

# *Securing System Access*

---

## **Student Guide**



Sun Microsystems Computer Corporation  
Technical Education Services  
MS UMIL07-14  
2550 Garcia Avenue  
Mountain View, CA 94043  
U.S.A.

Part Number SA-285  
Revision B, July 1993

© 1993 Sun Microsystems, Inc.—Printed in the United States of America.  
2550 Garcia Avenue, Mountain View, California 94043-1100 U.S.A.

All rights reserved. This product and related documentation are protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or related documentation may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Portions of this product may be derived from the UNIX® and Berkeley 4.3 BSD systems, licensed from UNIX System Laboratories, Inc. and the University of California, respectively. Third-party font software in this product is protected by copyright and licensed from Sun's Font Suppliers.

#### RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the United States Government is subject to the restrictions set forth in DFARS 252.227-7013 (c)(1)(ii) and FAR 52.227-19.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

#### TRADEMARKS

Sun, Sun Microsystems, the Sun logo, Solaris, NFS, and OpenWindows are trademarks or registered trademarks of Sun Microsystems, Inc. UNIX and OPEN LOOK are registered trademarks of UNIX System Laboratories, Inc. All other product names mentioned herein are the trademarks of their respective owners.

All SPARC trademarks, including the SCD Compliant Logo, are trademarks or registered trademarks of SPARC International, Inc. SPARCstation, SPARCserver, SPARCengine, SPARCworks, and SPARCcompiler are licensed exclusively to Sun Microsystems, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK® and Sun™ Graphical User Interfaces were developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

X Window System is a trademark and product of the Massachusetts Institute of Technology.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

# Contents

---

<b>Basic Security Features</b> .....	1-1
Introduction .....	1-2
Basic Security Features .....	1-3
What Is a User Account? .....	1-4
What Is a User Group? .....	1-6
The /etc/shadow Database .....	1-7
System Administration Databases.....	1-8
Identifying Special Users and Groups .....	1-9
File and Data Security Review .....	1-11
How File and Directory Access Is Determined .....	1-12
Identifying and Setting Permissions Review .....	1-13
Setting Default File and Directory Permissions Review .....	1-14
The setuid and setgid Permissions .....	1-15
Using setuid and setgid Permissions .....	1-16
Identifying and Setting Sticky Permission .....	1-18
Summary .....	1-19
Exercise 1-1.....	1-20
<b>Creating User Accounts</b> .....	2-1
Introduction .....	2-2
Using Administration Tool.....	2-3
Exercise 2-1.....	2-4
Starting the OpenWindows Environment.....	2-5
Starting Administration Tool .....	2-6
Loading the Group Database .....	2-7
Accessing the Add Entry Form.....	2-8
Creating a New Group .....	2-9
Starting the User Account Manager .....	2-10
Accessing the Add User Form .....	2-11
Filling in the Add User Form .....	2-12
Verifying the New Account.....	2-16
Using the New Account.....	2-17
Examining the /etc/passwd File .....	2-18
Examining the /etc/group File .....	2-19



Creating User Accounts Manually .....	2-20
Using the su Command .....	2-21
Summary .....	2-22
<b>User Account Maintenance .....</b>	<b>3-1</b>
Introduction .....	3-2
Password Requirements .....	3-3
Password Aging Features .....	3-4
Exercise 3-1.....	3-6
Modifying an Existing User Account.....	3-7
Examining the User Account Manager Edit Choices .....	3-8
Selecting a User Account to View or Modify.....	3-9
Examining the /etc/shadow File .....	3-10
Using the Copy User Option.....	3-12
Locking an Account.....	3-13
Deleting an Account .....	3-14
The passwd Command.....	3-16
Exploring the /etc/default Directory .....	3-17
The /etc/default/passwd File.....	3-18
The /etc/default/login File.....	3-19
The /etc/default/login File.....	3-20
Using Restricted Shells.....	3-21
Summary .....	3-22
<b>Answer Key.....</b>	<b>A-1</b>
Lesson 1: Basic Security Features.....	A-2
Lesson 2: Creating User Accounts .....	A-4
Lesson 3: User Account Maintenance .....	A-5
Appendix A: ASET .....	A-6
<b>Using the Automated Security Enhancement Tool (ASET).....</b>	<b>B-1</b>
Introduction .....	B-2
ASET Security Options.....	B-3
ASET Tasks .....	B-4
ASET Terminology .....	B-5
The aset Command .....	B-6
Checking on Task Status .....	B-7
ASET Report Files .....	B-8
Changing Security Levels .....	B-10
Restoring Pre-ASET System Files .....	B-11
Exercise A-1.....	B-12

# *Basic Security Features*

---



## Objectives

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

- Recall the format of the `/etc/passwd` and `/etc/group` files and describe their importance to system security.
- Describe the purpose of the `sysadmin` group.
- Recall how to set file permissions.
- Recall how to display and change default permissions.
- Explain how the `setuid` and `setgid` permissions relate to system security.
- Describe how to identify and set the sticky permission.
- Describe how the sticky permission protects files and directories.

## References

*SunOS 5.1 Setting Up User Accounts, Printers, and Mail*,  
Chapter 1, "Setting Up User Accounts and Groups," and  
Chapter 2, "Administering User Accounts and Groups"

*SunOS 5.1 Administering Security, Performance, and Accounting*,  
Chapter 1, "Introduction to Security"



## Introduction

Users are to system administrators what air traffic is to air traffic controllers—the source of both headaches and paychecks.

User accounts (including passwords) are the first line of defense in securing systems. Allowing system access by creating user accounts is one of the most regularly performed system administration activities.

File and data security provide another level of security beyond requiring user logins and passwords.

Several additional file and data security features are included—the importance of directory permissions and several additional permission bits beyond the basic read, write, and execute bits that are important to system security.

This lesson also presents information about users and groups with special powers, and it contains suggestions about not abusing those powers.



## Basic Security Features

To protect the system from intruders, the Solaris® distributed computing environment offers two basic security features: login and file security.

### Login Security

In order to use a Sun Workstation®, a potential user must present a public login name and a private password, which is checked against a list of login names and passwords known to the system. (This process is also referred to as *user authentication*.)

### File and Data Security

Upon gaining access to the system, a user's access to files and directories is controlled by permissions. Users can set or change permissions on files and directories they own. (File and data security is also referred to as *discretionary access control*.)

	read	write	execute
owner			
group			
other			

The components of a user account as well as the system files that pertain to user account management are described first.



## What Is a User Account?

*user password  
info.*

A user account consists of four main components: `/etc/passwd` and `/etc/shadow` entries, a home directory, and startup files.

### The `/etc/passwd` Database

A list of essential user information is stored in the `/etc/passwd` database. Each `passwd` record contains the following seven fields:

```
user-name:x:uid:gid:comment:home-dir:login-shell
```

The fields are:

*user-name* This field represents the user's login name. For clarity, it should be related to the user's real name, and it should also be unique. The field is restricted to eight characters in length and should contain no uppercase characters.

x This field is a placeholder for the user's encrypted password, which is stored in the `/etc/shadow` file.

*uid* This field contains the user identification number that is used by the operating system to identify the user. UID numbers for users usually range from 100 to 60000.

Values 0 through 99 are reserved for system accounts. UID 60001 is reserved for the `nobody` account. UID 60002 is reserved for the `noaccess` account.

Duplicate UIDs are legal but should be avoided. If two users have the same UID, they have identical access to the files each user creates.

*gid* This field contains the group identification number that is used by the system to identify the user's primary group. GID numbers for users usually range from 100 to 60000.



## What Is a User Account?

### The `/etc/passwd` Database (continued)

<i>comment</i>	This field usually contains the user's full name. This field is also referred to as the <code>gcos-field</code> for historical reasons.
<i>home-dir</i>	This field contains the path name of the user's home directory.
<i>login-shell</i>	This field defines the user's default login shell, which can be <code>/bin/sh</code> , <code>/bin/csh</code> , or <code>/bin/ksh</code> .

### A Home Directory

This directory is set as the user's current directory when the user first logs in and typically houses the bulk of the user's files.

### Initialization (Startup) Files

The initialization files control how the user's environment is set up when the user accesses the system. They are set up by the system administrator when the user account is created, and usually start with a `.` (dot) in order to be transparent to the user.

Files in the `/etc/skel` directory can be used as a prototype for the user's initialization files.



## What Is a User Group?

All files and directories are associated with a group as well as an owner. All users are members of one or more groups. A user's primary group is specified by the `gid` stored in the `passwd` database. The `/etc/group` database specifies any additional groups to which a user belongs. A user can have one primary group and up to 15 secondary groups.

In addition, while each `passwd` record specifies both a UID number and user name, it only specifies a user's GID, not the name of the group. The correspondence between a GID and a group name is established by the `/etc/group` database.

### The `/etc/group` Database

Each group database record contains the following four fields:

*group-name* : *password* : *gid* : *user-list*

The fields are:

- |                   |  |
|-------------------|--|
| <i>group-name</i> | This field contains the group name, which can be up to eight characters in length. |
| <i>password</i>   | This unused field is for a group password.   |
| <i>gid</i>        | This field contains the group identification number.                               |
| <i>user-list</i>  | A comma-separated list of users to identify their secondary group membership.      |

*A user can be a part of at max 16 groups*



---

## The `/etc/shadow` Database

The `/etc/shadow` file is used to store a user's encrypted password and related password information. This ASCII file can only be read by the superuser for security reasons.

The contents of this file is described in a later lesson.



# System Administration Databases

The `passwd` and `group` databases are two of many databases that are important to system administration. Most of these databases are either implemented as files within the `/etc` directory or are served by the Network Information Services (NIS) or Network Information Services Plus (NIS+) naming service.

Administration Tool is used to administer the local `/etc` files or the NIS+ databases, which is covered in the next lesson.

## The `/etc` Directory

System administration information, including `passwd` and `group` information that is only required by a single machine, is stored as ASCII files within the `/etc` directory. Local password information is stored in the `/etc/passwd` file, and local group information is stored in the `/etc/group` file.

## The NIS Product

The NIS product makes system administration information available over the network. This reduces the need for redundant information to be stored on many machines and, therefore, reduces the opportunities for this information to become inconsistent.

## The NIS+ Environment

The Network Information Service Plus environment provides the same features as the NIS product with additional security mechanisms and supports hierarchical domains.

# Identifying Special Users and Groups

Several user IDs and group IDs are reserved by the system for special accounts.

## The Superuser Account

The user with a UID of 0 (often called the *superuser* or *root* user) is granted read and write access to all files stored on a system's local disk and can send kill signals to all processes under the control of the system's CPU. Up until now you have been doing all of your work as the superuser. You will create a regular user account in the next lesson. For safety and security reasons it is important to do most of your work as a regular user and only use the superuser account when required.

## Reserved UIDs

The UIDs between 0 and 99 are reserved for special accounts used by the system such as the `lp` and `listen` users.

The reserved system accounts are used to provide exclusive file access for certain system services such as the `lp` print service daemon.

The `nobody` account is used for securing NFS® resources. When a user is logged in as superuser on an NFS client and attempts to access a remote file resource, the UID is changed from 0 to the UID of `nobody` (60001), who gets the same access permissions as those defined for the world (or everyone else).

## The `sysadmin` Group *→ Group 14*

Members of the `sysadmin` group are allowed to use the Administration Tool, which essentially gives them superuser permissions. When there is no `sysadmin` group, only the superuser can use the `admintool` utility. It is, however, more secure to set up and use the `sysadmin` group. This gives you another way to avoid logging in as the superuser.



---

# Identifying Special Users and Groups

## Other Reserved GIDs

The GIDs between 0 and 100 are reserved for special accounts used by the system such as the `bin`, `other`, and `staff` user accounts. The reserved GIDs are also used to provide exclusive group file access.

# File and Data Security Review

## Permissions Summary

The owner of a file or directory, or the superuser, can set or change permissions on that file or directory.

### Read Permission

Read permission allows the contents of a file to be read and displayed. Viewing a file with an editor or other command or utility requires read permission.

Together with execute permission, read permission allows a directory's contents to be listed.

### Write Permission

Write permission allows the contents of a file to be modified. Changing the contents of a file with an editor or other command or utility requires write permission.

Together with execute permission, write permission allows a directory's contents to be modified: that is, you can add or remove entries.

### Execute Permission

Execute permission allows a file containing a shell script or executable code to be run.

Referred to as *access* permission for directories, this permission is necessary for any type of directory access.

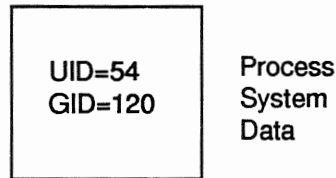
### None

None means there is no file or directory access.



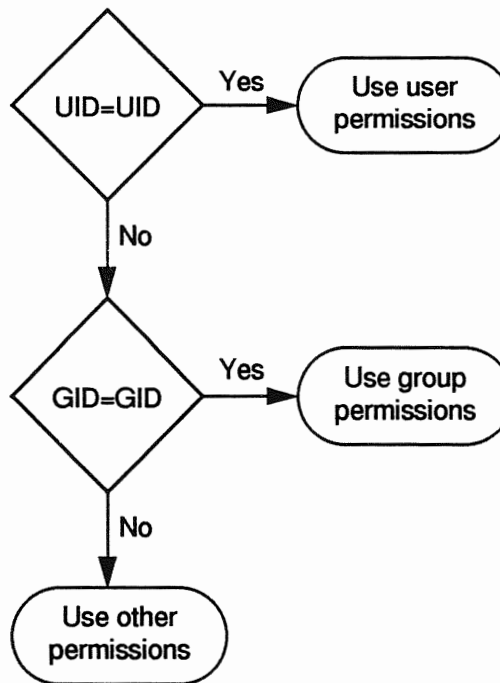
# How File and Directory Access Is Determined

All processes have a UID variable with a single value and a GID variable with one or more values.



All file and directory inodes contain, along with other data, a UID, a GID, and a set of permissions.

When a process attempts to read, write, or execute a file, the process' system data information is compared to the file's inode information.





# Identifying and Setting Permissions Review

## Identifying File and Directory Permissions

The following `ls -l` output for an ordinary file has read, write, and execute permissions for owner, execute permission for group members, and no permissions for others.

```
# ls -l file.a
-rw-r--r-- 1 rimmer  staff  4601 May 27 08:58 file.a
#
```

The only difference between directory and file permission listings is that directories always have a `d` instead of a `-` (dash) in the first field.

```
# ls -ld programs
drwxr-xr-x 2 rimmer  staff  37 May 27 08:57 programs
#
```

## Setting File and Directory Permissions

File permissions can be set with the `chmod` command using either symbolic or numeric notation.

```
# chmod u=rwx,g=rx,o= hello
# chmod 750 hello
```

The steps for setting normal directory permissions are the same as the steps for setting file permissions.

*ls -la*  
*long all size in kb*



# Setting Default File and Directory Permissions Review

Files and directories are created with a set of default permissions. Users' file creation permissions are determined by their `umask` value.

## Determining Your `umask` Value

Users can determine the value of their `umask`.

```
$ umask  
0022
```

## Interpreting the Meaning of a `umask` Value

A `umask` value is the inverse of a user's default file creation permissions. (Ignore the first digit, which indicates that the value is in octal.) For regular files the value is subtracted from 666. For directories and object files created by compilers, the value is subtracted from 777.

A `umask` of 0022 means that regular files will be created with a mode of 644 (read and write by owner; read only for group members and the others) while directories and executables will be created with a mode of 755 (read, write, and execute by owner; read and execute for group members and others).

## Setting a User's `umask` Value → Can be set in *.login*

Users can set the value of their `umask`.

```
$ umask 026
```

The above command would only be valid for the duration of the login session (in the window where it was issued). Users typically set their `umask` in their `.profile` or `.cshrc` file. The system-wide Bourne and Korn shell default of 022 is set in the `/etc/profile` file. The C shell's default is also 022. The `/etc/skel/local.cshrc` also contains a `umask 022` value.



## The `setuid` and `setgid` Permissions

The owner and the superuser can also set `setuid` and `setgid` permissions on a file and `setgid` permissions on a directory. These special permissions modify the effects of normal permissions.

### Executable Programs

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

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

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

### Directories

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



## Using setuid and setgid Permissions

### Identifying setuid and setgid Permissions

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

```
$ ls -l /bin/passwd /etc/passwd /etc/shadow
-r-sr-sr-x  1 root    sys      22208 Mar 27 06:21 /bin/passwd
-r--r--r--  1 root    sys      1043  May 26 09:57 /etc/passwd
-r-----   1 root    sys      529   May 26 09:57 /etc/shadow
$
```

These permissions explain why users can change certain fields in the `/etc/passwd` and `/etc/shadow` files when using the `passwd` command to change their passwords.

### Setting setuid and setgid Permissions

These permissions are set with the `chmod` command using either symbolic or numeric notation for files. Numeric notation uses the left-most number to refer to these special permissions:

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

```
# chmod 4750 setuid_program
# chmod 2750 setgid_program
```

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

```
# chmod gas some_directory
```

u	g	o	o
u=4	r=4	r=4	r=4
G=2	U=2	U=2	U=2
sticky=1	X=1	X=1	X=1

---

## The Sticky Bit

The sticky permission bit was used to cause a program's executable image to remain in swap space. This is no longer the case.

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

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

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

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



# Identifying and Setting Sticky Permission

## Identifying Sticky Permission

The sticky bit is displayed as the letter `t` in the execute field for everyone else (a `T` is an undefined bit state indicating that the sticky bit is on and execution is off).

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

## Setting Sticky Permission

The sticky bit is set using symbolic or octal notation.

```
# chmod 1777 sticky_directory
# chmod a=rwt sticky_directory
```

## Summary

In this lesson, you learned that:

- The basis of system security is login security and file security, which are built on user accounts consisting of a `passwd` record, a `shadow` record, home directory, and initialization files.
- A user group is defined by a group record and all users are members of one or more groups.
- Both `passwd` and `group` records, as well as many other system administration databases, are implemented as ASCII files in the local `/etc` directory or as part of the NIS or NIS+ services.
- There are several special users and groups, the most important being the superuser and the `sysadmin` group.
- File and directory access is determined by the match between process and inode UIDs and GIDs and inode permission bits.
- File, directory, and the special `setuid`, `setgid`, and sticky permissions are set with the `chmod` command.



## Exercise 1-1

Write down the answers to the following questions.

1. List the components of a user account.

---

---

---

---

2. Describe the format of a passwd record.

---

---

---

---

---

---

---

---

3. List the reserved UIDs.

---

---

---

4. Which directory contains files that may be used as a prototype for a user's initialization files?

---

5. What is the purpose of the group database?

---

---

6. What is the range of group numbers typically used for user accounts?

---

7. Name the most important user account name and group name.

---

8. What command is used to determine and set default file creation permissions?

---



## Exercise 1-1

9. Name the special permissions that modify the effects of normal permissions and describe the characteristics of each.

---

---

---

---

10. What command will set the `setuid` bit of a file called `my_file` with permissions of 755?

---

11. What command will set the `setgid` bit of a directory called `our_project` with permissions of 770?

---

12. What is the benefit of setting the sticky bit on a directory?

---

---

---

---

13. Describe how the system determines directory and file access.

---

---

---

---



# *Creating User Accounts*

---

2

## Objectives

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

- Use Administration Tool to create a new group and a user account.
- Use the appropriate default environment files from `/etc/skel` to set up a user environment.
- Use the `id` command to determine your user ID number and group ID number.
- List the four steps to manually add a user account.

## References

*SunOS 5.1 Setting Up User Accounts, Printers, and Mail,*  
Chapters 1 and 2



## Introduction

This lesson covers the procedures required to create user accounts and groups using Administration Tool, Sun's graphical administration utility.

## Using Administration Tool

Administration Tool is a system administration utility with a graphical interface that allows administrators to maintain:

- System database files
- Printers
- User accounts
- Hosts

Administration Tool is run under the OpenWindows™ environment.

In this module Administration Tool is used to modify system files on the local system only. Using Administration Tool to modify the NIS+ database is covered in another module.



## Exercise 2-1

The following extended lab procedure is intended to provide familiarity with Administration Tool for adding users and groups, and experience with other security tasks. In this exercise you will:

- Start the OpenWindows environment.
- Start Administration Tool.
- Use the User Account Manager in Administration Tool to add a user account, specifying the following information:
  - User name and UID
  - Primary GID
  - Secondary GID
  - Real name as a comment
  - Login shell
  - Password
  - Home directory information
- Use the Database Manager in Administration Tool to add the `sysadmin` group.
- Log in as the new user.
- Examine the `/etc/passwd` file.
- Examine the `/etc/group` file.
- Use the `su` command.

## Starting the OpenWindows Environment

Set up the superuser environment so that the OpenWindows environment is automatically started when you become superuser.

1. Log in as the superuser.
2. Copy the `local.profile` file from the `/etc/skel` directory into the superuser's home directory.

```
# cd
# cp /etc/skel/local.profile .profile
```

3. Add the `/usr/openwin`, `/usr/sbin`, and `/sbin` directories to root's `PATH` value in the `.profile`. Make sure that the `/usr/sbin` directory precedes the `/usr/ucb` directory.

```
PATH=/usr/openwin:/usr/sbin:/sbin:/usr/bin:/usr/ucb
:/etc:.
```

4. Execute the `.profile` to start the OpenWindows environment.

```
# . /.profile
```

The OpenWindows environment displays the message:

```
Starting OpenWindows in 5 seconds (type Control-C to interrupt)
```

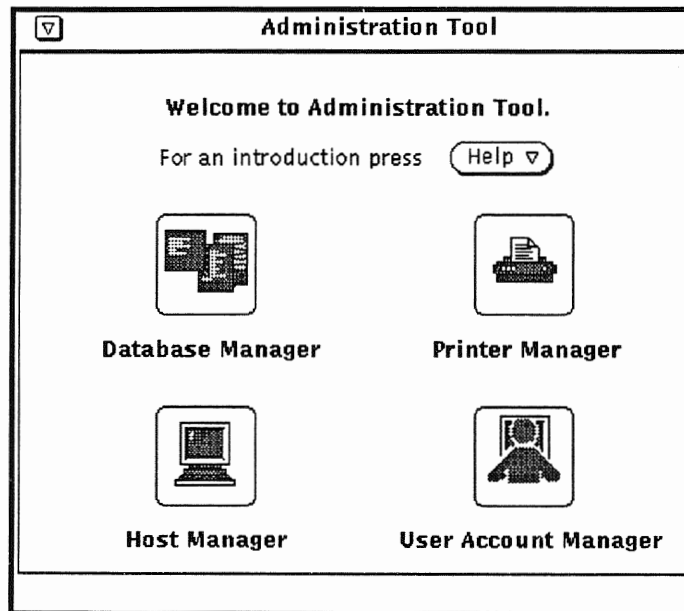


## Starting Administration Tool

5. Start Administration Tool in a Shell Tool or Command Tool window.

```
# admintool &
```

The Administration Tool window appears.

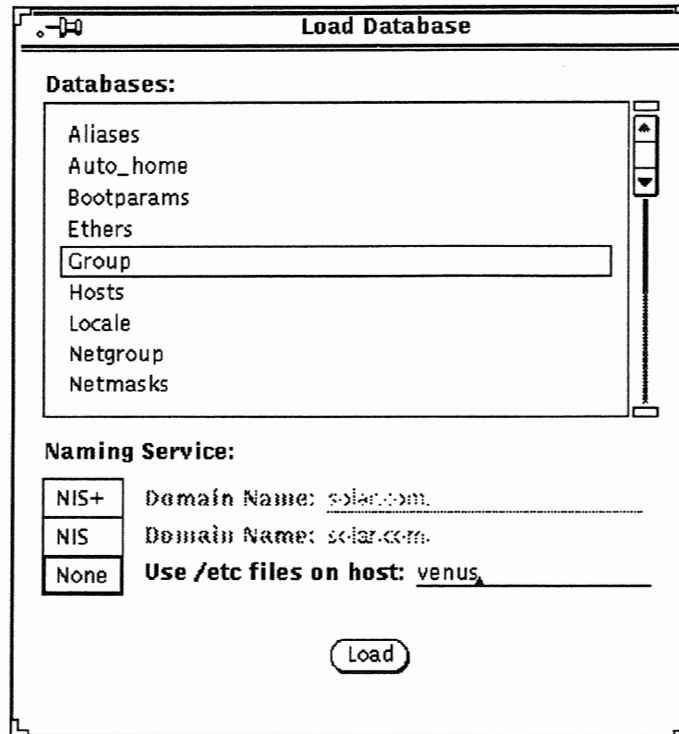


6. Click on the Database Manager icon to access the system database files.



## Loading the Group Database

The Load Database window appears.

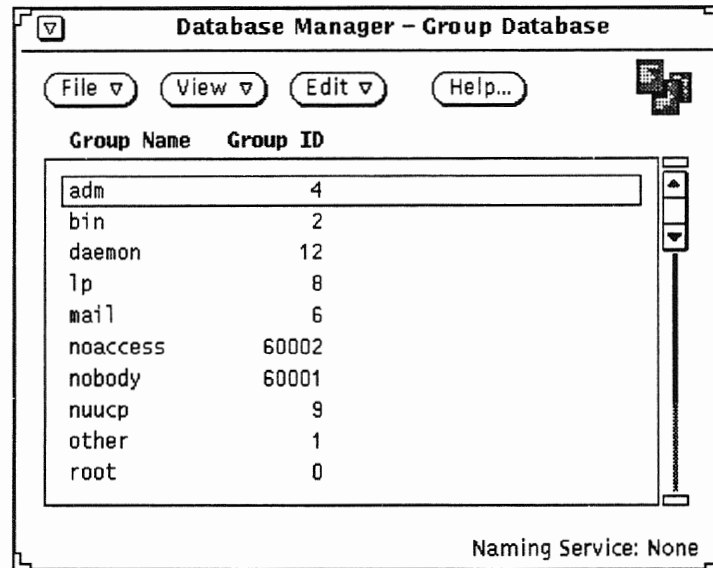


7. Select Group in the Load Database list.
8. Select None from the Naming Service options to use the local host's /etc files.
9. Click on the Load button.



## Accessing the Add Entry Form

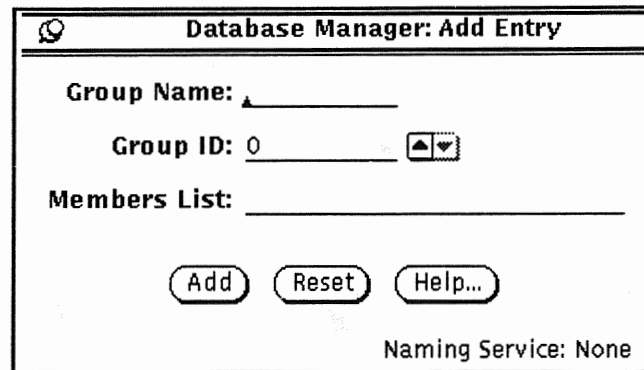
The Group Database window appears.



10. Choose Add Entry from the Edit menu. (Because Add Entry is the first selection on the Edit menu, you can just click SELECT on the Edit button as a shortcut).

## Creating a New Group

The Add Entry window is displayed.



Database Manager: Add Entry

Group Name: \_\_\_\_\_

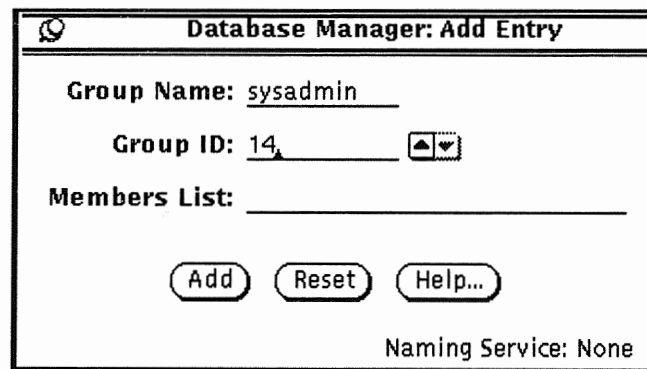
Group ID: 0 \_\_\_\_\_ ▲▼

Members List: \_\_\_\_\_

Add Reset Help...

Naming Service: None

11. Enter the following information:
  - a. Group name is `sysadmin`.
  - b. Group ID number (GID) is 14.



Database Manager: Add Entry

Group Name: sysadmin

Group ID: 14 \_\_\_\_\_ ▲▼

Members List: \_\_\_\_\_

Add Reset Help...

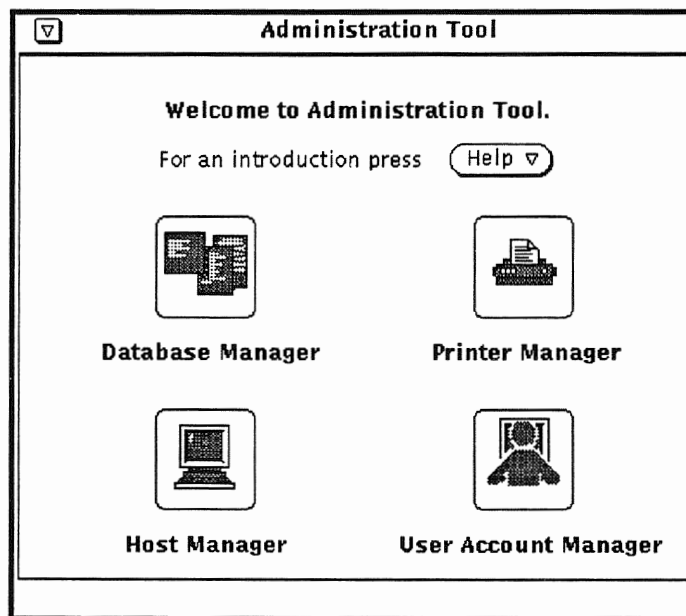
Naming Service: None

12. Click on Add.
13. Add another group called `students` with a GID of 110.
14. Dismiss the Add Entry window.
15. Quit the Group Database window.

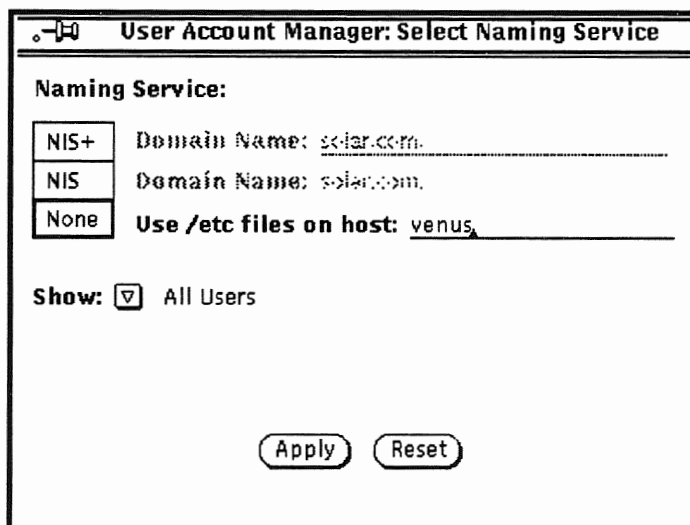


## Starting the User Account Manager

1. Click on the User Account Manager icon from the main Administration Tool window to create a new user.

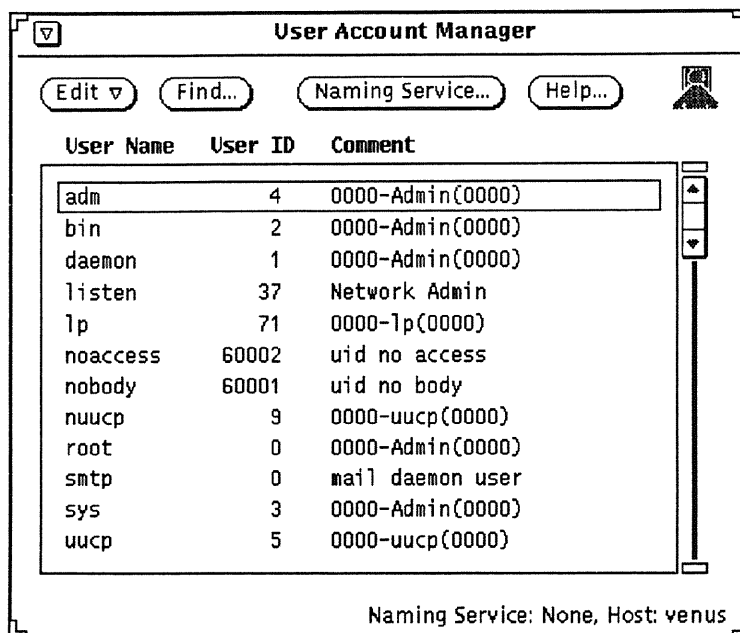


2. Set the name service to None to use the /etc files stored on the local system. Click on Apply.

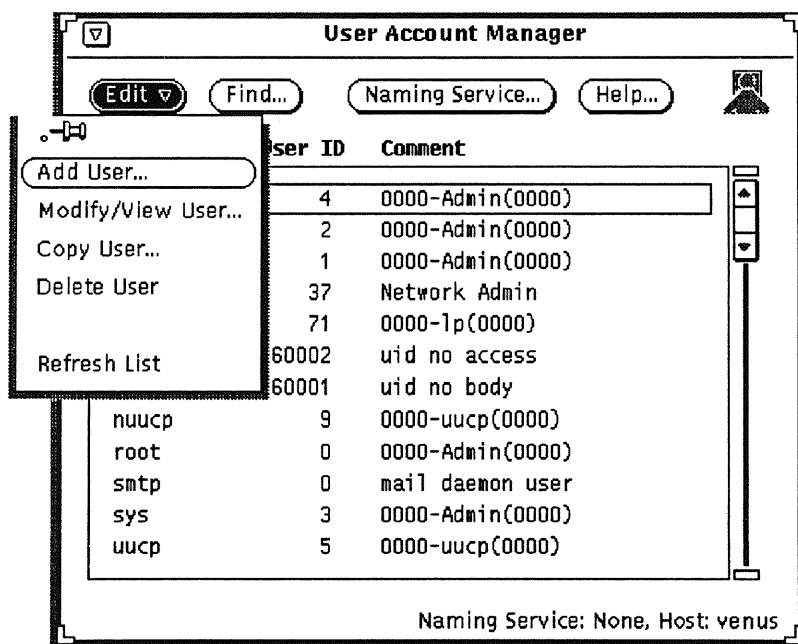


## Accessing the Add User Form

The User Account Manager Window is displayed.



3. Choose the Add User option from the Edit menu.





## Filling in the Add User Form

The Add User form is displayed.

**User Account Manager: Add User**

---

**USER IDENTITY**

User Name:

User ID:

Primary Group:

Secondary Groups:

Comment:

Login Shell:  Bourne /bin/sh

**ACCOUNT SECURITY**

Password:  Normal password...

Min Change:  days

Max Change:  days

Max Inactive:  days

Expiration Date:  None  None  None

Warning:  days

**HOME DIRECTORY**

Create Home Dir:  Yes if checked

Path:

Server:

Skeleton Path:

AutoHome Setup:  Yes if checked

Permissions: Read Write Execute

Owner:

Group:

World:

**MISCELLANEOUS**

Mail Server:

## Filling in the Add User Form

4. Specify the USER IDENTITY values for the variables listed.
  - a. Use your first name as the user name.
  - b. Use a UID of 100 plus the host number assigned by your instructor. For example, if your host number is 1, your UID would be 101.
  - c. Use 110 as the primary group
  - d. Specify 14 as the secondary group.
  - e. Use the user's full name as the comment.

### Account Security—Setting Passwords

There are four choices for specifying a user's password.

Password Status	Description
Cleared until first login	Account will not have a password and the user is prompted to supply a password upon first login (by default).
Account is locked	Account is locked and the user will not be able to login until the administrator assigns a password.
No password--setuid only	Account cannot be logged in to, but allows account programs, such as lp or uuCP, to run.
Normal password	Allows the administrator to apply a password to the account while adding the user.

5. Specify the ACCOUNT SECURITY values for the variables listed.
  - a. Use the submenu from the Password button to apply a normal user password.
  - b. Fill in the password information prompted for on the pop-up set password form.
  - c. Leave the other account security options blank. (They are covered in the next lesson.)



## Filling in the Add User Form

6. Specify HOME DIRECTORY values for the listed variables.
  - a. Click on the Create Home Dir check box to create the user's home directory.
  - b. Specify the Path variable as `/export/home/username`.
  - c. Enter the name of your system as the server.
    - a. Enter `/etc/skel` as the Skeleton Path (this is where the user's initialization files will be copied from).
    - b. Do *not* click the AutoHome Setup check box.
    - c. Locate the following matrix towards the bottom of the form.

Permissions	Read	Write	Execute
Owner:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Group:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
World:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

This matrix enables the administrator to set the user's home directory permission.

- d. Change the user's home directory permission to read, write, and execute by owner; read and execute by group; and no access for the world by clicking the appropriate check boxes.







## Verifying the New Account

10. Log out and log in as the user you just created.
11. Display the contents of your home directory.

```
$ ls -a
.          .profile      local.login
..         local.cshrc    local.profile
$
```

Administration Tool copied four default environment files from the skeleton directory you specified (/etc/skel).

The `.profile` file is very bare and was read by your Bourne shell when you logged in. The `local.profile` file is more complete and starts OpenWindows. The `local.cshrc` and `local.login` files are provided for C shell users.

12. If you want to use the OpenWindows environment automatically upon login enter the following:

```
$ mv local.profile .profile
```

## Using the New Account

From now on you should use this new account as your working account. Because the new account uses the `sysadmin` group, you can use Administration Tool directly from this account.

You can also temporarily switch user to the superuser, further reducing the need to log in directly as the superuser.



## Examining the /etc/passwd File

The password database is implemented as the /etc/passwd file for local user accounts.

1. Display the /etc/passwd file in a Command Tool window:

This file consists of a number of one-line entries. Colons are used to delimit the individual fields.



The password file is read-only to the world. Only the superuser can edit the file. Ordinary users can modify certain information, such as their password, or their login shell, using the `passwd` command.

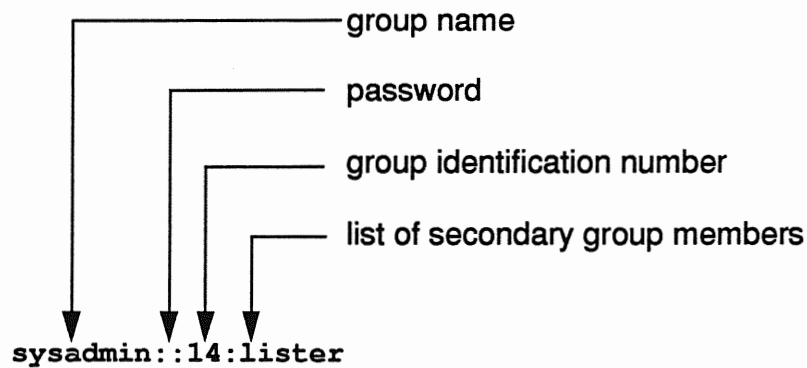
The `x` in the password field is a place holder. The actual password is stored in the /etc/shadow file along with other password related information. (The /etc/shadow file is examined in the next lesson when password aging values are set.)

## Examining the /etc/group File

The group database is implemented as the /etc/group file for local user accounts.

2. Display the /etc/group file in a Command Tool window:

This file also consists of a number of one-line entries and colons are used to delimit the individual fields.



When the list of users is greater than one, commas are used to separate user names.



## Creating User Accounts Manually

User accounts can be added manually if using Administration Tool is not feasible because you are using an ASCII terminal as the system console.

1. Use the `groupadd` command to create the user's group (if necessary).

```
# groupadd -g 100 explorer
```

2. Use the `useradd` command to add the user and create the user's home directory.

```
# useradd -u 115 -g 100 -c "Lt. Ripley" \  
-d /export/home/ripley -m -s /bin/sh riple
```

3. Issue the `passwd` command for the new user account to apply the user's password.

```
# passwd riple
```

4. Add any initialization files to the user's home directory.

## Using the `su` Command

The `su` (switch user) command is used to become a different user without logging out. Perform the following exercises in the same Command Tool window.

1. Display your current UID, user name, GID, and group name by typing:

```
$ id  
uid=113(ripley) gid=100(explorer)
```

2. Switch user to the superuser by typing:

```
$ su  
#
```

If no argument is supplied, `su` switches user to the superuser.

3. Display your current UID, user name, GID, and group name by typing:

```
# id  
uid=0(root) gid=1(other)
```

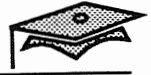
4. Display the processes associated with the current window by typing:

```
# ps  
PID TTY TIME CMD  
2697 pts/2 0:00 ps  
2695 pts/2 0:00 sh  
2689 pts/2 0:00 sh  
#
```

A `ps` process and two shell processes should be listed. The `su` command creates a new shell.

5. Exit the new shell by typing:

```
# exit
```



## Summary

In this lesson, you learned that:

- Administration Tool is a graphical administration interface that is used to manage system database files, printers, and user accounts.
- The User Account Manager of Administration Tool is used to create user accounts.
- Both the `/etc/passwd` and `/etc/group` files store user account information.
- The `su` command is used to switch to the superuser or another user without logging out.



# *User Account Maintenance*

---



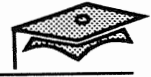
## Objectives

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

- List the system's four password requirements.
- Set up password aging on an existing user account using Administration Tool.
- Create a new user account using Administration Tool.
- Lock a user account using Administration Tool.
- Delete a user account using Administration Tool.
- Modify several system default files that allow the administrator to control and monitor superuser access to the system.

## References

*SunOS 5.1 Administering Security, Performance, and Accounting*,  
Chapter 2, "Securing System Access"



## Introduction

This lesson presents additional features for controlling how often users can and must change their passwords, and procedures for facilitating the creation and removal of user accounts from the system.

Using system default files to protect and monitor superuser access is also covered.

## Password Requirements

Passwords must meet these requirements:

- Have at least six characters (only the first eight are significant).
- Contain at least two alphabetic characters (uppercase or lowercase) and one numeric or special character.
- Differ from the login name.
- Differ from the previous password by at least three characters.

Superuser and privileged users are not forced to comply with these requirements or with password aging requirements.



## Password Aging Features

Recall from the previous lesson that password aging parameters are included in the Account Security section of the Add User form of the User Account Manager.

Passwords that are not changed or that remain active beyond their intended termination time are a security risk. The Solaris 2.x computing environment provides several parameters for controlling passwords that can be set on a per user basis using Administration Tool.

The following table describes the different password aging parameters.

Parameter	Meaning
Min Change	The minimum number of days required between password changes.
Max Change	The maximum number of days the password is valid.
Max Inactive	The number of days of inactivity allowed for that user.
Expiration Date	An absolute date specifying when the login may no longer be used.
Warning	The number of days before the password expires that the user is warned.

Users receive the following message if they attempt to change their password before the Min Change parameter:

"Sorry, less than x days since last change"

If users exceed the Max Change parameter they will see the following message:

"Your password has expired. Choose a new one."

## Password Aging Features

### Example:

The following example modifies an existing user account to apply password aging parameters: minimum number of days between password changes is one, maximum number of days the password is valid is 60, and an account expiration date.

**User Account Manager: Modify User**

---

**USER IDENTITY**

User Name: lister

User ID: 112

Primary Group: 110

Secondary Groups: sysadmin

Comment: Dave Lister

Login Shell:  Bourne /bin/sh

**ACCOUNT SECURITY**

Password:  Normal password...

Min Change: 1 days

Max Change: 60 days

Max Inactive:      days

Expiration Date:  None  None  None

Warning:      days

**HOME DIRECTORY**

Path: /export/home/lister

Server:                     

AutoHome Setup:  Yes if checked

Permissions	Read	Write	Execute
Owner:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Group:	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
World:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

**MISCELLANEOUS**

Mail Server:



## Exercise 3-1

The following extended lab procedure is intended to provide experience with the following tasks:

- Use Administration Tool to modify a user account to include password control.
- Examine the format of the `/etc/shadow` file where password control information is stored.
- Examine and modify several system default files that allow the administrator to control and monitor superuser access to the system.

## Modifying an Existing User Account

1. Start Administration Tool by typing the following command in a Command Tool window:

```
# admintool &
```

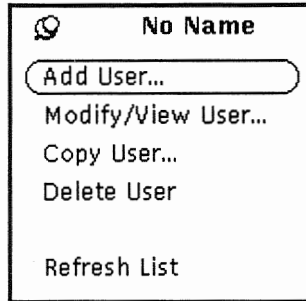
2. Start the User Account Manager by clicking on its icon in the Administration Tool window.
3. Select None for the name service option because a local user account is modified.
4. Click on Apply.

A User Account Manager window listing all of the currently defined user accounts on the system is displayed.



## Examining the User Account Manager Edit Choices

5. Press MENU on the Edit button to display the Edit menu.



The available choices are:

Add User	Displays a form that is used to specify and create a new user account.
Modify/View User	Displays the same form with the values for the currently selected user already filled in. This form is used to modify or view an existing account.
Copy User	Displays the same form with the values for the currently selected user already filled in. This form is used to create a new account based on an existing account.
Delete User	Displays a form that is used to specify which components of an account to delete.
Refresh List	Updates the list of users to reflect any changes that may have occurred since the utility was started.

You have already used the Add User item. In this procedure you will use the Modify/View User, Copy User, and Delete User items (you can try the Refresh List selection at any time).



## Selecting a User Account to View or Modify

6. Select the login name of the user account you created as part of the previous lesson in the User Account Manager window.
7. Choose Modify/View User from the Edit menu.

The form is displayed with the current values filled in.

## Adding Account Security

8. Ensure that the user must wait seven days between password changes by entering 7 in the Min Change field. This value is typically used to prevent users from changing their password back immediately after a forced change.
9. Specify that the user must change his password at least once a month by entering 30 in the Max Change field.
10. Set up the account to automatically disable itself if the user does not log in for over 30 days by entering 30 in the Max Inactive field.
11. Use the Expiration Date submenus to set an account expiration date. The left-most submenu is used to set the day of the month, the middle to set the month, and the right to set the year.
12. Set up the account to warn the user five days before the user's account will expire by entering 5 in the Warning field.
13. Apply the changes you have made by clicking on Apply. After the changes are made the panel is removed from the screen.

These account security features can also be controlled by the superuser with options to the `passwd` command. See the `passwd(1)` manual page for more information.



## Examining the /etc/shadow File

In the preceding lesson you learned that an x in the password field of the passwd database is a place holder for the password that is stored in the /etc/shadow file, which also stores other password-related information.

The account security information you just specified (Password, Min Change, Max Change, Max Inactive, Expiration Date, and Warning) is all stored in the /etc/shadow file which is only readable by the superuser.

Become superuser and display the /etc/shadow file. A file similar to the following is displayed.

```
# cat /etc/shadow
root:..6HR9T6ZHueuU:8911:~::~:
daemon:NP:6445:~::~:
bin:NP:6445:~::~:
sys:NP:6445:~::~:
adm:NP:6445:~::~:
lp:NP:6445:~::~:
smtp:NP:6445:~::~:
uucp:NP:6445:~::~:
nuucp:NP:6445:~::~:
listen:*LK*:~::~:
nobody:NP:6445:~::~:
noaccess:NP:6445:~::~:
lister:5XpyNVSZTYWBE:8912:1:60::~8738:
```

*NP Password*

*\*LK\* Password in locked*

*→ 6445 the # of days since 11/170 which is the birthday of unix*

This file consists of a number of one-line entries. Colons are used to delimit the individual fields.

## Examining the `/etc/shadow` File

This file contains the following eight fields.

```
user-name:password:lastchg:min:max:warn:inactive:expire
```

The fields are:

<i>user-name</i>	This field contains the user's login name.
<i>password</i>	This field may contain the following entries: a 13-character encrypted user password; the string <code>*LK*</code> , which indicates an inaccessible account; or the string <code>NP</code> , which indicates no password for the account.  The <code>passwd</code> command creates an encrypted password 13 characters in length chosen from a 64-character alphabet ( <code>, /, 0-9, A-Z, a-z</code> ).
<i>lastchg</i>	This field indicates the number of days between January 1, 1970 and the last password modification date.
<i>min</i>	This field contains the minimum number of days required between password changes.
<i>max</i>	This field contains the maximum number of days the password is valid before the user is prompted to specify a new password.
<i>warn</i>	This field contains the number of days before the password expires that the user is warned.
<i>inactive</i>	This field contains the number of inactivity days allowed for that user before the user's account is locked.
<i>expire</i>	This field contains the absolute date when the user account expires. Once exceeded, the user can no longer log in to the system.



## Using the Copy User Option

1. Choose Add User from the Edit menu and create a user with the following specifications:
  - a. User name is student.
  - b. UID is 999.
  - c. GID is 110 (students).
  - d. Comment is: Network-Wide User
  - e. Set a normal password using cangetin as the password in the Set Password window.
  - f. Use /export/home/student as the home directory path.
  - g. Use your system name as the home directory server.
  - h. Click on Add to create the user account.
2. Dismiss the Add User Form.
3. The last user created (student) should still be selected from the last Add User session. If this user is not selected, select it now from the User Account Manager display window.
4. Choose Copy User from the Edit menu. This form is used to create a new account based on an existing account.
5. Change the following three values:
  - a. User name is student2.
  - b. UID is 9999.
  - c. Use /export/home/student2 as the home directory path.
6. Click on Add.

A message is displayed in the lower left corner of the form indicating that the user is being added.
7. Dismiss the Copy User form.

## Locking an Account

When a user leaves a site or no longer requires access to the system for some other reason, the administrator will, in most cases, want to make that user's account inaccessible. Rather than deleting the account, the administrator should consider locking the account. This makes the account inaccessible without destroying potentially important shared files.

1. Click on the User Account Manager icon, if necessary.
2. Select the last user added: student2.
3. Choose Modify User from the Edit Menu.
4. Choose Account is Locked from the Password menu in order to lock the account.
5. Click SELECT on the Apply button.

After the changes are made the panel is removed from the screen.

6. Verify the account is locked by displaying the user account in the `/etc/shadow` file.

```
# grep student2 /etc/shadow
```

```
student2:*LK*:8414:0:::::
```

*equiv to \**  
The user's password is set to `*LK*` which is an unmatchable password that locks the account and indicates to the administrator that the account is locked.

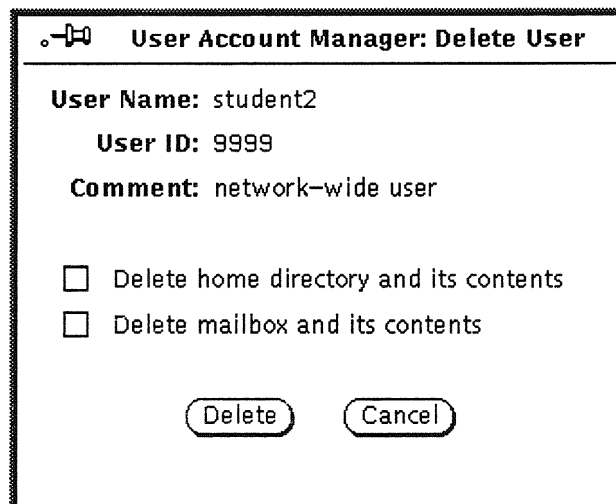
This account security function can also be controlled by the superuser with the `passwd -l` command. See the `passwd(1)` manual page for more information.



## Deleting an Account

After archiving or otherwise accounting for the user's files, the administrator may decide to delete the user account.

7. Display the User Account Manager window.
8. The user named `student2` should still be selected from the last time you used this window. If this user is not selected, select it now.
9. Choose Delete User from the Edit Menu.



10. Deleting a user account without checking the check boxes only deletes references to the user from the user databases. Delete all references to the user, as well as the contents of the user's home directory and mailbox, by clicking on both check boxes and then clicking on Delete.

Any files owned by the user but not located within the user's home directory hierarchy remain on the system unless they are located (using the `find` command) and removed.

---

## Deleting an Account

11. Dismiss the User Account Manager window.
12. Close Administration Tool to an icon.



## The `passwd` Command

The superuser can maintain passwords with the `passwd` command.

### Command format:

```
passwd [ -l | -d ] [ -f ] [ -n min ] [ -x max ] name
```

```
passwd -s [ name | -a ]
```

### Options:

- l           Locks the password entry for user name.
- d           Deletes the password for user name.
- f           Force the named user to change his password at the next login.
- n *min* -x *max*  
              Set the minimum and maximum field for user name.
- s           Show password attributes for user name.
- a           Show password attributes for all entries.



## Exploring the /etc/default Directory

Several ASCII files containing variables that specify system defaults are located in the /etc/default directory.

```
# cd /etc/default
# ls
cron    fs      init    login   passwd  su      tar
```

The login, passwd, and su files relate to system security.

### Enable su Logging

1. Display the contents of the /etc/default/su file.

```
#ident    "@(#)su.df11.5    92/07/14 SMI"/* SVr4.0 1.2*/
SULOG=/var/adm/sulog
#CONSOLE=/dev/console
```

All of the system default files contain variables, one per line, set equal to default values. Lines that begin with # are comment lines and render any variable undefined.

The value of the SULOG variable specifies the name of the file where all su (switch user) attempts to switch to another user are logged. If undefined, su logging is turned off.

If the value of the CONSOLE variable is defined as /dev/console, all successful attempts to su to the superuser are logged on the console.

2. Uncomment the CONSOLE variable. All of the default files are read-only, so if you use vi to make the changes, you'll need to use :wq! to write the changes and exit vi. Now use su to become superuser. Notice that the attempt is logged in the Console window.



## The /etc/default/passwd File

3. Display the contents of the /etc/default/passwd file.

```
# cat passwd
#ident "@(#)passwd.dfl 1.3 92/07/14 SMI"
MAXWEEKS=
MINWEEKS=
PASSELENGTH=6
```

*mean days not weeks*

This file defines three important variables.

The value of the `MAXWEEKS` variable specifies the maximum number of weeks (7 days) a password is valid before it must be changed for all normal users. If defined as null, only users who have a value for `MAX` specified in the `/etc/shadow` file must change their passwords at the specified times.

However, a `MAX` value in the `/etc/shadow` file is measured in days.

The value of the `MINWEEKS` variable specifies the minimum number of days between password changes for all normal users. If defined as null, only users who have a value for `MIN` specified in the `/etc/shadow` file are limited as to when they may change their passwords.

However a `MIN` value in the `/etc/shadow` file is measured in days.

The purpose of the `PASSELENGTH` variable is to specify a minimum password length for all normal users. Currently, changing the `PASSELENGTH` variable has no effect. The `passwd` command requires a password length greater than 5 characters.

## The /etc/default/login File

4. Display the contents of the /etc/default/login file.

```
#ident "@(#)login.dfl1.5 92/07/14 SMI" /* SVr4.0 1.1.1.1 */
#TIMEZONE=EST5EDT
HZ=100
#ULIMIT=4096
CONSOLE=/dev/console
PASSREQ=YES
ALTSHELL=YES
```

*→ disables remote root login  
→ no device only allow su to root  
from console*

This file defines two important security parameters.

The ALTSHELL variable is used to set the SHELL environment variable under certain conditions.

- If the shell field in the /etc/passwd file contains a value, and the ALTSHELL variable is set to YES (which is the default), then the value of SHELL is set to the value in the password file.
- If the shell field in /etc/passwd contains a value and the ALTSHELL variable is commented out or is set to NO, then the value of SHELL is not set as an environment variable. In this case, the user has a shell to work in, it is just that the SHELL environmental variable is not set.

If the value of the PASSREQ variable is set to YES, (which is the default) users with null passwords are required to choose a password the next time they log into the system. Otherwise, null passwords are permitted.



## The `/etc/default/login` File

The `CONSOLE` variable can be used to specify three conditions for superuser logins.

- If the variable is defined as `/dev/console`, (which is the default) logging in as the superuser is only permitted from the console.
- If the variable is not defined, logging in as the superuser is permitted from anywhere (e.g. over the network, through an attached modem, or through an attached terminal).
- If the variable is defined as `null` (`CONSOLE=`), logging in as the superuser is not permitted. In this case the only way to gain superuser privileges is to log in as a normal user and become superuser with the `su` command.

5. Attempt to use the `rlogin` command as root on your system:

```
$ rlogin system_name -l root
Not on system console
Connection closed.
```

Edit the `/etc/default/login` file and place a comment symbol in front of the `CONSOLE` variable.

Attempt the remote root login again. You should be successful.

## Using Restricted Shells

The Solaris 2.x environment provides restricted versions of the Korn shell (`rksh`) and the Bourne shell (`rsh`) to allow administrators better control over a user's execution environment. They are particularly useful for providing temporary system access with restricted permissions on login sessions.

The actions of `rsh` and `rksh` are identical to those of `sh` and `ksh` with restrictions. Users are prevented from:

- Changing directory
- Setting the value of `$PATH`
- Using absolute command path names
- Redirecting output (`>` and `>>`)

It is possible to provide the user with standard shell features, while restricting access to all commands. This means administrators must also create a limited set of commands (such as `/usr/rbin`) for a restricted user. The next step is restrict permissions in the user's home directory so that the user cannot change this restricted environment.

The result of these restrictions is that the writer of the `.profile` script has complete control over user actions by providing a limited set of commands and limiting the user to a specified directory.

Make sure not to confuse the restricted shell with the remote shell.

- `/usr/lib/rsh` = restricted shell
- `/usr/bin/rsh` = remote shell



## Summary

In this lesson, you learned that:

- The User Account Manager can be used to modify a user account to include extra account security (password aging).
- The `/etc/shadow` file contains encrypted passwords and password aging information.
- The User Account Manager can be used to create a new user account based on an existing account, lock an account, or delete a user account.
- Important system-wide security variables are specified in ASCII files in the `/etc/default` directory.

# *Answer Key*

---





## Lesson 1: Basic Security Features

### Exercise 1-1

1. A `passwd` record and a `shadow` record, a home directory, and initialization files.
2. The format of a `passwd` record is:

*user-name:x:uid:gid:comment:home-dir:login-shell*

The fields are:

*user-name* is the login name

*x* is a placeholder for the encrypted password

*uid* is a number used to identify the user

*gid* is a number used to identify the user's primary group

*comment* is usually the user's full name

*home-dir* is the absolute path name of the user's home directory

*login-shell* defines the user's default login shell which may be one of `/bin/sh`, `/bin/csh`, or `/bin/ksh`

3. 0-99 for system accounts, 60001 for `nobody`, and 60002 for `noaccess`.
4. `/etc/skel`
5. The `group` database specifies the primary and secondary groups to which each user belongs.
6. 100-60000
7. Superuser account and `sysadmin` group
8. `umask`



## Lesson 1: Basic Security Features

### Exercise 1-1 (continued)

9. A program that has the `setuid` bit set allows anyone with execute access to run the program as if they were the program's owner.

A program that has the `setgid` bit set allows anyone with execute access to run the program as if they were a member of the program's group.

A directory with the `setgid` bit set propagates the GID to any files created below it. This is useful for shared project directories.

10. `chmod 4755 my_file`
11. `chmod 2770 our_project`
12. Setting the sticky bit on a directory that has write access prevents users from deleting other users' files in public directories. That is, files within that directory can only be removed or renamed if one or more of the following is true:
- The user owns the file
  - The user owns the directory
  - The file is writable by the user
  - The user is the superuser
13. The system compares the process' system data to the information stored in the file's inode. First the UID is compared and then the GID. If neither of these match, the other permissions are used.



## Lesson 2: Creating User Accounts

Follow the steps as described.

---

## Lesson 3: User Account Maintenance

Follow the steps as described.



## Appendix A: ASET

### Exercise A-1

1. su
2. Follow the steps as described.
3. /usr/aset/aset
4. /usr/aset/reports/latest/taskstatus
5. Use cat, more, or a text editor to view the file.

# *Using the Automated Security Enhancement Tool (ASET)*

---



## **References**

*SunOS 5.1 Administering Security, Performance, and Accounting,*  
Chapter 5, "Monitoring and Controlling Security Using ASET"



## Introduction

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

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

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

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

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

- ✓ ***This appendix is included for students who are interested in learning about ASET.***
- ✓ ***Use your own discretion in using valuable class time to cover this material.***

## ASET Security Options

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

- Low

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

- Medium

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

- High

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



## ASET Tasks

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

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

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

- ✓ **The task scripts in the `/usr/aset/task` directory use the `/usr/aset/masters` scripts based on the specified security level. Even though the task scripts identify what is being evaluated, the `tune` scripts identify the specific file systems changes that will be made.**



# ASET Terminology

## Firewall

The `firewall` task script checks to see if a machine can be used safely as a network *router*. A router is any machine that has two Ethernet (network) interfaces whose purpose is to pass information from one network to another.

A firewall machine is a router that does not forward data or advertise routes from one network to the other. This means users must login into the firewall system to gain access to the other network. (Router configuration is beyond the scope of this course.)

The `firewall` task script in the `/usr/aset/tasks` directory does two things:

- Turns off IP forwarding to ensure that the `firewall` system does not send information from one network to another.
- Makes sure the `in.routed` daemon is started with the `-q` flag to prevent broadcasting visible routing information.

## Eliminating the Firewall Task

The `firewall` task script is run at every level, but only takes action at the highest security level.

If your system does not require firewall protection and you want to run at the highest security level, you can eliminate this task by editing the `asetenv` file in the `/usr/aset` directory to remove the `firewall` task from the `TASKS` list.



## The `aset` Command

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

The system runs considerably slower for approximately 15 minutes while the `aset` processes are running in the background.

### Example:

1. Become superuser.
2. Execute the `/usr/aset/aset` command.

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

ASET running at security level low

Machine = venus; Current time = 0527_15:11

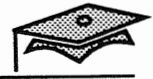
aset: Using /usr/aset as working directory

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

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

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

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



## Checking on Task Status

### The taskstat Command

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

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

```
The following tasks are done:
  firewall
  env
  sysconf
  usrgrp
```

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

### The taskstatus File

Or, display the `/usr/aset/reports/latest/taskstatus` file to verify that all tasks are done.

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

## ASET Report Files

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

```
cklist.rpt
eeprom.rpt
env.rpt
firewall.rpt
sysconf.rpt
tune.rpt
usrgrp.rpt
```

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

---

**Note:** In order for the Solaris 2.x `/usr/aset/tasks/firewall` script to work correctly, change the occurrences of `/unix` to `/kernel/unix` on lines 33 and 35.

---



# ASET Report Files

## Example:

```
# more *.rpt
cklist.rpt
::::::::::::
*** Begin Checklist Task ***
No checklist master - comparison not performed.
... Checklist master is being created now. Wait ...
... Checklist master created.
*** End Checklist Task ***
::::::::::::
eeprom.rpt
::::::::::::
*** Begin EEPROM Check ***
::::::::::::
env.rpt
::::::::::::
*** Begin Environment Check ***
Warning! umask set to umask 022 in /etc/profile - not recommended.
*** End Environment Check ***
::::::::::::
firewall.rpt
::::::::::::
*** Begin Firewall Task ***
Could not find unix!
::::::::::::
sysconf.rpt
::::::::::::
*** Begin System Scripts Check ***
Warning! The use of /.rhosts file is not recommended for system security.
*** End System Scripts Check ***
::::::::::::
tune.rpt
::::::::::::
*** Begin Tune Task ***
... setting attributes on the system objects defined in /usr/aset/masters/tune.low
*** End Tune Task ***
::::::::::::
usrgrp.rpt
::::::::::::
*** Begin User And Group Checking ***
Checking /etc/passwd ...
Warning! Password file, line 20, invalid login directory:
        newuser:x:9004:1:::/bin/sh
Checking /etc/shadow ...
Warning! Shadow file, line 19, no password:
        lister::8391:0::::
Warning! Shadow file, line 23, no password:
        hollie::8414:0::::
... end user check.
Checking /etc/group ...
... end group check.
*** End User And Group Checking ***
```

---

## Changing Security Levels

The aset command below sets system security to the high level.

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

ASET running at security level high

Machine = venus; Current time = 0114_14:45

aset: Using /usr/aset as working directory

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

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

Run /usr/aset/util/taskstat to check their status:

```
/usr/aset/util/taskstat [aset_dir]
```

where aset\_dir is ASET's operating directory, currently=/usr/aset.

When the tasks complete, the reports can be found in:

```
/usr/aset/reports/latest/*.rpt
```

You can view them by:

```
more /usr/aset/reports/latest/*.rpt
```

```
#
```

---

**Note:** Running the aset utility at this level will greatly limit your ability to perform subsequent lab exercises.

---



## Restoring Pre-ASET System Files

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

### Command format:

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

### Options:

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

### Example:

```
# /usr/aset/aset.restore  
aset.restore: beginning restoration ...  
Executing /usr/aset/tasks/firewall.restore  
Beginning firewall.restore...  
.  
.  
.
```



## Exercise A-1

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

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

1. Become superuser.
2. Edit the `passwd` file manually to add duplicate users and users without passwords. Issue the `pwconv` command to update the `/etc/shadow` file.
3. Then run the Automated Security Enhancement Tool, at low security, to identify any user and/or group security risks.

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

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

---

✓ *The `pwconv` has not been covered; it is used to update the `/etc/shadow` file with information in the `/etc/passwd` file.*

