to an audit hub, direct the audit data from each remote host into its own, dedicated audit log file on the hub system. This is necessary to prevent corruption of audit data.

When you use the audit tool to retrieve data from these logs of audit data from remote systems, there is a chance that the first and last audit log entries will be fragments, rather than complete entries. This is due to the fact that remote audit information is fed in a continuous stream to the audit hub, rather than as discrete audit entries.

The audit tool notifies you when it encounters a fragmented entry. This does not affect the retrieval of other records from the audit log.

Turn off the audit subsystem

The `-k` option kills the audit daemon, disabling the audit subsystem and leaving you without the ability to track activity on your system. Avoid using this option during normal system operation.

Roll Over the current audit log

To start a new auditlog with the number of the current auditlog incremented by 1, use the `-x` option. When you use this option, the old audit log is automatically compressed by the `compress` command.

The name of an audit log file is always appended with a generation number. These generation numbers range from 0 to 999. (Generation numbers can be overridden with the `-l` option). If the current audit log is `auditlog.275`, the `-x` option creates a new file, `auditlog.276`, and begins writing audit data to it.

If the system shuts down abnormally, it might be necessary to clean up after previous audit daemons before a new audit daemon can start successfully. The `-z` option provides this feature. Typically, you will not use this option when you invoke the `auditd` command. The `-z` option is intended for use in the invocation of the audit daemon in `/etc/rc.local`.

Note

Use the `-z` option only when you are certain no audit daemons are running on the system.

As an example of how you might use the `auditd` command, assume you wanted to configure the audit subsystem as follows:

- The audit log is to reside in `/usr/users/secadm`.
- The alternate audit log is to reside in `/var/users/secadm`.
- When the file system for the current audit log becomes 90% full, a warning message is to be displayed at `/dev/console`, the system console, and the auditing data is to be routed to the alternate audit log.
- The current audit log is to be named `auditlog.100`

You can do all this with the following command:

```
# /etc/sec/auditd -b /var/usr/secadm -c /dev/console -f 10 -l \
/usr/users/secadm/auditlog.100 -o a
```

4.6 Filtering Audit Log Information

The `audit_tool` command enables you to reduce and filter information stored in the audit log and to display the audit information in a format you can read. The command handles both compressed and uncompressed files. If the audit log you apply the command to is compressed, `audit_tool` will uncompress the contents before extracting information and recompress the file when it is finished with the extraction process.

The syntax of the `audit_tool` command is as follows:

/usr/etc/sec/audit_tool [*option*] *filename*

Use *filename* to designate the audit log from which audit information is to be extracted.

Within a single option type, audit records are returned for each use of the option. For example, assuming the name of the audit log is `auditlog.100`, you can retrieve records of all file opens, closes, and renames, by using the `-e` option as follows:

```
# /usr/etc/sec/audit_tool -e open -e close -e rename /usr/adm/auditlog.100
```

When you mix different options, only audit records are returned that match the attributes specified for all the different options. For example, to get reports only on those file opens that are also associated with user name smith, you can use the following command:

```
# /usr/etc/sec/audit_tool -e open -U smith /usr/adm/auditlog.100
```

The command can be highly selective in the audit records it extracts. For example, the following command extracts only records of open, close or rename events that are associated with user name smith or brown.

```
# /usr/etc/sec/audit_tool -e open -e close -e rename -U smith \
-U brown auditlog.100
```

If you don't supply any arguments to the `audit_tool` command, a brief help message is displayed.

Some often-used features of the tool are described here. For a complete description of the command, see the reference pages for `audit_tool`(8).

Use `audit_tool` interactively

To run `audit_tool` interactively, use the `-i` option. Each option along with its current or default setting is then offered to you. You can enter your choice, or press RETURN to accept the current setting or default.

When in interactive mode, the default, unless stated otherwise, is to retrieve all audit records.

Select audit records for a particular user identity

There are three options that let you select records associated with user identity. The differences relate to the method used to define user identity.

- Select records by user name. The `-U` option enables you to specify one or more user names. Log entries for processes with user names that match your input are then returned. Note that a user name is associated with a log record only if `login` is logged; for this option to be useful, you must be logging the `login` event. The default is all user names.
- Select records by effective UID. The `-u` option enables you to specify one or more UIDs. Log entries for processes with effective UIDs that match your input are then returned. The default is all UIDs.
- Select records by real UID (RUID). The `-r` option enables you to specify one or more RUIDs. Log entries for processes with RUIDs that match your input are then returned. The default is all RUIDs.

  For a discussion of effective and real UIDs, refer to the *Security Guide for Users*.
- Select records by audit ID. The `-a` option enables you to specify one or more audit IDs. Log entries for processes with audit IDs that match your input are then returned.

  As with the user name, the audit ID is associated with a log record only if `login` is logged; You must be logging the `login` event in order for the audit ID to be useful.

Generate a report for each audit ID

The `-R` option causes an ascii report to be generated for each audit ID associated with the events being logged. All events with a common audit ID are are placed in a report with the name `report.`*n*, where in place of *n* , the actual audit ID appears.

Select audit records of events that occurred in a certain time range

Use the `-t` option followed by a time specification to designate a start time. Then only those records with a timestamp equal to or greater than that start time are selected. Use the `-T` option followed by a time specification to designate an end time. Then only records with timestamps equal to or less than that end time are selected. Use `-t` and `-T` together to limit the audit records retrieved to only those that occurred within a given period.

The format for start and end times is yymmdd[hh[mm[ss]]]. At most, only one start time and one end time can be specified. The default is to select for all timestamps.

Select audit records for specific events

Use the `-e` option to retrieve those audit records that match a list of designated events and their success/failure status.

The syntax for the event specification is

event[:*success*:*failure*]

In place of *event,* enter the name of the event for which you want to retrieve information. A list of all auditable events can be found in Section 4.7.2 and in the file `/etc/sec/audit_events`.

Event selection for retrieval is the same as event selection in the `auditmask` command. In place of *success* enter

1 if you want an audit report about the event when it succeeded.

0 if you do not want an audit report about the event when it succeeded.

Similarly, in place of *failure,* enter

1 if you want an audit report about the event when it failed.

0 if you do not want an audit report about the event when it failed.

If you don't specify success/failure extraction, both are included in the audit report.

For example, the following retrieves audit information only for failed attempts to change file ownership:

```
-e chown:0:1
```

Perform continuous audit reporting

If you use the `-f` option, the audit tool reads the audit log continuously, generating a report as it goes. The log is read even after the end of the file is reached. This allows you to extract audit data as it is being written to the audit log, which gives you the current audit information as it is generated. If you direct this report to a terminal and direct the audit log status information from the `-c` option of the `auditd` command to the same destination, then that terminal acts as a system audit monitor, displaying status reports on the log file and audit reports on system activities as they occur.

Note

As long as the `-f` option is in use, the audit log file is held open. This makes it impossible for the audit daemon to overwrite the current log in order to save disk space. No disk space is released until the `-f` option is no longer in effect and the current audit log is closed.

Select audit records for specific process IDs

To retrieve records associated with specific process IDs (PIDs), use the `-p` option and enter the PIDs you are interested in. The default is to select for all process IDs.

Filter out specific audit records that otherwise would be returned

You can also filter information in the audit log by using the `-d` option to designate the name of a deselection file.

The deselection file specifies a set of rules, and audit log records matching those rules are not included in the audit report. This feature can help you minimize the number of audit log records you have to review. Only one deselection file can be specified for each invocation of the `audit_tool` command.

Each deselection rule has the following form:

hostname audit_ID RUID event pathname flag

Use *flag* only to specify read or write mode for open events. Wildcards and simple pattern matches are supported with limitations. A "*" in a field always gives a match, and a string ending with a "+" matches any string that starts with the designated string. For example, the rule

```
* * * open /usr/lib/+ r
```

indicates that any open operations for read access on any object whose pathname starts with `/usr/lib/` is filtered out. Deselection takes precedence over other selection options.

For examples of the `audit_tool` command, see Section 4.8.

4.7 Suggested Audit Events

When deciding which system events to audit, you need to keep in mind that auditing uses system resources. Logging a large number of events to allow for in-depth auditing has a cost in terms of system performance.

The safest practice is to log all events at all times. But unless your system requires very thorough security protection, this is unnecessary. Typically, you can achieve an adequate level of protection by regularly logging a limited number of events and by performing deeper logging at more widely spaced intervals. This provides a reasonable level of auditing capability and minimizes the impact on system performance.

Note

If you vary the depth of logging, avoid signalling to the user community the times when the deep audits occur; otherwise, would-be violators, hoping to avoid detection, will simply avoid those times for their illicit activity.

4.7.1 Dependencies Among Audit Events

In order for certain information to be available, particular system calls must be logged. For example, to be able to retrieve audit records by user name, `login` must be logged. You should log the following at all times:

`login`	Provides a connection between the user name and the audit ID or real UID. When given an audit ID or RUID, the audit tool returns a user name only when `login` is being logged.
`chroot, chdir`	Log records do not include the current directory or absolute pathname of a process. If `chroot` and `chdir` are logged, then the audit tool can return the current working directory of a process.
`open`	Provides a connection between the file descriptor and the filename. When events such as `dup` are logged, the file descriptor allocated by the event but not the file name is returned. If you log `open` in addition to logging the process that allocates the descriptor, then you can learn the file name associated with the events.
`fcntl, dup, dup2`	Events that allocate file descriptors.
`exit`	Makes the audit tool more efficient. The audit tool keeps track of information for each process in the audit log it is analyzing. By knowing when a process exited, the audit tool can stop maintaining information on that process.

4.7.2 Auditable Events

The following tables lists all of the auditable events, dividing them into two categories: auditable system calls and auditable trusted events. A trusted event is an event that is associated with a security protection mechanism. It does not always correspond directly to a system call.

If you don't use the `auditmask` command to tailor the list of audited events to the needs of your site, then the events in the table are audited as indicated. A list of all auditable events can also be found in the the file `/etc/sec/audit_events`.

Table 4-2: Auditable System Calls

Event	Audited Conditions
accept	succeed fail
access	succeed fail
acct	succeed fail
audcntl	succeed fail
bind	succeed fail
chdir	succeed fail
chmod	succeed fail
chown	succeed fail
chroot	succeed fail
close	succeed
connect	succeed fail
creat	succeed fail
dup	succeed
dup2	succeed
execv	succeed fail
execve	succeed fail
exit	succeed

Table 4-2: (continued)

Event	Audited Conditions
exportfs	succeed fail
fchmod	succeed fail
fchown	succeed fail
fcntl	succeed
fork	succeed fail
ioctl	succeed fail
kill	succeed fail
killpg	succeed fail
link	succeed fail
lstat	succeed fail
mkdir	succeed fail
mknod	succeed fail
mount	succeed fail
msgctl	succeed fail
msgget	succeed fail
msgrcv	succeed fail
msgsnd	succeed fail
old setgid	succeed fail
old setpgrp	succeed fail
old setuid	succeed fail
old stat	succeed fail
old stime	succeed fail
old utime	succeed fail
old vfork	succeed fail
open	succeed fail
ptrace	succeed fail
reboot	succeed fail
recvmsg	succeed fail
rename	succeed fail
rmdir	succeed fail
semctl	succeed fail
semget	succeed fail
sendmsg	succeed fail
sendto	succeed fail
setdomainname	succeed fail
setgroups	succeed fail
sethostid	succeed fail
sethostname	succeed fail
setpgrp	succeed fail
setregid	succeed fail
setreuid	succeed fail
setsysinfo	succeed fail
settimeofday	succeed fail
shmat	succeed fail
shmctl	succeed fail
shmdt	succeed fail
shmget	succeed fail
shmsys	succeed fail
shutdown	succeed
stat	succeed fail

Table 4-2: (continued)

Event	Audited Conditions
symlink	succeed fail
truncate	succeed fail
umount	succeed fail
unlink	succeed fail
utimes	succeed fail
vhangup	succeed fail

Table 4-3: Auditable Trusted Events

Event	Audited Conditions
audgen8	succeed fail
audit_daemon_exit	succeed fail
audit_log_change	succeed fail
audit_log_creat	succeed fail
audit_log_overwrite	succeed fail
audit_reboot	succeed fail
audit_setup	succeed fail
audit_shutdown	succeed fail
audit_suspend	succeed fail
audit_xmit_fail	succeed fail
auth_event	succeed fail
login	succeed fail

The `audgen8` and `login` trusted events correspond to the use of those commands. The other trusted events relate to auditing and authentication activity as follows:

`audit_daemon_exit`

The audit daemon exited abnormally. This occurs only when there is insufficient memory available during initialization of the audit daemon. The exit is recorded in the new audit log, and a message is displayed on the system console.

`audit_log_change`

The audit daemon closed the current audit log and began writing a new log (for example, in response to the `-x` auditd option). The change in logs is recorded in the current (pre-change) audit log, and a message is displayed on the system console.

`audit_log_creat`

A new audit log was created in response to the accidental loss of the current log file. The new file has the generation number of the lost log file incremented by 1. The creation of the new log is recorded at the beginning of the new audit log, and a message is displayed on the system console.

`audit_log_overwrite`

The audit daemon began overwriting the current audit log in response to an overflow of the log (you previously used the `-o` auditd option to select `e`, which specifies overwrite as the overflow action). The overwrite is recorded at the beginning of the newly overwritten audit log, and a message is displayed on the system console.

`audit_reboot`

The audit daemon initiated a system reboot in response to an overflow of the log (you previously used the `-o` auditd option to select `b`, which specifies reboot as the overflow action). The reboot is recorded at the end of the current audit log, and a message is displayed on the system console before the reboot occurs.

`audit_setup`

The `-o` option of the `auditd` command was used to change the specified overflow action. The change in the audit setup is recorded in the current audit log.

`audit_shutdown`

The audit daemon was killed normally (typically, with the `-k` auditd option. The shutdown is recorded at the end of the current audit log, and a message is displayed on the system console when the shutdown occurs.

`audit_suspend`

The audit daemon suspends auditing in response to an overflow of the log (you previously used the `-o` auditd option to select `d`, which specifies suspension as the overflow action). The suspension is recorded in the current (pre-suspension) audit log, and a message is displayed on the system console.

`audit_xmit_fail`

The audit daemon was sending audit records across a network and the transmission failed. The failure is recorded in the local log specified as the alternate path (with the `-b` option) or the default local path `(/usr/adm)`.

`auth_event`

An event associated with user-authentication and the management of user accounts occurred. Trusted `auth_events` include `adduser`, `vipw`, `passwd`, `login`, and `edauth`. The event is recorded in the current audit log.

For a description of the information contained in a report for `login` and other `auth_events`, see Section 4.8.3

4.8 Reading Audits

The long format for an audit record returned by the `audit_tool` has three parts: header, body, and trailer. These parts can contain the following information:

- Header: information identifying the event:

```
auid:uid:username:
ruid:ppid:dev:
pid:
```

- Body: information about the event. Not all of the following information is returned for every event. The content of the body is determined by the particular event:

```
event:
shell:
char param:
gnode id:gnode dev:
descriptor:
directory:
int param:
request:
mask:
```

- Trailer: where and when the event occurred and the results:

```
result:
error:
ip_addr:
timestamp:
```

Items with a special relevance to security are:

auid	The audit ID.
ruid	The real user ID.
uid	The effective user ID.
pid	The process ID.
ppid	The parent process ID. If a security violation occurs for an event that is the offspring of another event, the ppid allows you to trace back through a list of processes to the originating event, so you can learn the RUID and UID (and user) associated with the original event.
event	The event that was logged, such as `open` or `chmod`.
char parameter	The parameter for the event; for example, if the event is the `chmod` system call, `char parameter` is the filename that was the argument for the system call.
error	If the event failed, the reason for failure. "Permission denied" is among the types of errors logged. For example, if a user attempts a file access for which the user does not have permission, the attempt will fail with the message "error: Permission denied" in the audit report.

Note that not all information is returned for all events. For example, if there is an `error`, there is no `result`.

4.8.1 Generating Abbreviated Audit Reports

The following `audit_tool` command extracts all the current content of the file `/var/adm/auditlog.000` and presents it in abbreviated format (the `-B` option):

```
# /usr/etc/sec/audit_tool /var/adm/auditlog.000 -B
```

```
AUID     RUID     R     E   PID
0        0        0     0   110   : stat ( /usr/lib/crontab )
0        0        0     0   1674  : open ( /dev/tty24 02 )
0        0        1     0   1674  : dup2 ( /dev/tty24 /dev/tty24 )
0        0        2     0   1674  : dup2 ( /dev/tty24 /dev/tty24 )
0        0        3     0   1674  : open ( /etc/gettytab 0 )
0        0        0     0   1674  : close ( 3 )
0        0        3     0   1674  : open ( /etc/gettytab 0 )
0        0        0     0   1674  : close ( 3 )
0        0        0     0   1674  : execve ( /bin/login )
0        0        0     0   1674  : setgroups ( )
0        0        0     0   1674  : audcntl ( 07 0 )
0        0        82    0   1674  : audcntl ( 05 0 )
1123     0        0     0   1674  : audcntl ( 013 0 02143 )
1123     0        0     0   1674  : login
1123     0        0     0   1674  : audgen ( )
1123     1123     0     0   1674  : setreuid ( 02143 02143 )
1123     1123     6     0   1674  : open ( /etc/ttys 0 0666 )
1123     1123     0     0   1674  : close ( 6 )
1123     1123     0     0   1674  : open ( /usr/ucb 0 )
1123     1123     0     0   1674  : close ( 0 )
1123     1123     0     0   1674  : open ( /bin 0 )
```

The column headings of the report indicate the following:

AUID — The audit ID associated with the event.

RUID — The real UID associated with the event.

R — The result number. Refer to the reference page for the specific system call for information about the result number.

E — The error code. For a list of error codes and their meanings, see the reference pages for `errno`(2). Error codes of particular relevance to security are:

- 1 — Typically this indicates an attempt by someone other than the owner to modify a file in a way reserved to the file owner or superuser. It can also indicate an attempt by an ordinary user to do things limited to the superuser.
- 13 — An attempt was made to access a file in some way forbidden by the protection system.
- 30 — An attempt was made to access a file or directory on a mounted file system when that permission has been revoked. For example, an attempt was made to write a file on a file system mounted read only.

PID — The process ID.

The event and its argument appear in the last column.

4.8.2 Generating Audit Reports for System Calls

The following command extracts records of `execve` and `open` system calls (the `-e` option) from the file `/var/adm/auditlog.000` and presents them in long format.

```
# /usr/etc/sec/audit_tool -e execve -e open /var/adm/auditlog.000
```

```
ruid: 0                 uid: 0
pid: 1674               ppid: 1             dev: (39,0)
event:        open
char param:   /dev/tty24
int param:    2
result:       0
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:03 1989 EDT

ruid: 0                 uid: 0
pid: 1674               ppid: 1             dev: (39,0)
event:        execve
char param:   /bin/login
gnode id:     12348   gnode dev:    5
result:       0
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:05 1989 EDT

audit_id: 1123          uid: 1123           username: jones
pid: 1674               ppid: 1             dev: (39,0)
event:        open
char param:   /etc/ttys
gnode id:     4432    gnode dev:    0
directory:    /
int param:    0
result:       6
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:22 1989 EDT
```

4.8.3 Generating Audit Reports for Trusted Events

The following command extracts records of the trusted event auth_event. from the file `/var/adm/auditlog.000` and presents them in long format.

```
# /usr/etc/sec/audit_tool -e auth_event /var/adm/auditlog.000

ruid: 0                 uid: 0
pid: 4152               ppid: 1             dev: (39,0)
event:        login
login name:   sdf
home dir:
shell:        /bin/sh
char param:   Login incorrect
char param:   tty00
char param:   AB32K1
error:        1
ip_addr:      130.180.6.62 (systmG)
timestamp:    Tue Dec 19 13:19:44 1989 EDT
```

```
audit_id: 1242          uid: 0              username: jones
pid: 4155               ppid: 1             dev: (39,0)
event:        login
login name:   jones
home dir:     /
shell:        /bin/csh
char param:   Login succeeded
char param:   tty00
char param:   AB32K1
directory:    /
result:       0
ip_addr:      130.180.6.62 (systmG)
timestamp:    Tue Dec 19 13:24:23 1989 EDT


ruid: 0                 uid: 0
pid: 4165               ppid: 4164          dev: (20,4)
event:        login
login name:   jones
home dir:     /
shell:        /bin/csh
char param:   Too many users
char param:   ttyp4
char param:   systmG
error:        1
ip_addr:      130.180.6.62 (systmG)
timestamp:    Tue Dec 19 13:24:36 1989 EDT
```

In addition to the standard log record information, every record contains fields for a login name, a home directory (home dir), and shell. The record also contains up to three "char param" fields, which describe the event, the incoming terminal line, and the remote system name, where applicable.

The remote system name is not present if the terminal line is a hardwired line. The remote system name is the LAT server name if the line is a LAT line. The remote system name is the host name if the login is via rlogin. There is always either an error field or a result field. It corresponds to the exit code that login exits with. The result is zero only when the login was successful.

If login fails for a reason other than incorrect password, the reason for failure is given, for example,

- The account was disabled
- The password was expired

For the auth_event passwd, the audit ID and UID are recorded. This allows you to determine whether the user or administrator changed a password.

4.8.4 Generating Audit Reports for Process IDs

The following command extracts records of events with a process ID of 1674 (`-p` 1674) from the file `/var/adm/auditlog.000` and presents them in long format.

```
# /usr/etc/sec/audit_tool -p 1674 /var/adm/auditlog.000
```

```
ruid: 0                 uid: 0
pid: 1674               ppid: 1             dev: (39,0)
event:        dup2
gnode id:     4244    gnode dev:    0
descriptor:   /dev/tty24 (0)
descriptor:   /dev/tty24 (1)
result:       1
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:03 1989 EDT

ruid: 0                 uid: 0
pid: 1674               ppid: 1             dev: (39,0)
event:        dup2
gnode id:     4244    gnode dev:    0
descriptor:   /dev/tty24 (1)
descriptor:   /dev/tty24 (2)
result:       2
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:03 1989 EDT

ruid: 0                 uid: 0
pid: 1674               ppid: 1             dev: (39,0)
event:        open
char param:   /etc/gettytab
gnode id:     4219    gnode dev:    0
int param:    0
result:       3
ip_addr:      130.180.4.82 (sys_A)
timestamp:    Fri Jun  2 14:07:03 1989 EDT
```

4.9 Traditional Auditing Tools

You are urged to use the ULTRIX audit subsystem for security-relevant auditing. It is designed as a secure subsystem to protect audit information from tampering or loss, and its features are flexible and easy to use. However, the traditional UNIX auditing tools do provide some auditing capabilities for the following categories of events:

- local login and logouts
- file Transfer Protocol (FTP) logins
- external logins and logouts for TCP/IP (rlogin and telnet)
- external logins and logouts for DECnet (dlogin and set host)
- failed logins
- failed attempts to become super user (the `su` command)
- reboots and crashes
- rsh and rcp file transfer requests
- DECnet file transfer requests

Auditing for each of these event categories involves a data file, which stores the pertinent information, and a method for viewing the stored data. In some cases this method is a specific command, such as `last` or `lastcomm`. In other cases the contents of the file are viewed directly, for example, with the `more` command.

4.9.1 Auditing Logins and Logouts

Information on logins and logouts is stored in the file `/usr/adm/wtmp`. This file contains information pertaining to all types of logins including local, LAT, DECnet, TCP/IP and ftp. The file is updated by the system for each login and logout.

Use the `last` command to read `/usr/adm/wtmp`. Entering the command alone will provide you with the complete contents of the wtmp file, starting with the most recent login. You can also select login information for particular users and ttys. See the reference pages for `last`(1) for a complete description of the command's use.

To display the login records for user names smith, jones, and doe, enter the following command:

```
# last smith jones doe
```

To display the login records for special devices `/dev/tty12` and `/dev/tty14`, enter the following command:

```
# last tty12 tty14
```

Specifications are additive. For example, combining both user names and `/dev/tty` specifications in the same `last` command results in the display of all records relevant to both the user names and the ttys.

Typical output from the `last` command is as follows:

```
doe      tty19                    Thu Jan  5 08:56   still logged in
smith    ttyp1    sysA            Wed Jan  4 22:31 - 22:31  (00:00)
jeff     ttyp2    sysD            Thu Jan  5 09:29 - 10:29  (01:00)
jeff     ttyp2    sysD            Thu Jan  5 10:29   still logged in
jeff     ttyp4    sysD            Thu Jan  5 10:11 - 10:20  (00:08)
jeff     ttyp4    sysD            Thu Jan  5 10:29 - 10:46  (00:17)
jones    tty23    SYSC            Thu Jan  5 10:17   still logged in
```

The information in each column of a report from the `last` command has the following significance:

column 1 The user name

column 2 The device where the login occurred

column 3 The system where login originated. If the entry is in lowercase letters, the access was through TCP/IP (rlogin or telnet). If the entry is in uppercase letters, the login was a remote DECnet login.

A blank third column reflects a local login unless the tty shown in column two is in the range of tty[pq]*, in which case the login was initiated from an already established remote session or was a localloop login.

column 4 The date, time and duration of the login session. If the user is still logged into the system at the time the `last` command is executed, that fact is indicated.

A report of the total hours of system usage can be obtained with the `/etc/ac` command, which summarizes the `wtmp` file.

4.9.2 Auditing Failed Logins

After five consecutive failed attempts to login under a single login invocation, the fifth failure is recorded in `/usr/spool/mqueue/syslog`. You can retrieve this information and store it in a file named `Report`, by entering the following command:

```
# grep FAILURES /usr/spool/mqueue/syslog >> Report
```

Each record reflects five consecutive failed attempts to login. The date and time, system of origin, device of origin, and user name are reported.

The following is a sample of the output obtained from the search for failed logins:

```
Jan  4 15:49:58 localhost: login: REPEATED LOGIN FAILURES ttyp2, root
Jan  4 15:50:29 localhost: login: REPEATED LOGIN FAILURES ttyp2, daemon
Jan  4 15:56:47 localhost: login: REPEATED LOGIN FAILURES ttyp2, smith
```

Notice that login failures are attributed to the user name used on the fifth login attempt. This means that someone could attempt to login four times as root, then try once as smith, and the report would attribute the failed attempts to smith. The fact of four failures to login as root would not be reported.

4.9.3 Auditing Superuser Access

All attempts, successful and failed, to use the `su` command to login as superuser are logged to the file `/usr/adm/sulog`.

This is a plain text file, and its contents can be viewed with the `more` command or by similar means. You can retrieve records of failed `su` attempts, by entering the following command:

```
# grep BADSU /usr/adm/sulog >> Report
```

The following is a sample of output from the search for BADSU. The user name, device, and date and time are provided.

```
BADSU: doe /dev/tty19 Fri Dec  9 13:52:43 1988
BADSU: smith /dev/tty20 Wed Dec 14 10:18:12 1988
BADSU: smith /dev/ttyp3 Wed Dec 28 11:26:14 1988
BADSU: jones /dev/ttyp1 Tue Jan  3 09:34:29 1989
```

4.9.4 Auditing Reboots and Crashes

As security administrator, one of your duties is to maintain the availability of system resources. The files `/usr/adm/wtmp` and `/usr/adm/shutdownlog` contain information regarding system shutdowns and reboots. Unexpected shutdowns are recorded in the `/usr/adm/wtmp` file and in the `/dev/errlog` reporting system, in records of the system startup following a shutdown. The `shutdown` command records shutdown requests in `/usr/adm/shutdownlog`. The shutdownlog records the time, date, whether the event was a halt, a halt and reboot, or a shutdown, and the reason for the shutdown and who invoked the command. For example,

```
12:29  Wed Jan 18, 1989.  Halted for reboot.
20:03  Fri Jan 20, 1989.  Halted for reboot.
12:06  Thu Feb 23, 1989.  Shutdown: service (by sys_1!jones)
07:29  Fri Mar  3, 1989.  Halted.
07:04  Thu Jun  1, 1989.  Shutdown: power failure (by sys_1!root)
```

You can also get shutdown information with the `last` command, by designating `reboot` and `crash` by entering the following command:

```
# /usr/ucb/last crash >> Report
# /usr/ucb/last reboot >> Report
```

The following is an example of the resulting `Report` file. Each reboot record shows the time and date of execution. No crashes are reported in this example.

```
reboot    ~                         Mon Jan  9 11:50
reboot    ~                         Fri Dec 16 07:20
reboot    ~                         Thu Dec 15 17:52
```

4.9.5 Auditing External File Transfers

The ability to track external file accesses provides you with a means of monitoring attempts at compromising files. Excessive file transfers, for example, might indicate attempts to ferret out system vulnerabilities. The external file accesses described here are of the DECnet and TCP/IP (rcp) type.

Records of DECnet file transfers are logged in the file `/usr/spool/mqueue/syslog`. Each request for DECnet file transfer is identified by the string "fal" (for file access listener) in the syslog.

Extract relevant DECnet file accesses with simple search of `/usr/spool/mqueue/syslog`. You can do this by entering the following command:

```
# grep fal /usr/spool/mqueue/syslog >> Report
```

Each record in `Report` shows

- the date and time
- the system to which the request was directed
- the process ID
- the type of access
- the net address of the user making the request
- the directory to which access was requested
- the files involved

A complete access record can span many individual records. This is a result of extended system logging at debug level 9, which allows problems, as well as completed and successful accesses, to be logged.

An example of `Report` follows:

```
Jan  4 11:35:22 localhost: 13541 dnet_spawner: Connect from
     SYSA::DOE for object 17 (fal)
Jan  4 11:35:23 localhost: 13541 fal: DIRECTORY access from
     SYSA::DOE, user=guest, directory=/, filename=[USR.USERS]*.*;*
Jan  4 11:35:41 localhost: 13554 dnet_spawner: Connect from
     SYSA::DOE for object 17 (fal)
Jan  4 11:35:41 localhost: 13554 fal: DIRECTORY access from
     sysA::DOE, user=guest, directory=/, filename=*.*;*
Jan  4 11:37:21 localhost: 13570 dnet_spawner: Connect from
     SYSC::JONES for object 17 (fal)
Jan  4 11:37:22 localhost: 13570 fal: DIRECTORY access from
     SYSC::JONES, user=guest, directory=/, filename=USR.USERS;*
```

In order to audit TCP/IP file accesses, you must have accounting active on your system. To activate accounting, enter the commands that follow. The first command creates the accounting file `/usr/adm/acct`. The second command activates accounting and directs reporting to `/usr/adm/acct`.

```
# touch /usr/adm/acct

# /etc/accton /usr/adm/acct
```

Once accounting is active, you can extract information from `usr/adm/acct` with the `lastcomm` command.

All `rsh` (remote shell), `rshd` (remote shell server), and `rcp` (remote file copy) requests to the system are logged in `/usr/adm/acct`. To use the output from `lastcomm` to search for rsh,rcp and rshd command occurrences, use the following command:

```
# lastcomm | grep 'r[cs][ph]' >> Report
```

Each record in a report shows

- the relevant command
- whether or not it ran as superuser (an S in the second column indicates superuser status)
- the user name associated with the command
- the controlling tty if the command was executed on the local system (-- for remote access)
- the CPU time
- the date and time of command start

For example,

```
rsh         S     doe       ttyp0      0.11 secs Tue Jan 10 13:33
rcp               smith     __         0.12 secs Tue Jan 10 13:33
rcp               smith     __         0.14 secs Tue Jan 10 13:33
rcp         S     doe       ttyp0      0.09 secs Tue Jan 10 13:32
rcp         S     doe       ttyp0      0.08 secs Tue Jan 10 13:30
rshd        S     root      __         0.08 secs Tue Jan 10 10:03
```

4.10 Archiving Log Files

By their nature, log files continue to grow over time and can fill the available disk space. This is especially true of `/usr/adm/wtmp`, which receives a great deal of accounting information. If a file fills the available disk space, information important to maintaining security might be lost. To avoid the loss of log data due to the overflow of disk space, you must periodically archive the current log files and begin new files. This is especially true of `/usr/adm/wtmp`, which receives a great deal of accounting information.

Archiving for the audit log of the audit subsystem is largely automatic. A description of the process is given in Section 4.5.

The accounting files used by traditional audit tools must be regularly archived. You can automate the archiving process by using the `/usr/lib/crontab` file. Because archive files can be quite large, you might want to use the `compress` command to reduce the files.

In most cases, the archive procedure is as follows:

1. Create an empty file with the command

```
# touch new_log_file
```

2. Copy the current log file to an archive file with the command

```
# cp current_log_file archive_log_file
```

3. Overwrite the current log file with the empty file with the command

```
# mv new_log_file current_log_file
```

4. Compress the archived file with the command

```
# compress archive_log_file archive_log_file
```

Note

The procedure is ordered to minimize the time during which information written to a log file is lost. However, any log information written between the executions of steps 2 and 3 is lost. This gap can be minimized by combining steps 2 and 3 as follows:

```
# cp current_log_file archive_log_file ;
# mv new_log_file current_log_file
```

4.11 Responding to Audits

Whenever you suspect an effort is being made to violate security, you should consider increasing the frequency of auditing. Additionally, you might want to tailor the list of events being audited to gather more specific information about the attempted violations. For example, if the attacks are against the file system, you might want to log all failed and successful file `opens` and `closes`, `links`, `unlinks`, `chdirs`, `chmods`, `chowns`, and other file-related activities.

When the audit trail implicates a specific authorized user in attempts to violate security, you can take the following steps:

- Talk with the user, reminding him or her of the importance of maintaining security and the need for all users to contribute to that effort.
- Restrict the user's access to the system by placing the user in a group of one.
- In extreme cases, deny the user system access by removing the user's account. This can be on a temporary or permanent basis.
- Audit the offending user's activities for indications that the user's behavior has changed. When you extract audit information, pay close attention to activities associated with the user's audit ID, UID, RUID, and user name. Use the `-a -u` and `-U` options of the `audit_tool` command to select audit records for specific audit IDs, UIDs, and user names. Refer to Section 4.6 for more information about extracting information from the audit log.

If the audit trail indicates attempts to violate security but points to no specific user, it is possible that you are faced with intrusion by an outsider. Your responses must then be directed to the system and the larger user community. In this case you can take the following steps:

- Have users change their passwords and inform them about the selection of safe passwords.
- Hold meetings with users to discuss the importance of system security.
- Increase physical security to make sure that only authorized users can gain physical access to the system.
- Perform backups of the file system more frequently, to minimize the damage if a break-in should occur and data on the system is lost or altered.
- If attacks seem to be coming in over a network, increase the auditing of network-related activities.

Protecting the File System 5

This chapter describes steps to take to maintain the security of files and the file system. It is assumed you are familiar with the `umask`, `chmod`, and `chown` commands, and with using octal values to specify file modes. For more information about these subjects, refer to the *Security Guide for Users*.

The major subjects covered in this chapter are

- Protecting files and directories
- Mounting file systems
- Creating groups for common file access
- Suggestions for secure file system management

5.1 Protecting Files

In this section you will find suggestions for owners, groups, and modes for various categories of files. After you make the changes indicated here, you can use the command

```
# ls -lgd
```

for directories or

```
# ls -lg
```

for files to verify that the owners, groups and modes have been correctly set.

5.1.1 Suggested Modes and Owners for Command Files

By protecting commands, you prevent the possibility of an illicit command being substituted for an actual command. Such an illicit command could, among other things, read the unwitting user's files and pass the information to those not authorized to see it.

Typically, command directories should be owned by the root account, and no other users (who are not superusers) should have write access to the directories.

The following table lists files and directories associated with commands and executables, and the appropriate owners and modes for those files.

Table 5-1: Suggested Protections for Command Directories

File/Directory	Owner	Group	Mode	Security Relevance
/	root	system	755	root of all file systems and home directory of the superuser
/usr	root	system	755	a file system hierarchy
/bin	root	system	755	user commands
/usr/bin	root	system	755	additional user commands
/etc	root	system	755	commands for system maintenance/administration
/usr/etc	root	system	755	more commands for system maintenance/administration
/etc/sec	root	system	755	commands for security administration
/usr/etc/sec	root	system	755	more commands for security administration
/lib	root	system	755	a link to /usr/lib
/usr/lib	root	system	755	many system executables, such as the compiler and system libraries.
/usr/ucb	root	system	755	certain commands that are Berkeley extension
/usr/new	root	system	755	additional, optional commands
/usr/local	root	system	755	commands with a local origin
/usr/local/bin	root	system	755	additional commands with a local origin

Individual commands within a directory can require special modes set in addition to the permission mode bits. In particular, some commands require the setuid, setgid, and sticky bits to be set.

You can use the `fverify` command to ensure the correctness of the owner, group, and mode (setuid, setgid, sticky bit, and permission bits) of the command files. Correctness is determined by the defaults in the software subsets inventory, `/usr/etc/subsets/*.inv`. Any inconsistencies between the values in the subsets and the installed files are reported to the file `/verify.log`.

The command can act as follows:

- Repair inconsistencies after first checking with you (`fverify` command without any options)
- Repair inconsistencies without asking first (`-y` option)
- Only report inconsistencies; do not repair them (`-n` option)

The following example checks the installed files against all the subset inventories and reports in `/verify.log` on any inconsistencies.

```
# fverify -n < /usr/etc/subsets/*.inv
```

For more information about the command, see `fverify`(8) in the *ULTRIX Reference Pages.*

5.1.2 Suggested Modes and Owners for Special Files

The `/dev` directory is the location of special files that are entry points to such resources as system disks and memory. A user with direct access to these files has access to any of the data they store, so it is important to protect the special files.

When implementing the changes suggested in the following table, in the file specifications for `/dev/<disktype>*` and `/dev/r<disktype>*`, substitute the actual disktype of the disks on your system for `<disktype>`. The asterisk is the shell wild card character and should be included when you enter the file name. You can learn the disk types for your system by looking at the entries for "disk" in `/usr/sys/conf/vax/`*SYSTEM_NAME* or `/usr/sys/conf/mips/`*SYSTEM_NAME*. Replace *SYSTEM_NAME* with the actual name of your system.

The following table lists files and directories associated with special files and the appropriate owners and modes for those files.

Table 5-2: Suggested Protections for Special Files

File/Directory	Owner	Group	Modes	Security Relevance
`/dev`	root	system	755	home directory for special files
`/dev/<disktype>*`	root	system	600	buffered disk system
`/dev/r<disktype>*`	root	system	600	unbuffered disk system

5.1.3 Suggested Modes and Owners for User Account Files

The user account files control who can legitimately log in to the system. As such, they are vital to protecting your system against unauthorized access.

The following table lists files and directories associated with user accounts, and the appropriate owners and modes for those files.

Table 5-3: Suggested Protections for User Account Files

File/Directory	Owner	Group	Modes	Security Relevance
`/etc/passwd`	root	system	644	user accounts in BDS mode
`/etc/passwd.pag`	root	system	644	data for password data base
`/etc/passwd.dir`	root	system	644	directory for password data base
`/etc/auth.pag`	root	authread	640	data for authentication data base

Table 5-3: (continued)

File/Directory	Owner	Group	Modes	Security Relevance
/etc/auth.dir	root	authread	640	directory for authentication data base

5.1.4 Suggested Modes and Owners for Log Files

Log files are your record of activity on the system. They allow you to determine which users are accountable for which actions.

The following table lists log and accounting files and the appropriate owners and modes for those files.

Table 5-4: Suggested Protections for Log and Accounting Files

File/Directory	Owner	Group	Modes	Security Relevance
/usr/adm/auditlog	root	system	600	log file for auditing subsystem
/usr/adm/acct	root	system	644	raw system accounting data, including user commands executed
/usr/adm/usracct	root	system	644	reduced system accounting data
/usr/adm/savacct	root	system	644	reduced system accounting data
/usr/adm/wtmp	root	system	644	successful logins, logouts, shutdowns and reboots
/usr/adm/eventlog	root	system	640	DECnet events
/usr/adm/shutdownlog	root	system	640	system shutdowns and reboots
/usr/adm/sulog	root	system	640	failed and successful attempts to login as superuser
/usr/spool/ mqueue/syslog	root	system	640	failed logins, requests for DECnet file transfers, mail transfers, remote DECnet login connections

5.1.5 Protecting Network-Related Files

To maintain your control over system access, assign write protection from general users to files that grant users from remote hosts access to your system. In some cases, read protection is also appropriate to prevent unnecessary access to sensitive information.

The following table lists sensitive files related to networking and appropriate protections for those files.

Table 5-5: Suggested Protections for Network Files

File/Directory	Owner	Group	Modes	Security Relevance
`/etc/exports`	root	system	644	names of file systems available for NFS export and restrictions on the exportations
`/etc/hosts`	root	system	644	names and addresses of remote hosts with network access to local host
`/etc/hosts.equiv`	root	system	600	grants remote user access to local system without password
`<user's>.rhosts`	user's name	user's group	640	user-owned file that grants remote user access to local system without password
`<user's>.netrc`	user's name	user's group	600	information used with ftp login and initialization

5.1.6 Protecting Other Files

Suggestions for files that require protection but are not part of a larger class of files are presented here.

5.1.6.1 Software Subsets – The inventories of software subsets reside in `/usr/etc/subsets/*.inv` (the asterisk indicates the shell wild card). The `fverify` command uses these files to check and, if necessary, reset the modes and other attributes of the installed software. A user with write access to subset inventories can control the permission modes set by `fverify`. Protect the file with permissions 644.

5.1.6.2 File System Description – The contents of `/etc/fstab` describe the file system. The file can be used by the `mount` command to configure the file system, including mounting files and adding (or removing) the security-relevant restrictions, `nodev`, `nosuid`, and `noexec`. A user with write-access to `/etc/fstab` could create a version of the file that, when applied by the `mount` command, would

remove protective restrictions from mounted file systems. Assign permissions 644 to this file.

5.1.6.3 Terminal Port Configuration – The `/etc/ttys` file contains information used by the system to initialize terminal special files and control their use. Each line in the file describes a configuration for a terminal. If the `secure` flag appears in a terminal description, then superuser login at that terminal is supported. A user must invoke the `su` command to gain superuser status at terminal that does not have the `secure` flag set. The `/etc/ttys` file also specifies a program to be executed when the terminal initializes.

Because of the secure flag and the possibility of a Trojan horse being designated as the initialization startup program, do not allow users write access to this file. Assign permissions 644.

5.1.6.4 Login Tables – The `/etc/gettytab` file can control various aspects of the login process, including the name of the login program executed and the login greeting. Because of this, protect the file with permissions 644.

5.1.6.5 Scheduled Administrative Commands – The file `/usr/lib/crontab` contains commands and instructions on when and how those commands are executed. Usually the commands in this file are used periodically for system administration and maintenance. For example, the file might contain instructions for performing a nightly backup of certain files, thus relieving the manager of having to manually invoke the command every evening.

The file is used by the `cron` command, which is invoked once and then continues to run, monitoring `/usr/lib/crontab`. Commands in `/usr/lib/crontab` are executed with superuser status, so only trusted individuals should be allowed to write to the file. Set permissions to 644.

For more information, see `cron`(8) and `crontab`(5) in the *ULTRIX Reference Pages.*

5.1.6.6 Automatic Startup Command Scripts – Two files determine the automatic startup process: `/etc/rc` and `/etc/rc.local`. The command script `/etc/rc` is generic in nature and applies to a broad range of startup activities; `/etc/rc.local` has site-specific information. Protect both files against write access by others by setting permissions to 640. If others require read access, permissions of 644 are acceptable.

5.1.6.7 Crash Dump Files – If the `savecore` command appears in the `/etc/rc.local` file, then after a system crash, the contents of main memory are written to `/usr/adm/crash.vmcore.n`, and the kernel image is written to `/usr/adm/crash.vmunix.n`. The trailing `n` in the file names is a number incremented each time the `savecore` command is invoked.

Because sensitive data can be part of the contents of the dump files, all access to the files should be restricted. The files are owned by root, and are created with access permissions determined by the umask for the root account.

For more information, see `savecore`(8) in the *ULTRIX Reference Pages.*

5.1.6.8 Configuration File – The file `/etc/svc.conf` contains information about system configuration, including the minimum required password length and the grace period for expired passwords. Protect the file with permissions 644.

5.1.6.9 User Environment Files – Several files are common to every user's home directory, although the contents can vary from user to user. The files `.profile` and `.login`, `.logout`, and `.cshrc` describe aspects of the user's working environment on the system. The file `.forward` directs where mail for the user is to be forwarded.

Appropriate modes for these files are 640; the files should restrict group and other access. If others require read access to the files, permissions of 644 are acceptable. For more information about the user environment files and suggestions for their contents, see Section 3.6.

Table 5-6: Suggested Protections for Miscellaneous Files

File/Directory	Owner	Group	Modes	Relevance
`/usr/etc/subsets/*.inv`	root	system	644	inventories of software subsets
`/etc/fstab`	root	system	644	file system configuration
`/etc/ttys`	root	system	644	terminal port initialization data
`/etc/gettytab`	root	system	644	terminal configuration data
`/usr/lib/crontab`	root	system	644	commands and instructions for their execution by the cron command
`/etc/rc`	root	system	640	controls automatic startup process
`/etc/rc.local`	root	system	640	site-specific startup information
`/var/adm/crash`	root	system	700	directory for system dump files
`/etc/svc.conf`	root	system	644	system configuration information

Table 5-6: (continued)

File/Directory	Owner Group	Modes	Relevance
<user's>.profile	user's name and group	640	environment file
<user's>.login	user's name and group	640	environment file
<user's>.logout	user's name and group	640	environment file
<user's>.cshrc	user's name and group	640	commands for execution prior to invocation of the C shell
<user's>.forward	user's name and group	600	mail forwarding address

5.2 Mounting and Unmounting File Systems

For a file system to be generally available as a resource, it must be mounted as part of the main file system. The directory at which a file system is mounted is referred to as the mount point for that file system. The `root` directory (/) is at the top of the file system, and it is always mounted: there is no way to unmount `root`.

You mount a file system with the `mount` command, by designating the file system to be mounted and the mount point, or by using the `-a` option to direct the command to mount file systems according to the specifications in the file `/etc/fstab`.

The `-o` option of the `mount` command enables you to choose among three options that are especially useful for placing protective restrictions on the file system being mounted:

nodev Denies access to any device files in the file system being mounted.

nosuid Denies users the ability to execute setuid and setgid programs in the file system being mounted.

noexec Denies users the ability to execute any binaries in the file system being mounted.

5.2.1 Guidelines for Mounting File Systems

Apply the following restrictions when mounting file systems. As such, they should appear in the `/etc/fstab` file.

- Use the `nodev` option when mounting any file system other than `root` (/).
- Use the `nosuid` and `nodev` options when mounting general user directories.
- Use the `noexec` and `nodev` options when mounting any file system that is not intended to contain executables.

- Use the `nosuid` option if you mount `/tmp`, `/usr/tmp`, or general user directories.

5.2.2 Mounting Foreign File Systems

Any file systems that come from outside the controlled environment of your system must be assumed to be unsafe. Imported file systems might contain any of the following:

- computer viruses
- Trojan horses
- executables that do not follow safe programming practices, including executables that are inappropriately SUID

Before you make imported file systems available on your system, ascertain that the files are safe. This means that you must examine the files and learn their content, or you must know and trust the origins of the files and also know the files were not tampered with during transport.

Take the following precautions with imported file systems:

- Check the file system for technical corruption with the following command, substituting the name of your file system for *file_system_name*:

  ```
  # /etc/fsck -P dev/file_system_name
  ```

  If there is any corruption, notify the source of the file system and get a new, clean version. Repeat the check with the new file system.

- Create a directory owned by superuser (you), to which only you have access:

  ```
  # umask 077
  # mkdir safe_mount
  ```

- Mount the imported file system, read-only, at the new directory:

  ```
  # mount -r dev/file_system_name f(CB/safe_mount
  ```

- Check the imported file system for privileged programs, executables that are SUID, and special files, such as device files. Satisfy yourself that you understand how all executables will behave.
- After you are confident the file system is safe, unmount it and remount it at the desired location.

5.3 Creating Groups for Common File Access

Maintaining the appropriate level of security is a balancing act between the need to restrict access to system resources so that those resources are protected, and the need to share system resources so that users can do their work efficiently. Groups are a feature that allow controlled sharing of resources.

A group is a collection of users who, in addition to having their individual identities on the system, share a common name--the group name, and a common ID--the group ID (GID). Members of a group can be allowed group access to objects (files and processes) that are owned by the group.

The system stores definitions of groups in the file /etc/group. Each line in this file corresponds to a group and has the following form:

```
<group name>:*:<GID>:<group members--user names separated with commas>
```

The second field, the asterisk, is a password field. The ULTRIX system does not support group passwords, because a user is automatically treated as a member of any group the user belongs to.

A typical group definition might look as follows:

```
accounting:*:31:doe,smith,barron,jones,stuart
```

This line makes users Doe, Smith, Barron, Jones, and Stuart members of the group named accounting, which has a GID of 31.

To define a group, use the /etc/addgroup command.

This command enables you to edit the /etc/group file, entering a group name, a GID, and a list of user names that represent the members of the group.

A user is a member of a group if either of the following is true:

- The GID in the user's account in /etc/passwd is the same as the GID for the group in /etc/group.
- The user name for the user appears in the list of group members in the group definition in /etc/group.

Note that while a user has only one GID, the one that appears in /etc/passwd, you can make the user a member of many groups by using the addgroup command to edit /etc/group and add the user name to the membership lists for the groups.

An object is considered to be owned by the group if the object and group have the same GID. The degree of access a group member has to an object is determined by the group permissions of the object.

If a group member creates a file and wants to make it a group owned file, the member uses the chgrp command to give the file the GID for the group, and then sets appropriate group permissions for the file.

5.4 Suggestions for Secure File System Management

Maintaining a secure file system requires the cooperation of the entire user community and regular monitoring on your part of the file system.

5.4.1 Setting the Default Protection Mask

The umask command enables you to assign a default mask for file modes for the current session on the system. In the .login and .profile files for each user, the following line should appear:

```
umask 027
```

With this line in the files, each time the user logs in, his or her umask is set to 027. Files created during the session have at most permission 750 (user=rwx, group=rx, other=no access), although more restrictive permissions are still allowed. The user retains the option of later using the chmod command to assign more or less restrictive permissions to files. For more information about the suggested contents of .profile and .login, see Section 3.6.

5.4.2 Backing Up the File System

By following the suggestions presented in this chapter, you provide a high level of security for files and the file system. As an additional precaution, you should perform frequent backups of the file system and archive these backups in a secure area. If a serious security break-in occurs, the backup files provide you with a method for restoring all files to a known base level.

For information on backing up the file system, see the *Guide to System Backup and Restore*.

5.4.3 Monitoring the File System

In your work with the file system, you have available to you commands for checking the status of files, disk systems, and the file system. Three commands offer particularly helpful features:

To display information about the disk system:

The `df` command returns disk system statistics including the amount of disk spaced used, the amount remaining, and the mounted file systems. For more information, see `df`(1) in the *ULTRIX Reference Pages*.

To search path hierarchy for files with matching attributes:

The `find` command is very flexible and returns file names according to a wide range of characteristics you specify, including the permission modes. The command searches recursively down the directory hierarchy from the pathname designated in the command.

For example, the following command searches the entire file system, finding all executables owned by root that are setuid, and places the file names in `suid_check`.

```
# find / -perm -4000 -user root -print > suid_check
```

For more information, see `find`(1) in the *ULTRIX Reference Pages*.

To find files containing security vulnerabilities:

The `ncheck` command with the `-s` option generates a report of all executable files that are SUID and all special files. To use the command, enter the following:

```
# /etc/ncheck -s
```

For more information, see `ncheck`(8) in the *ULTRIX Reference Pages*.

Protecting Systems in Networks 6

This chapter describes methods of protecting a system that is part of a network of computers. The major subjects covered are

- Protecting systems in LAT (Local Area Transport) networks
- Protecting systems in DECnet networks
- Protecting systems in TCP/IP (Transmission Control Protocol/Internet Protocol) networks
- Protecting NFS (Network File System) files
- Managing network accounts
- Restricting the environment of remote users

By their nature, networks are intended to provide users access to remote systems. In such an environment, providing security while providing network access requires careful planning. The object of this chapter is to describe measures to prevent network access to your system by unauthorized users of remote systems.

6.1 Protecting Systems in a LAT Network

Because LAT networks are local in nature, you have a high degree of control over the LAT environment and who has physical access to LAT devices. In addition to controlling physical access, you have available two features that enhance LAT security:

LAT login

You can require users to enter a password to gain access to the LAT.

LAT groups

You can establish LAT groups and then restrict host communication to particular groups by designating those groups with the `-g` option of the `lcp` command. Usually this command appears in the file `/etc/rc.local`.

For more information about LAT configuration, see `lcp`(8) in the *ULTRIX Reference Pages*.

6.2 Protecting Systems in a DECnet Network

The DECnet protocol has several inherent security features. One of these is the protocol, which includes the identification of the remote user. This allows the local system to identify the user and the node of origin from information in the protocol packet. DECnet provides you with a number of ways to control network access to your system.

6.2.1 Open DECnet Access: Default User Accounts

A default user account, often referred to as a guest account, is an entry in the `/etc/passwd` file that has `nologin` in the password field. Such accounts usually look something like this:

```
guest:Nologin:268:10:DECnet guest:/usr/network/guest:/bin/csh
```

While such an account cannot be used for local access to the system, it allows access by any remote DECnet users, including those without accounts on your system. Any files on your system that have read or execute permission for `other` can be read or executed by the guest user.

Note

To maintain the security of a system in DECnet, do not establish a default user account in the `/etc/passwd` file.

6.2.2 Controlling DECnet Access to Your System

In the absence of a default user account, remote DECnet users must have an account on your system in order to access it. This means that remote users must identify themselves with a valid user name and password. You can set up accounts for remote users in two different ways:

1. Create a standard account in `/etc/passwd` for the remote user.
2. Create a proxy account for the user. A proxy account maps the remote node and user into a local user account. You create a proxy account as follows:
    1. Define the proxy relationship in the file `/etc/dnet_proxy`. Each record in this file matches a remote node and user name to a local user name by means of the following syntax:

        remote node::*remote user local user # comment*

        For example

        ```
        SYS2::jsmith sys2user # Jim Smith--needs access to Q project files
        ```

        This example allows `jsmith` from remote host `SYS2` to log in locally on the `sys2user` account.
    2. After defining the proxy relationship, create the local account in `/etc/passwd`, if the local account does not already exist. For instance, if the preceding example is to allow jsmith access to the local system, there must be a user account for user name `sys2user` defined in `/etc/passwd`.

        Once the proxy relationship and local account have been defined, the remote user can log in locally using the user name and password defined in the local `/etc/passwd` file.

Of these two approaches, the first, allowing one remote user per account, provides more security because it allows you to audit activities for a particular user: you can determine exactly who is responsible for the activities on your system associated with that user name or UID.

With a proxy account, there are likely to be a number of remote users mapped into a single account. You cannot determine which of those remote users is the individual

responsible for the activities associated with the proxy account.

If you limit proxy accounts to trusted users at remote hosts, then the lack of per-user auditing capability should not be a problem. For remote users of unknown trustworthiness, a standard account in `/etc/passwd` is a good solution.

The advantage of proxy accounts is that they simplify the task of allowing remote users access to your system. With proxy accounts, you do not constantly have to modify the user account files, adding and removing accounts for remote users.

You have great flexibility in how you group proxy accounts.

- Map all users from all remote hosts into a single proxy account. This is easy to manage, but gives you a large pool of users who are all a single person as far as auditing and accountability go.
- Map all users sharing certain characteristics into a single proxy account. You can group users according to the remote host they are from, the project they are working on, the account they are charging to, their level of trust, or some other attribute. This results in multiple proxy accounts, and within an account, a certain degree of user homogeneity.

6.2.3 Restricting DECnet Features

You can deny access to a particular DECnet feature by removing the command and the DECnet objects supporting that feature. As well as removing those features you want to disallow, you can remove those that are not used. For example, if you want to disallow remote file copy, remove the `fal` data-base objects and the file `/usr/bin/dcp`.

DECnet objects must be removed from both the volatile and permanent DECnet object data bases. The syntax for removing an object from the volatile DECnet data base is

ncp clear object *object_name*

The syntax for removing an object from the permanent DECnet data base is

ncp purge object *object_name*

For example, to remove the ability to do DECnet copies, enter the following commands:

```
# ncp clear object fal
# ncp purge object fal
# rm /usr/bin/dcp
```

You might also want to consider disallowing the `TELL` feature, which enables a user to remotely execute a program on a system without logging in to that system.

6.3 Protecting Systems in TCP/IP Networks

The TCP/IP network is widely used in ULTRIX and UNIX environments. At the IP level, the protocol contains only the address of the remote host, unlike DECnet, which includes at the session level both the identity of the remote user and the address of the remote host. Even so, TCP/IP offers you a great deal of control over network access to your system.

6.3.1 Controlling TCP/IP Access to Your System

No remote user can gain TCP/IP access to your system without knowing the user name and password of an account in `/etc/passwd` on your system. By removing the `telnet`, `ftp`, and `tftp` services on your system, you can further restrict remote users to only those from remote hosts that are named in your system's `/etc/hosts` file. This gives you control over which remote hosts can access your system.

Note

The discussion that follows assumes that `telnet`, `ftp`, and `tftp` are not available on your system. A description of how to remove these services appears in Section 6.3.2.

Access in a TCP/IP environment can be regulated by the following files, which you create and maintain locally:

- `/etc/hosts`
- `/etc/hosts.equiv`
- the `.rhosts` files owned by individual users

`/etc/hosts`

A file containing the names of hosts on the network that can communicate with the local host. Each line in the file describes a remote host in terms of the host name and an internet address.

If a user on a remote host that is listed in `/etc/hosts` has the same user name (in the `/etc/passwd` files) on both the remote system and your local system, then the remote user need supply only a password when accessing your local system. Alternately, the remote user can access your system under a local user name that is different from his or her user name on the remote host. To do this, the remote user must supply both the local user name and password.

For more information about `/etc/hosts`, see `hosts`(5) in the *ULTRIX Reference Pages.*

Note that if you are using the Berkeley Internet Name Domain (BIND) service, BIND takes precedence over `/etc/hosts` in regulating which remote hosts have access to your system. See the *Guide to BIND Service* for more information.

`/etc/hosts.equiv`

A file containing the names of remote hosts, the users of which are allowed access to your system without supplying passwords. Each line in the file gives the name of a remote host. If a user on a remote host that is listed in `/etc/hosts.equiv` has the same user name (in the `/etc/passwd` files) on both the remote system and your local system, then the remote user does not need to supply a password when accessing your local system.

If one or more user names follow a host name in `/etc/hosts.equiv`, then those remote users can use the `rlogin` command with the `-l` option to log in on any account on your system. All the remote user needs to do is supply the user name for the account. No password is required. Because user names are rarely secret, a situation like this can badly compromise the security of your system.

A remote user who wishes to have superuser status locally must always identify him or herself with a password.

It is recommended that no entries in appear `/etc/hosts.equiv`. Require every remote user to enter a password when accessing your system. If you must use an `/etc/hosts.equiv` file, then never follow a host name with a user name. Permissions for the file should be 600.

For more information about `hosts.equiv`, see `hosts.equiv`(5) in the *ULTRIX Reference Pages.*

`.rhosts`

A file in the individual user's directory that allows remote users to access that user's account without having to give a password. As with `/etc/hosts.equiv`, if an entry designates only a remote host and not a user name, then anyone from the remote host has access if that remote user has the same user name on both the remote and local hosts.

Because each user can have his or her own version of `.rhosts`, each user has the potential to create a network security problem. It is important that you inform users of the potential danger of `.rhosts`.

The safest course is to have a policy disallowing the use of `.rhosts` files on your system.

If you do allow `.rhosts` files, then each file should designate only the owner of that `.rhosts` file. That is, a user can list remote hosts and his or her own name, to allow the user remote access to his or her own account. Permission for the file should be 600.

Site policy should state that if a local user feels there is a need to grant a remote user access to your system, the local user should talk with you and arrange for the proper entries to be made in `/etc/hosts`.

If you are concerned about the use and contents of `.rhosts` files on your system, you can use the following command to look for the files:

```
# find / -name .rhosts -print
```

For more information about `.rhosts`, see `.rhosts(5)` in the *ULTRIX Reference Pages.*

If your users employ the `ftp` command, then they might have `.netrc` files, which contain login and initialization information supporting autologin across the network for the user. The user's user name and unencrypted password can be part of the information in the `.netrc` file.

Site policy should clearly state that a user's password should not appear in `.netrc`, and the file should have permission 600.

For more information about `.netrc`, see `netrc`(5) and `ftp`(1) in the *ULTRIX Reference Pages.*

6.3.2 Restricting TCP/IP Features

The file `/etc/inetd.conf` describes the configuration for the internet daemon. Each line in the file describes a particular internet service. Some of these services have security implications. The third column of each line identifies the protocol of that service. You can restrict TCP/IP services by placing the comment character, # at the beginning of the line describing the service.

6.3.2.1 Disabling ftp – If your system supports `ftp`, any user on any host on the network can access your system if the user knows a user name and password for your system. To prevent remote users from using the `ftp` command to access your system, remove or comment out the following line in `/etc/inetd.conf`:

```
ftp     stream  tcp     nowait  /usr/etc/ftpd   ftpd
```

6.3.2.2 Disabling telnet – If your system supports telnet, any user on any host on the network can access your system if the user knows a user name and password for your system. To prevent remote users from using the `telnet` command to access your system, remove or comment out the following lines in `/etc/inetd.conf`:

```
telnet  stream  tcp     nowait  /usr/etc/telnetd.gw     telnetd
telnet  stream  tcp     nowait  /etc/telnetd    telnetd
```

6.3.2.3 Disabling tftp – The `tftp` service does not require user identification and provides any remote user with the ability to copy files that allow read-access by `other`. Do not support this service on your system. Remove or comment out the following line from `/etc/inetd.conf`:

```
tftp   dgram    udp     nowait  /usr/etc/fttpd  tftpd   -r /tmp
```

6.3.2.4 Disabling Remote Logins – If you want to disable remote logins to your system, remove or comment out the following lines in `/etc/inetd.conf`:

```
login   stream  tcp     nowait  /etc/rlogind    rlogind
telnet  stream  tcp     nowait  /usr/etc/telnetd.gw     telnetd
telnet  stream  tcp     nowait  /etc/telnetd    telnetd
```

Note that the second line in the preceding list supports TCP/IP on DECnet and is automatically installed when DECnet is installed. If you later install or reinstall DECnet, you will need to go back into `/etc/inetd.conf` and again remove or comment out the line

```
telnet  stream  tcp     nowait  /usr/etc/telnetd.gw     telnetd
```

6.4 Protecting NFS (Network File System) Files

NFS allows simplified sharing of files across an internet network. NFS achieves this by providing the user with transparent network access to files, making it appear as though remote files are mounted locally. The host system on which the files are stored is called the server; the systems that use those files are the clients. Local access to files on a remote server is controlled locally. Permissions for the files can be assigned locally and apply only to the local host.

6.4.1 Exporting File Systems

To make your system an NFS server, that is, to make files on your system available on remote hosts, list the appropriate file systems in `/etc/exports`. Each line in that file names a file system for export, any qualifications about the export, and, optionally, the hosts to which the file system can be exported. The format of a line in `/etc/exports` is as follows:

file_system_name [*options*] *export_target_hosts*

Consider the following export restrictions when exporting a file system:

- Denying write access to the exported file system.

  The `-o` option marks the file system for read-only export.

- Denying superuser access by the client to the exported file system.

  The `-r` option lets you equate a UID for client users who are superusers on their host systems. This equated UID applys only to the exported file system. The default value is `-2`, which means that client superusers do not have superuser status with the exported file system. If you export a file system with `-r=0`, then client super users have superuser status with the exported file system.

  Avoid setting `-r=0` unless it is necessary and you know that all client superusers with access to the exported file system are trusted.

- Denying clients the ability to change directories above the local root of the exported file system.

  The command `cd ..` allows a user to change his or her working directory, moving up one directory toward `root` in the file system hierarchy. By repeated invocations of `cd ..`, the user can change the working directory to `root` (/).

  On a client system with an exported file system, there is the `root` of the client, where the exported file system is locally mounted, but there is also the `root` of the server, where the file system is actually mounted. By repeated invocations of `cd ..`, a client user can move above the local root and into the file system hierarchy of the server.

  The `-n` option prevents a client from using the `cd` command to move above the local `root` of the client system.

  It is recommended you use the `-n` option for all exported file systems.

- Allowing export only to remote systems that need access.

  Rather than exporting a file system to all hosts on the network, follow the file system name and options with the names of remote hosts or Yellow Pages host groups. This limits export of the file systems to only those hosts.

When you mount a file system for export, use the `mountd` command with the `-i` option. The `-i` option enables verification of the Internet addresses against the systems in the `/etc/hosts` file, the YP hosts map, or the Internet domain. For more information, see the *Introduction to Networking, Guide to the BIND/Hesiod Service,* and `mountd`(8) in the *ULTRIX Reference Pages.*

For a detailed description of NFS and how to implement it, see the *Guide to Network File System.*

6.5 Managing Network Accounts

To simplify the job of managing a distributed environment, it is recommended you use the BIND/Hesiod service and Kerberos.

BIND/Hesiod provides a central source where members of a network can look up needed address and user information. For example, BIND/Hesiod can provide a single source of information about user accounts, such as passwords, which is easier to update and maintain than multiple individual files of user accounts on each of the members in a network. BIND/Hesiod is compatible with the secure user authentication provided by the ULTRIX UPGRADE and ENHANCED levels. Kerberos offers secure verification of user and workstation identities in the distributed environment, creating a high degree of confidence in identity authentication.

For information about BIND/Hesiod, see the *Guide to BIND/Hesiod Service*. For information about Kerberos, see the *Guide to Kerberos*. For information about managing user accounts in a network of UPGRADE or ENHANCED systems, see Section 3.4

A service similar to that offered by BIND/Hesiod is provided by Yellow Pages. However, Yellow Pages cannot be used with the UPGRADE or ENHANCED user authentication levels, and it does not provide the same degree of network authentication security that Kerberos does.

The two security-sensitive files that Yellow Pages provides centralized handling for are

- `/etc/hosts`
- `/etc/passwd`

A system that is part of a network using the Yellow Pages feature can alter its local `/etc/hosts` and `etc/passwd` files to indicate that information, such as a user's password, is found in the Yellow Pages data base.

If you choose to use Yellow Pages, do the following:

- Keep all superuser accounts in `/etc/passwd` on your system, rather than in the Yellow Pages data base.
- Make certain that each account in the Yellow Pages data base has a unique UID and a unique user name.

For a detailed description of Yellow Pages and how to implement the feature, see the *Guide to Yellow Pages Service*.

6.6 Restricting the Environment of Remote Users

Section 3.3.1 presents an example of the establishment of a restricted environment for a user. A restricted environment is one that places limits on a user's access to files or other system resources. You can force a user into a restricted environment as follows:

- Develop a script or program that places the desired restrictions on the user's activities.
- Designate that program or script as the user's startup program in the user's account in `/etc/passwd`.

When you have remote users who require only limited access to your system, it is prudent to place them in a restricted environment.

Starting and Configuring a Secure System 7

This chapter describes security considerations for starting a system and how to install and configure the system security software.

The major subjects covered in this chapter are

- Starting a system securely
- Installing and configuring the security software

7.1 Starting a System Securely

When you boot the system, you have two choices for the mode of the newly booted system:

- single-user mode
- multiuser mode

7.1.1 Starting a System in Single-user Mode

When a system establishes itself in single-user mode, only the system console is active; no terminal devices have access to the system. No login is required at the system console; it is owned by root and any commands issued from it have privileged status. Anyone with access to the console when the system is in single-user mode can gain complete access to the system. Because of this, the system console must always be physically secure when the system is in single-user mode.

7.1.2 Starting a System in Multiuser Mode

As a system establishes itself in multiuser mode, it uses several files to determine the startup processes and the system configuration:

`/etc/rc`	script controlling system startup processes
`/etc/rc.local`	site-specific startup information
`/etc/fstab`	instructions for mounting the file systems
`/etc/ttys`	instructions for initializing and configuring terminal devices

Because these files can affect the security of the system, they must be write-protected from the user community.

Once multiuser mode is established, the terminals are active and the user account system is functional; only authorized users can log in to the system. Login is required at the system console and grants superuser status only to users with a UID of 0.

Before bringing a system up in multiuser mode, configure system security to provide the protection you want as follows:

- Configure the user account system to provide needed security:
  - Make certain that only authorized users have accounts.
  - Grant superuser status only to trusted users with a need for the privilege.
  - Configure the the enhanced authentication features--password aging, password generation, minimum password length, and others, if you elect to use them.
- Configure the accounting subsystem, if you are using it:
  - Designate auditlog backup destinations for logging data.
  - Assign events to the system audit mask and user audit masks, so that appropriate system and user activities are logged.
- Assign appropriate owners, groups, and permissions to sensitive files and directories.

7.2 Installing and Configuring Enhanced Security Features

Before you can take advantage of the added security offered by trusted path, the auditing subsystem, and the enhanced authentication features, you must install and configure the security software. The software has been designed to make this process quick and efficient.

To provide the optimal security for your system, you should perform the tasks described in this section while your system is in single-user mode, before you bring up multiuser mode.

There are four steps to installing and configuring security on your system:

1. Installing the security software subset
2. Determining the appropriate configuration
3. Invoking the security setup script and configuring the security features
4. Editing the security configuration file, `/etc/svc.conf`.

7.2.1 Installing Enhanced Security Software during System Software Installation

1. To install the ULTRIX security enhancements at the time of system software installation, the software subset ULTSEC040 must be selected along with the other software subsets being installed.

   For more information about loading software subsets, refer to the *Advanced Installation Guide*.

2. After software installation is complete, you will want to select the security configuration for your system. Use the the security setup script to do this. Before you invoke the script, the following must be true:

   1. The software subset ULTSEC040 is installed.

2. The system is in single-user mode.
3. You are logged in as root.
4. The `/usr` file system is mounted.
5. You have decided upon the security configuration appropriate to the system, and you know the answers to the following questions:
   - Do you want the trusted path feature?
   - Do you want the audit subsystem enabled, and do you know which audit events you want logged?
   - Do you want improved login security? If so,
     - Do you want BSD, UPGRADE, or ENHANCED security level?
     - What is to be the maximum lifetime for passwords, in days? (A long password lifetime increases the chances of the password being compromised. A short password lifetime increases the chances that users will forget their passwords or write them down to avoid forgetting them, a potential security problem. A value between 60 and 120 days should prove workable. The maximum password lifetime should be no longer than 365 days.)
     - What is to be the minimum lifetime for passwords, in hours? (This is intended to discourage users who have been forced to change to new passwords from quickly changing back to their previous, familiar passwords.)

If you are not sure about the answers to any of these questions, see the sections indicated in the following table:

Table 7-1: Where to Find Information for Configuration Planning

Topic	Section
trusted path	Section 3.1.2.3
audit subsystem	Section 4.2
audit events	Sections 4.3, 4.7
improved login security	Section 3.2
BSD, UPGRADE, and ENHANCED security levels	Section 3.2.3.

3. After you complete the needed planning, invoke the security setup script by entering the following command:

```
# /usr/etc/sec/secsetup
```

For a detailed description of using the security setup script, see Section 7.2.3.

The script asks you a series of questions about security configuration. These questions correspond to the bulleted questions in entry number 5 of the preceding list.

After you have completed the configuration process, the system kernel might need to be rebuilt in order to install the configuration. If a kernel rebuild is necessary, you are given the choice of having the script rebuild it for you, or of rebuilding the kernel manually. The security setup script then exits.

4. If a kernel rebuild was necessary, after the rebuild you must move the new kernel to `/vmunix` and reboot the system.

   If you later decide to change the security configuration, you can re-invoke the script at any time to alter the setup.

5. At this point you might want to edit the the file `/etc/svc.conf` and alter the values for the grace period for expired passwords and the minimum required password length. The values shipped with the software are a 7-day grace period for expired passwords, and 6 characters as the minimum length for passwords. The value determining the expired password grace period appears in the field to the right of SOFTEXP. The value for minimum password length appears in the field to the right of PASSLENMIN.

   For more information about the file `/etc/svc.conf`, the grace period for expired passwords, and the minimum password length, see Section 3.3.4.

7.2.2 Installing Enhanced Security Software after System Software Installation

If you decide to install the ULTRIX security enhancements at some time after the general installation of system software, the procedure you follow is similar to the one given in the previous section. Read items 2-4 in the main list of Section 7.2.1 and then continue reading here.

1. Load ULTSEC040 with the `setld` command.
2. Plan the security configuration appropriate to your installation.
3. Invoke the security setup script by entering the following command:

   ```
   # /usr/etc/sec/secsetup
   ```

   Based on the planning you did in step 2, answer the questions asked by the script.

   For a detailed description of using the security setup script, see Section, 7.2.3.

   After you have completed the configuration process, the system kernel might need to be rebuilt in order to install the configuration. If a kernel rebuild is necessary, you are given the choice of having the script rebuild it for you, or of rebuilding the kernel manually. The security setup script then exits.

4. If a kernel rebuild was necessary, after the rebuild you must move the new kernel to `/vmunix` and reboot the system.
5. At this point you might want to edit the the file `/etc/svc.conf` and alter the values for the grace period for expired passwords and the minimum required password length. The values shipped with the software are a 7-day grace period for expired passwords, and 6 characters as the minimum length for passwords.

7.2.3 Security Setup Script

After you invoke the script with the `/usr/etc/sec/secsetup` command, the setup script first informs you that you must have superuser status and the `/usr` file system must be mounted.

The script then asks you a series of questions about security configuration. To terminate the security setup script during the question period, enter the CTRL/C sequence. No changes are made to the configuration, and you can restart the process at a later time.

7.2.3.1 Security Setup in a BIND/Hesiod or Kerberos Environment – If your system is part of a network of computers and BIND/Hesiod with Kerberos are used to manage the distributed environment, then the order in which you install and configure BIND/Hesiod, Kerberos, and the enhanced local security features is important.

To allow for secure distributed management of user names and passwords, you must incrementally increase the security level of your system, starting with the BSD level, then moving to UPGRADE level and only then to ENHANCED level. Between each move up in security levels, you must make certain alterations to the configuration of BIND/Hesiod and Kerberos.

- Initially, use the security setup script only to set the security level for your system to BSD level. The procedure is described in Section, 7.2.3.2.
- Exit the security setup script and follow the procedure given in the *Guide to Kerberos* for installing a distributed environment for systems at the ENHANCED level.

For a detailed description of installing and configuring Kerberos and BIND/Hesiod in an environment where you want to run UPGRADE or ENHANCED security level, see the *Guide to Kerberos*. For instructions on installing and configuring BIND/Hesiod, see the *Guide to BIND/Hesiod Service*.

7.2.3.2 Security Setup in a Non-networking Environment – If you are not working with a system that is part of a network using BIND/Hesiod or Kerberos, then you can use the security setup script in a straightforward fashion to configure system security.

The security setup script asks for the following information:

- Do you want to enable trusted path? (y/n)
- Do you want to enable security auditing? (y/n)

  The script informs you that the system default for security auditing is a full audit of all events. Because this can adversely affect system performance, you might want to tailor the security auditing system to record only the security events that you want to audit.
- Do you want to edit the list of system audit events now? (y/n)

  Answer y to tailor system auditing to your special needs. If your `EDITOR` environment variable designates an editor, then that editor is invoked, and the audit events list is displayed for editing. Otherwise, the `vi` editor is invoked. If `vi` is not available, `ed` is invoke.

- Do you want to enable the enhanced login functionality? (y/n)

  Answer **y** to set the security level. If you answer **y** , the script briefly explains the different security levels (BSD, UPGRADE, and ENHANCED), and then prompts you to enter the level you desire.

- Enter the new security level

  If you choose BSD level, the script warns you that if you are already running a system in something other than BSD level, you will have to reassign all user passwords.

  If you choose UPGRADE or ENHANCED level, the script informs you that a new authorization database will be built using the information in the `/etc/passwd` file.

  If you choose UPGRADE, the script informs you that new passwords must be assigned for each user account. Until this is done, the accounts will have invalid passwords.

  If you choose UPGRADE or ENHANCED level, the script then does the following:

  - Informs you that each account has a maximum and minimum password lifetime. When the maximum password lifetime has expired the password must be changed or the account will become unusable. If the maximum password lifetime is set to 0 (zero), the password will not expire.
  - Prompts you for the maximum password lifetime.

- Enter maximum password lifetime in days

  You are then informed that the minimum password lifetime determines how often users can change their passwords. If set to 0 (zero), there is no minimum password lifetime and users can change their passwords at any time.

  You are prompted for the minimum password lifetime in hours:

- Enter minimum password lifetime in hours

  This ends the configuration questions of the security setup script. The script informs you of the options you have chosen and asks you whether you want the changes to take effect. You are given the choice of making the changes effective (**y**), restarting the script and beginning over (**n**), or exiting the script (**exit**).

- Is the selected set of security features correct? (y/n/exit)

  If you answer **y** , the script informs you that it is updating the configuration file to reflect your changes.

  If you chose UPGRADE or ENHANCED levels, you are told approximately how long it will take to build the authentication data base, `/etc/auth`, from the current `/etc/passwd` file. You are also warned that Yellow Pages does not serve `/etc/auth` entries. Refer to Sections 6.5 and 3.4 for information about BIND/Hesiod support of the authentication database in a distributed environment.

  If any UIDs in the current `/etc/passwd` file are illegal under the rules for the authentication data base, you are warned of the fact and the offending account is identified.

The installation of the security features is then complete.

If you chose the trusted path or audit subsystem options, you are told that a kernel rebuild is required to reconfigure the system's kernel. You are given the option of having the security setup script initiate the rebuild or of exiting the script and rebuilding manually.

- Would you like the installation process to rebuild your kernel? (y/n)

  If you answer **y** , the script rebuilds the kernel and exits. If you answer **n** , you are reminded to manually rebuild your system and reboot, then the script exits.

7.2.4 Enabling Password Expiration on Workstations

In order to support password expiration on workstations, the entry for the workstation in `/etc/ttys` must include the `/usr/bin/login` command with the `-e` option. If more than one option is designated for `/usr/bin/login`, then the `-e` option must be the last option specified.

The security setup script automatically enables password expiration for those workstations listed in `/etc/ttys` at the time the setup script is executed. If you later add workstations to `/etc/ttys`, you will need to manually include the `/usr/bin/login` command with the `-e` option.

Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Y

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-234-1998 using a 1200- or 2400-baud modem from anywhere in the USA, Canada, or Puerto Rico. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital Subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	————	Local Digital subsidiary or approved distributor
Internal*	————	SSB Order Processing - WMO/E15 *or* Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

* For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

ULTRIX
Security Guide for Administrators
AA-PBKTA-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

What would you like to see more/less of? _____

What do you like best about this manual? _____

What do you like least about this manual? _____

Please list errors you have found in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

What version of the software described by this manual are you using? _____

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Email _____ Phone _____

Do Not Tear – Fold Here and Tape

digital™

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST–CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OPEN SOFTWARE PUBLICATIONS MANAGER
ZKO3–2/Z04
110 SPIT BROOK ROAD
NASHUA NH 03062–9987

Do Not Tear – Fold Here

Cut
Along
Dotted
Line

Reader's Comments

ULTRIX
Security Guide for Administrators
AA-PBKTA-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

What would you like to see more/less of? _____

What do you like best about this manual? _____

What do you like least about this manual? _____

Please list errors you have found in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

What version of the software described by this manual are you using? _____

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Email _____ Phone _____

Do Not Tear – Fold Here and Tape

digital™

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST–CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OPEN SOFTWARE PUBLICATIONS MANAGER
ZKO3–2/Z04
110 SPIT BROOK ROAD
NASHUA NH 03062–9987

Do Not Tear – Fold Here

Cut
Along
Dotted
Line