EXAM PREP

Your Complete Certification Solution

Exam CX-310-202

Solaris 10 System Administration

Part II

Bill Calkins

Solaris 10 System Administration Exam Prep (Exam CX-310-202), Part II

ISBN-13: 978-0-7897-3817-2

ISBN-10: 0-7897-3817-1

Library of Congress Cataloging-in-Publication Data:

Calkins, Bill.

Solaris 10 system administration exam prep (Exam CX-310-200) / Bill Calkins.

p. cm.

ISBN 978-0-7897-3790-8 (pbk. w/cd)

1. Electronic data processing personnel--Certification. 2. Operating systems (Computers)--Examinations--Study guides. 3. Solaris (Computer file) I. Title.

QA76.3.C34346 2008

005.4'32--dc22

2008031592

Printed in the United States of America

First Printing: May 2009

Bulk Sales
Que Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact:

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact:

International Sales
+1-317-581-3793
international@pearsontechgroup.com

Associate Publisher
David Dusthimer

Acquisitions Editor
Betsy Brown

Senior Development Editor
Christopher Cleveland

Technical Editor
John Philcox

Managing Editor
Patrick Kanouse

Project Editor
Jennifer Gallant

Copy Editor
Gayle Johnson

Indexer
Lisa Stumpf

Proofreader
Arle Writing and Editing

Publishing Coordinator
Vanessa Evans

Book Designer
Gary Adair

Page Layout
Mark Shirar

Introduction

Bill Calkins has been training Solaris system administrators for more than 15 years. This book contains the training material that he uses in his basic and advanced Solaris administration courses that, over the years, have helped thousands of Solaris administrators become certified. This is our second edition of the *Solaris 10 System Administration Exam Prep*. It covers updates that Sun has made to the Solaris 10 operating environment as of the October 2008 release. It began with the *Training Guide* for Solaris 2.6, 7, 8, and 9 and is now the *Exam Prep* for Solaris 10. Instructors from universities and training organizations around the world have used the book as courseware in their Solaris administration courses. In addition, administrators from around the world have used this book for self-study when instruction from a Sun training center is either unavailable or not within their budget. Many of you have written with your success stories, suggestions, and comments. Your suggestions are what keep making this guide more valuable.

The *Solaris 10 System Administration Exam Prep* books, Parts I and II, provide training materials for anyone interested in becoming a Sun Certified System Administrator (SCSA) for Solaris 10. When used as a study guide, these two books will save you a great deal of time and effort searching for information you will need to know when taking the exam. Each book covers the exam objectives in enough detail for inexperienced administrators to learn the objectives and apply the knowledge to real-life scenarios. Experienced readers will find the material in these books complete and concise, making it a valuable study guide for the Sun Certified System Administrator exams.

This book is not a cheat sheet or cram session for the exam; it is a training manual. In other words, it does not merely give answers to the questions you will be asked on the exam. We have made certain that this book addresses the exam objectives in detail, from start to finish. If you are unsure about the objectives on the exams, this book teaches you what you need to know. After reading each chapter, assess your knowledge of the material covered using the review questions at the end of the chapter. When you have completed reading a section, use the practice exam at the end of the book and the ExamGear test engine on the CD-ROM to assess your knowledge of the objectives covered on each exam. This CD-ROM contains sample questions similar to what you are likely to see on the real exams. More sample questions are available at http://www.UnixEd.com, so make sure you visit this site to find additional training and study materials.

How This Book Helps You

This book teaches you advanced topics in administering the Solaris 10 operating system. It offers you a self-guided training course of all the areas covered on the CX-310-202 certification exam by installing, configuring, and administering the Solaris 10 operating environment. You will learn the specific skills that are required to administer a system and, specifically, to pass the second part of the Sun Certified System Administrator exam for Solaris 10 (CX-310-202). If you are an experienced administrator who is upgrading an existing Solaris certification, you'll find in-depth coverage of the new topics you need to learn for the CX-310-203 upgrade exam in both the SCSA Solaris 10 OS CX-310-200 and CX-310-202 *Exam Prep* books.

Throughout the book, we provide helpful tips and real-world examples that we have encountered as system administrators. In addition, we provide useful, real-world exercises to help you practice the material you have learned. This book is set up as follows:

- **Organization**: This book is organized according to individual exam objectives. Every objective you need to know to install, configure, and administer a Solaris 10 system is in this book. We have attempted to present the objectives in an order that is as close as possible to that listed by Sun. However, we have not hesitated to reorganize them as needed to make the material as easy as possible for you to learn. We have also attempted to make the information accessible in the following ways:
  - This book includes the full list of exam topics and objectives.
  - Read the "Study and Exam Prep Tips" element early on to help develop study strategies. This element provides you with valuable exam-day tips and information on exam/question formats such as adaptive tests and case study-based questions.
  - Each chapter begins with a list of the objectives to be covered, exactly as they are defined by Sun. Throughout each section, material that is directly related to the exam objectives is identified.
  - Each chapter also begins with an outline that provides you with an overview of the material and the page numbers where particular topics can be found.
- **Instructional features**: This book is designed to provide you with multiple ways to learn and reinforce the exam material. The following are some of the helpful methods:
  - **Objective explanations**: As mentioned, each chapter begins with a list of the objectives covered in the chapter.
  - **Study strategies**: The beginning of each chapter also includes strategies for studying and retaining the material in the chapter, particularly as it is addressed on the exam.
  - **Exam Alerts**: Throughout each chapter you'll find exam tips that will help you prepare for exam day. These tips were written by those who have already taken the Solaris 10 certification exams.

- **Key Terms**: A list of key terms appears near the end of each chapter.
- **Notes**: These contain various types of useful information, such as tips on technology or administrative practices, historical background on terms and technologies, or side commentary on industry issues.
- **Cautions**: When you use sophisticated information technology, mistakes or even catastrophes are always possible because of improper application of the technology. Cautions alert you to such potential problems.
- **Step By Steps**: These are hands-on lab exercises that walk you through a particular task or function relevant to the exam objectives.
- **Exercises**: Found near the end of the chapters, exercises are performance-based opportunities for you to learn and assess your knowledge.
- **Suggested Reading and Resources**: At the end of each chapter is a list of additional resources that you can use if you are interested in going beyond the objectives and learning more about the topics presented in the chapter.

- **Extensive practice test options**: The book provides numerous opportunities for you to assess your knowledge and practice for the exam. The practice options include the following:
  - **Exam questions**: Each chapter ends with questions. They allow you to quickly assess your comprehension of what you just read in the chapter. Answers to the questions are provided in a separate element titled "Answers to Exam Questions."
  - **Practice exam**: A practice exam is included in Part II, "Final Review," for each exam (as discussed in a moment).
  - **ExamGear**: The ExamGear software included on the CD-ROM provides further practice questions.

NOTE

ExamGear software For a complete description of the ExamGear test engine, see Appendix A, "What's on the CD-ROM."

- **Final Review**: This part of the book provides you with three valuable tools for preparing for the exam:
  - **Fast Facts**: This condensed version of the information contained in the book will prove extremely useful for last-minute review.
  - **Practice Exam**: A full practice exam is included, with questions written in styles similar to those used on the actual exam. Use the practice exam to assess your readiness for the real exam.

  - **Answers to Practice Exam**: This element provides the answers to the full practice exam, with detailed explanations. These should help you assess your strengths and weaknesses.

- **Appendixes**: The book contains valuable appendixes as well, including a glossary and a description of what is on the CD-ROM (Appendix A).

These and all the other book features mentioned previously will enable you to thoroughly prepare for the exam.

Conventions Used in This Book

- **Commands**: In the steps and examples, the commands you type are displayed in a special `monospace` font.

- **Arguments, options, and <cr>**: In command syntax, command options and arguments are enclosed in < >. The words within the < > stand for what you will actually type. You don't type the < >. The <cr> that follows the command means to press Enter. You don't type the <cr>.

  ```
  lp -d<printer name> <filename> <cr>
  ```

- **Using the mouse**: When using menus and windows, you select items with the mouse. Here is the default mapping for a three-button mouse:

  Left button: Select

  Middle button: Transfer/adjust

  Right button: Menu

  You use the Select button to select objects and activate controls. The middle mouse button is configured for either Transfer or Adjust. By default, it is set up for Transfer, which means that you use this button to drag or drop list or text items. You use the left mouse button to highlight text, and then you use the middle button to move the text to another window or to reissue a command. The middle button can also be used to move windows around on the screen. You use the right mouse button, the Menu button, to display and choose options from pop-up menus.

- **Menu options**: The names of menus and the options that appear on them are separated by a comma. For example, "Select File, Open" means to pull down the File menu and choose the Open option.

- **Code continuation character**: When a line of code is too long to fit on one line of the book, it is broken and continued to the next line. The continuation is preceded by a backslash.

Audience

This book is the second book in a series designed for anyone who has a basic understanding of UNIX and wants to learn more about Solaris system administration. Whether or not you plan to become certified, the *Solaris 10 System Administration Exam Prep* books, Part I and Part II, are the starting point to becoming a Solaris System Administrator. It's the same training material that the author uses in his Solaris 10 Intermediate and Advanced System Administration courses. This book covers advanced system administration topics you need to know before you begin administering the Solaris operating system. Our goal is to present the material in an easy-to-follow format, with text that is easy to read and understand. The only prerequisite is that you have read my *Solaris 10 System Administration Exam Prep Part I* book.

This book is intended for experienced system administrators who want to become certified, update their current Solaris certification, or simply learn about the features of the Solaris 10 operating environment. To pass the CX-310-202 and CX-310-203 certification exams, you need a solid understanding of the fundamentals of administering Solaris 10. This book helps you review the fundamentals required to pass the certification exam.

The Sun Certified System Administrator Exams

To become a Sun Certified System Administrator, you need to pass two exams: CX-310-200 (Part I) and CX-310-202 (Part II). This book covers the material on the Part II exam. You must pass the CX-310-200 exam before taking the CX-310-202 exam. You will not receive a certificate until you have passed both examinations. Also, if you are already certified in Solaris 2.6, 7, 8, or 9, you need to know the material covered in this book as well as in *Solaris 10 System Administration Exam Prep: CX-310-200 Part I* to take the upgrade exam, CX-310-203, to become certified on Solaris 10.

Beware of fakes. We have seen some websites promoting their own certification programs, so be sure to evaluate them carefully. Certification programs promoted by these sites are not the same as the Sun certification program. You will not receive a certificate from Sun until you pass Sun's exams from a certified Sun testing center. Go to my website (www.UnixEd.com) for links to the real exams and information on Sun's certification program if you are in doubt. In addition, feel free to visit our online Solaris certification discussion forum at www.UnixEd.com, where you can ask me questions directly.

Summary

It's not uncommon for Sun to change the exam objectives or to shift them around after the exams have been published. We highly recommend that before you begin reading this book, you visit my website at www.UnixEd.com to get the most up-to-date list of exam objectives, the errata for this book, up-to-date sample exam questions, and any other last-minute notes about these exams. We will provide all the information you need to pass the exam—all you need to do is devote the time. Learning the objectives is the first step; the next step is to practice. You need access to both SPARC and x86/x64-based systems running Solaris 10 so that you can practice what you have learned. Unless you have a supernatural memory, it's difficult to pass the exams without practice.

In the back of this book is the ExamGear software test CD that will prepare you for the questions you might see on the exam. The CD-ROM-based test engine was designed by educational experts to help you learn as you test. It is a preview of the types of questions to expect on the exams and tests your knowledge of all the exam objectives. If you are weak in any area, the sample questions will help you identify that area so that you can go back to the appropriate chapter and study the topic. Each question on the CD-ROM has a flash card to help you in case you get stuck. This flash card contains brief, concise textbook excerpts that explain why each answer is correct so that you can learn as you test.

Also, for an additional cost, you can purchase more questions for the ExamGear test engine from our website. You'll receive hundreds of questions that will take you deep into each exam objective. This will give you a comprehensive skills assessment and help you evaluate your readiness and retention of the materials.

Advice on Taking the Exam

More extensive tips are found in the "Study and Exam Prep Tips" element and throughout the book, but keep in mind the following advice as you study for the exam:

- **Read all the material.** This book includes information not reflected in the exam objectives to better prepare you for the exam and for real-world experiences. Read all the material to benefit from this.
- **Do the step-by-step lab exercises and complete the exercises in each chapter.** This will help you gain experience and prepare you for the scenario-type questions that you will encounter.
- **Use the questions to assess your knowledge.** Each chapter contains review questions and exam questions. Use these to asses your knowledge and determine where you need to review material.

- **Review the exam objectives.** Develop your own questions and examples for each topic listed. If you can develop and answer several questions for each topic, you should not find it difficult to pass the exam.
- **Relax and sleep before taking the exam.** The time for taking the examination is limited. However, if you have prepared and you know Solaris network administration, you will have plenty of time to answer all the questions. Be sure to sleep well the night before the exam because of the stress that the time limitations put on you.
- **Review all the material in the "Fast Facts" element the night before or the morning you take the exam.**
- **If you don't know the answer to a question, just skip it and don't waste time.** You need to complete the exam in the time allotted. Don't be lazy during the examination; answer all the questions as quickly as possible. Any unfinished questions will be marked incorrect.
- **Visit my website, www.UnixEd.com. It contains the following:**
  - Late-breaking changes that Sun might make to the exam or the objectives. You can expect Sun to change the exams frequently. Make sure you check my website before taking the exam.
  - A FAQs page with frequently asked questions and errata regarding this book or the exams.
  - Links to other informative websites.
  - Additional practice questions and sample exams for the ExamGear test engine. The ExamGear test engine has hundreds of questions that you can use to further assess your retention of the material presented in the book. The exams feature electronic flash cards that take the place of those sticky notes that you've used as bookmarks throughout the book. Don't attempt the real exam until you can pass every section of the practice exams with a 95% or better score.
  - An online forum where you can discuss certification-related issues with me and other system administrators, including some who have already taken the exam.
  - Additional study materials, training programs, and online seminars related to Solaris certification.
  - You can also email me directly from this website with questions or comments about this book. I always try to answer each one.

When you feel confident, take the real exams and become certified. Don't forget to drop me an email and let me know how you did on the exam (guru@UnixEd.com).

4
FOUR

Controlling Access and Configuring System Messaging

Objectives

The following test objectives for Exam CX-310-202 are covered in this chapter:

Configure Role-Based Access Control (RBAC), including assigning rights profiles, roles, and authorizations to users.

- **This chapter describes Role-Based Access Control (RBAC), and identifies the four main databases involved with RBAC. The system administrator needs to understand the function and structure of each of these databases and how to apply the RBAC functionality in real-world situations.**

Analyze RBAC configuration file summaries and manage RBAC using the command line.

- **You will see how to assign a role to a user and use rights profiles by using commands that are described in this chapter. These can greatly assist the system administrator when managing a large number of rights that are to be assigned to a number of users.**

Explain `syslog` function fundamentals, and configure and manage the `/etc/syslog.conf` file and `syslog` messaging.

- **This chapter describes the basics of system messaging in the Solaris operating environment, introduces the daemon responsible for managing the messaging, and describes the configuration file that determines what information is logged and where it is stored. It also describes the new method of restarting/refreshing the `syslog` process when changes are made to its configuration file.**

Outline

Introduction

Role-Based Access Control (RBAC)

- Using RBAC
- RBAC Components
  - Extended User Attributes (user_attr) Database
  - Authorizations (auth_attr) Database
  - Rights Profiles (prof_attr) Database
  - Execution Attributes (exec_attr) Database

syslog

- Using the logger Command

Summary

- Key Terms

Apply Your Knowledge

- Exercise
- Exam Questions
- Answers to Exam Questions

Suggested Reading and Resources

Study Strategies

The following strategies will help you prepare for the test:

- As you study this chapter, it's important that you practice each exercise and each command that is presented on a Solaris system. Hands-on experience is important when learning these topics, so practice until you can repeat the procedures from memory.
- Be sure you understand each command and be prepared to match the command to the correct description.
- Be sure you know all the terms listed in the "Key Terms" section near the end of this chapter. Pay special attention to the databases used in Role-Based Access Control (RBAC) and the uses and format of each. Be prepared to match the terms presented in this chapter with the correct description.
- Finally, you must understand the concept of system messaging—its purpose, how it works, and how to configure and manage it.

Introduction

This chapter covers two main topics—Role-Based Access Control (RBAC) and system messaging (`syslog`). These are both related in that they participate in the securing and monitoring of systems in a Solaris environment. The use of Role-Based Access Control makes the delegation of authorizations much easier for the system administrator to manage, as groups of privileges can easily be given to a role through the use of profiles. Also, the use of roles means that a user has to first log in using his or her normal ID and then use the `su` command to gain access to the role (and therefore assigned privileges). This has the advantage of being logged and therefore helps establish accountability. The system messaging service (`syslog`) stores important system and security messages and is fully configurable. The system administrator can tune the service so that certain messages are delivered to several places (such as a log file, a message, and the system console), greatly increasing the chances of it being noticed quickly.

Role-Based Access Control (RBAC)

Objectives

- Configure Role-Based Access Control (RBAC) including assigning rights profiles, roles, and authorizations to users.
- Analyze RBAC configuration file summaries and manage RBAC using the command line.

Granting superuser access to nonroot users has always been an issue in UNIX systems. In the past, you had to rely on a third-party package, such as `sudo`, to provide this functionality. The problem was that `sudo` was an unsupported piece of freeware that had to be downloaded from the Internet and installed onto your system. In extreme cases, the system administrator had to set the `setuid` permission bit on the file so that a user could execute the command as root.

With Role-Based Access Control (RBAC) in the Solaris 10 operating environment, administrators can not only assign limited administrative capabilities to nonroot users, they can also provide the mechanism where a user can carry out a specific function as another user (if required). This is achieved through three features:

- **Authorizations**: User rights that grant access to a restricted function.
- **Execution profiles**: Bundling mechanisms for grouping authorizations and commands with special attributes; for example, user and group IDs or superuser ID.
- **Roles**: Special type of user accounts intended for performing a set of administrative tasks.

CAUTION

Assigning superuser access using RBAC Most often, you will probably use RBAC to provide superuser access to administrative tasks within the system. Exercise caution and avoid creating security lapses by providing access to administrative functions by unauthorized users.

Using RBAC

To better describe RBAC, it's easier to first describe how a system administrator would utilize RBAC to delegate an administrative task to a nonroot user in a fictional setting at Acme Corp.

At Acme Corp., the system administrator is overwhelmed with tasks. He decides to delegate some of his responsibility to Neil, a user from the engineering department who helps out sometimes with system administration tasks.

The system administrator first needs to define which tasks he wants Neil to help with. He has identified three tasks:

- Change user passwords, but do not add or remove accounts.
- Mount and share file systems.
- Shut down the system.

In RBAC, when we speak of delegating administrative tasks, it is referred to as a role account. A *role account* is a special type of user account that is intended for performing a set of administrative tasks. It is like a normal user account in most respects except that users can gain access to it only through the `su` command after they have logged in to the system with their normal login account. A role account is not accessible for normal logins, for example, through the CDE login window. From a role account, a user can access commands with special attributes, typically the superuser privilege, which are unavailable to users with normal accounts.

At Acme Corp., the system administrator needs to define a role username for the tasks he wants to delegate. Let's use the role username "adminusr." After Neil logs in with his normal login name of `ncalkins`, he then needs to issue the `su` command and switch to `adminusr` whenever he wants to perform administrative tasks. In this chapter, you learn how to create a role account using the command line interface, although you should note that the Solaris Management Console can also be used.

So far we have determined that we want to name the role account `adminusr`. The system administrator creates the role account using the `roleadd` command. The `roleadd` command adds a role account to the `/etc/passwd`, `etc/shadow`, and `/etc/user_attr` files. The syntax for the `roleadd` command is as follows:

```
roleadd [-c comment] [-d dir] [-e expire] [-f inactive] [-g group] \
[-G group] [-m] [-k skel_dir] [-u uid] [-s shell] \
[-A authorization] [-P profile ] <role username>
```

You'll notice that `roleadd` looks a great deal like the `useradd` command. Table 4.1 describes the options for the `roleadd` command.

Table 4.1 `roleadd` Options

Option	Description
`-c <comment>`	Any text string to provide a brief description of the role.
`-d <dir>`	The home directory of the new role account.
`-m`	Creates the new role's home directory if it does not already exist.
`-e <expire>`	Specifies the expiration date for a role. After this date, no user can access this role. The `<expire>` option argument is a date entered using one of the date formats included in the template file `/etc/datemsk`. For example, you can enter 10/30/02 or October 30, 2002. A value of `" "` defeats the status of the expired date.
`-f <inactive>`	Specifies the maximum number of days allowed between uses of a login ID before that login ID is declared invalid. Normal values are positive integers.
`-g <group>`	Specifies an existing group's integer ID or character-string name. It redefines the role's primary group membership.
`-G <group>`	Specifies an existing group's integer ID, or character string name. It redefines the role's supplementary group membership. Duplicates between groups with the `-g` and `-G` options are ignored.
`-k <skeldir>`	A directory that contains skeleton information (such as `.profile`) that can be copied into a new role's home directory. This directory must already exist. The system provides the `/etc/skel` directory that can be used for this purpose.
`-s <shell>`	Specifies the user's shell on login. The default is `/bin/pfsh`.
`-A <authorization>`	Both of these options respectively assign authorizations and profiles to the role.
`-P <profile>`	Authorizations and profiles are described later in this section.
`-u <uid>`	Specifies a UID for the new role. It must be a nonnegative decimal integer. The UID associated with the role's home directory is not modified with this option; a role does not have access to its home directory until the UID is manually reassigned using the `chown` command.

The other options are the same options that were described for the `useradd` command, outlined in *Solaris 10 System Administration Exam Prep: CX-310-200, Part I*.

When creating a role account with the `roleadd` command, you need to specify an authorization or profile to the role. An *authorization* is a user right that grants access to a restricted function. It is a unique string that identifies what is being authorized as well as who created the authorization.

Certain privileged programs check the authorizations to determine whether users can execute restricted functionality. Following are the predefined authorizations from the `/etc/security/auth_attr` file that apply to the tasks to be delegated:

```
solaris.admin.usermgr.pswd:::Change Password::help=AuthUserMgrPswd.html
solaris.system.shutdown:::Shutdown the System::help=SysShutdown.html
solaris.admin.fsmgr.write:::Mount and Share File Systems::\
help=AuthFsMgrWrite.html
```

All authorizations are stored in the `auth_attr` database, so the system administrator needs to use one or more of the authorizations that are stored in that file. For the Acme Corp. example, the system administrator needs to specify the authorizations shown here:

```
solaris.admin.usermgr.pswd
solaris.system.shutdown
solaris.admin.fsmgr.write
```

The system administrator would therefore issue the `roleadd` command as follows:

```
# roleadd -m -d /export/home/adminusr -c "Admin Assistant" \
-A solaris.admin.usermgr.pswd,solaris.system.shutdown,\
solaris.admin.fsmgr.write adminusr<cr>
```

A role account named `adminusr` with the required directory structures has been created. The next step is to set the password for the `adminusr` role account by typing the following:

```
passwd adminusr
```

You are prompted to type the new password twice.

Now we need to set up Neil's account so he can access the new role account named `adminusr`. With the `usermod` command, we assign the role to the user account using the `-R` option:

```
usermod -R adminusr neil
```

NOTE

No need to be logged out Previously, you needed to ensure that the user was not logged in at the time of assigning a role; otherwise, you received an error message and the role was not assigned. This is no longer the case. A role can be assigned to a user while the user is still logged in.

To access the administrative functions, Neil needs to first log in using his regular user account named `neil`. Neil can check which roles he has been granted by typing the following at the command line:

```
$ roles<cr>
```

The system responds with the roles that have been granted to the user account `neil`:

```
adminusr
```

Neil then needs to su to the adminusr account by typing the following:

```
$ su adminusr<cr>
```

Neil is prompted to type the password for the role account.

Now Neil can modify user passwords, shut down the system, and mount and share file systems. Any other user trying to su to the adminusr account gets this message:

```
$ su adminusr<cr>
Password:
Roles can only be assumed by authorized users
su: Sorry
$
```

If the system administrator later wants to assign additional authorizations to the role account named adminusr, he would do so using the rolemod command. The rolemod command modifies a role's login information on the system. The syntax for the rolemod command is as follows:

```
rolemod [-u uid] [-o] [-g group] [-G group] [-d dir] [-m] [-s shell]\
[-c comment] [-l new_name] [-f inactive] [-e expire] [-A Authorization]\
[-P profile] <role account>
```

Table 4.2 describes options for the rolemod command where they differ from the roleadd command.

Table 4.2 `rolemod` Options

Option	Description
`-A <authorization>`	One or more comma-separated authorizations as defined in the `auth_attr` database. This replaces any existing authorization setting.
`-d <dir>`	Specifies the new home directory of the role. It defaults to `<base_dir>/<login>`, in which `<base_dir>` is the base directory for new login home directories, and `<login>` is the new login.
`-l <new_logname>`	Specifies the new login name for the role. The `<new_logname>` argument is a string no more than eight bytes consisting of characters from the set of alphabetic characters, numeric characters, period (.), underline (_), and hyphen (-). The first character should be alphabetic and the field should contain at least one lowercase alphabetic character. A warning message is written if these restrictions are not met. A future Solaris release might refuse to accept login fields that do not meet these requirements. The `<new_logname>` argument must contain at least one character and must not contain a colon (`:`) or newline (`\n`).
`-m`	Moves the role's home directory to the new directory specified with the `-d` option. If the directory already exists, it must have permissions read/write/execute by group, in which group is the role's primary group.

Table 4.2 `rolemod` Options

Option	Description
`-o`	Allows the specified UID to be duplicated (nonunique).
`-P <profile>`	Replaces any existing profile setting. One or more comma-separated execution profiles are defined in the `auth_attr` database.
`-u <uid>`	Specifies a new UID for the role. It must be a nonnegative decimal integer. The UID associated with the role's home directory is not modified with this option; a role does not have access to its home directory until the UID is manually reassigned using the `chown` command.

To add the ability to purge log files, you need to add `solaris.admin.logsvc.purge` to the list of authorizations for `adminusr`. To do this, issue the `rolemod` command:

```
# rolemod -A solaris.admin.usermgr.pswd,solaris.system.shutdown,\
solaris.admin.fsmgr.write,solaris.admin.logsvc.purge  adminusr<cr>
```

You can verify that the new authorizations have been added to the role by typing the `auths` command at the command line:

```
# auths adminusr<cr>
solaris.admin.usermgr.pswd,solaris.system.shutdown,solaris.admin.fsmgr.\
write,solaris.admin.logsvc.purge,...
[ output has been truncated]
```

CAUTION

`rolemod` warning The `rolemod` command does not add to the existing authorizations; it replaces any existing authorization setting.

If you want to remove a role account, use the `roledel` command:

```
roledel [-r] <role account name>
```

The `-r` option removes the role's home directory from the system. For example, to remove the `adminusr` role account, issue the following command:

```
# roledel -r adminusr<cr>
```

The next section discusses each of the RBAC databases in detail, describing the entries made when we executed the `roleadd` and `usermod` commands.

RBAC Components

RBAC relies on the following four databases to provide users access to privileged operations:

- **`/etc/user_attr` (extended user attributes database)**: Associates users and roles with authorizations and profiles.
- **`/etc/security/auth_attr` (authorization attributes database)**: Defines authorizations and their attributes and identifies the associated help file.
- **`/etc/security/prof_attr` (rights profile attributes database)**: Defines profiles, lists the profile's assigned authorizations, and identifies the associated help file.
- **`/etc/security/exec_attr` (profile attributes database)**: Defines the privileged operations assigned to a profile.

These four databases are logically interconnected.

EXAM ALERT

RBAC database functions You need to be able to correctly identify the function and location of each RBAC database. A common exam question is to match the description with the relevant RBAC database. Remember that the `user_attr` database resides in the `/etc` directory and not in the `/etc/security` directory.

Extended User Attributes (`user_attr`) Database

The `/etc/user_attr` database supplements the `passwd` and `shadow` databases. It contains extended user attributes, such as authorizations and profiles. It also allows roles to be assigned to a user. Following is an example of the `/etc/user_attr` database:

```
# more /etc/user_attr<cr>
# Copyright  2003 by Sun Microsystems, Inc. All rights reserved.
#
# /etc/user_attr
#
# user attributes. see user_attr(4)
#
#pragma ident   "@(#)user_attr  1.1     03/07/09 SMI"
#
adm::::profiles=Log Management
lp::::profiles=Printer Management
root::::auths=solaris.*,solaris.grant;profiles=All
adminusr::::type=role;auths=solaris.admin.usermgr.pswd,/
solaris.system.shutdown,solaris.admin.fsmgr.write;profiles=All
neil::::type=normal;roles=adminusr
```

The following fields in the `user_attr` database are separated by colons:

```
user:qualifier:res1:res2:attr
```

Each field is described in Table 4.3.

Table 4.3 `user_attr` Fields

Field Name	Description
`user`	Describes the name of the user or role, as specified in the `passwd` database.
`qualifier`	Reserved for future use.
`res1`	Reserved for future use.
`res2`	Reserved for future use.
`attr`	Contains an optional list of semicolon-separated (;) key-value pairs that describe the security attributes to be applied when the user runs commands. Eight valid keys exist: `auths`, `profiles`, `roles`, `type`, `project`, `defaultpriv`, `limitpriv`, and `lock_after_retries`: `auths` specifies a comma-separated list of authorization names chosen from names defined in the `auth_attr` database. Authorization names can include the asterisk (*) character as a wildcard. For example, `solaris.device.*` means all the Solaris device authorizations. `profiles` contains an ordered, comma-separated list of profile names chosen from `prof_attr`. A profile determines which commands a user can execute and with which command attributes. At a minimum, each user in `user_attr` should have the `All` profile, which makes all commands available but without attributes. The order of profiles is important; it works similarly to UNIX search paths. The first profile in the list that contains the command to be executed defines which (if any) attributes are to be applied to the command. Profiles are described in the section titled "Authorizations (`auth_attr`) Database." `roles` can be assigned to the user using a comma-separated list of role names. Note that roles are defined in the same `user_attr` database. They are indicated by setting the `type` value to `role`. Roles cannot be assigned to other roles. `type` can be set to normal, if this account is for a normal user, or to role, if this account is for a role. A normal user assumes a role after he has logged in. `project` can be set to a project from the projects database, so that the user is placed in a default project at login time. `defaultpriv` is the list of default privileges the user is assigned. `limitpriv`: The system administrator can limit the set of privileges allowed, and this attribute contains the maximum set of privileges the user can be allowed. Care must be taken when limiting privileges so as to not affect other applications the user might execute.

`lock_after_retries` specifies whether an account is locked out following a number of failed logins. The number of failed logins is taken from the `RETRIES` option in `/etc/default/login`. The default is `no`.

In the previous section, we issued the following `roleadd` command to add a role named `adminusr`:

```
# roleadd -m -d /export/home/adminusr -c "Admin Assistant"\
 -A solaris.admin.usermgr.pswd,solaris.system.shutdown,\
solaris.admin.fsmgr.write adminusr<cr>
```

The `roleadd` command made the following entry in the `user_attr` database:

```
adminusr::::type=role;auths=solaris.admin.usermgr.pswd,\
solaris.system.shutdown,solaris.admin.fsmgr.write;profiles=All
```

We can then issue the following `usermod` command to assign the new role to the user `neil`:

```
# usermod -R useradmin neil<cr>
```

and then make the following entry to the `user_attr` database:

```
neil::::type=normal;roles=adminusr
```

Authorizations (`auth_attr`) Database

An authorization is a user right that grants access to a restricted function. In the previous section, the system administrator wanted to delegate some of the system administrative tasks to Neil. Assigning authorizations to the role named `adminusr` did this. An authorization is a unique string that identifies what is being authorized as well as who created the authorization. Remember that we used the following authorizations to give Neil the ability to modify user passwords, shut down the system, and mount and share file systems:

```
solaris.admin.usermgr.pswd
solaris.system.shutdown
solaris.admin.fsmgr.write
```

Certain privileged programs check the authorizations to determine whether users can execute restricted functionality. For example, the `solaris.jobs.admin` authorization is required for one user to edit another user's `crontab` file.

All authorizations are stored in the `auth_attr` database. If no name service is used, the database is located in a file named `/etc/security/auth_attr`. Authorizations can be assigned directly to users (or roles), in which case they are entered in the `user_attr` database. Authorizations can also be assigned to profiles, which in turn are assigned to users. They are described in the "Rights Profiles (`prof_attr`) Database" section, later in this chapter.

The fields in the `auth_attr` database are separated by colons, as shown here:

```
authname:res1:res2:short_desc:long_desc:attr
```

Each field is described in Table 4.4.

Table 4.4 `auth_attr` Fields

Field Name	Description
`authname[suffix]`	A unique character string used to identify the authorization in the format `prefix`. Authorizations for the Solaris operating environment use `solaris` as a prefix. All other authorizations should use a prefix that begins with the reverse-order Internet domain name of the organization that creates the authorization (for example, `com.xyzcompany`). The suffix indicates what is being authorized—typically, the functional area and operation. When no suffix exists (that is, the `authname` consists of a prefix and functional area and ends with a period), the `authname` serves as a heading for use by applications in their GUIs rather than as an authorization. The `authname` `solaris.printmgr` is an example of a heading. When the `authname` ends with the word `grant`, the `authname` serves as a grant authorization and allows the user to delegate related authorizations (that is, authorizations with the same prefix and functional area) to other users. The `authname` `solaris.printmgr.grant` is an example of a grant authorization; it gives the user the right to delegate such authorizations as `solaris.printmgr.admin` and `solaris.printmgr.nobanner` to other users.
`res1`	Reserved for future use.
`res2`	Reserved for future use.
`short_desc`	A shortened name for the authorization suitable for displaying in user interfaces, such as in a scrolling list in a GUI.
`long_desc`	A long description. This field identifies the purpose of the authorization, the applications in which it is used, and the type of user interested in using it. The long description can be displayed in the help text of an application.
`attr`	An optional list of semicolon-separated (;) key-value pairs that describe the attributes of an authorization. Zero or more keys can be specified. The keyword `help` identifies a help file in HTML. Help files can be accessed from the `index.html` file in the `/usr/lib/help/auths/locale/C` directory.

The following are some typical values found in the default `auth_attr` database:

```
solaris.admin.usermgr.pswd:::Change Password::help=AuthUserMgrPswd.html
solaris.system.shutdown:::Shutdown the System::help=SysShutdown.html
solaris.admin.fsmgr.write:::Mount and Share File Systems::\
help=AuthFsMgrWrite.html
```

Look at the relationship between the `auth_attr` and the `user_attr` databases for the `adminusr` role we added earlier:

```
adminusr::::type=role;auths=solaris.admin.usermgr.pswd,\
solaris.system.shutdown,solaris.admin.fsmgr.write;profiles=All
```

Notice the authorization entries that are **bold**. These authorization entries came out of the `auth_attr` database, shown previously. The `solaris.system.shutdown` authorization, which is defined in the `auth_attr` database, gives the role the right to shut down the system.

Rights Profiles (`prof_attr`) Database

We referred to rights profiles, or simply profiles, earlier in this chapter. Up until now, we assigned authorization rights to the role account. Defining a role account that has several authorizations can be tedious. In this case, it's better to define a profile, which is several authorizations bundled together under one name called a *profile name*. The definition of the profile is stored in the `prof_attr` database. Following is an example of a profile named `Operator`, which is in the default `prof_attr` database. Again, if you are not using a name service, the `prof_attr` file is located in the `/etc/security` directory.

```
Operator:::Can perform simple administrative tasks:profiles=Printer
Management,Media Backup,All;help=RtOperator.html
```

Several other profiles are defined in the `prof_attr` database. Colons separate the fields in the `prof_attr` database:

```
profname:res1:res2:desc:attr
```

The fields are defined in Table 4.5.

Table 4.5 `prof_attr` Fields

Field Name	Description
`profname`	The name of the profile. Profile names are case-sensitive.
`res1`	A field reserved for future use.
`res2`	A field reserved for future use.
`desc`	A long description. This field should explain the purpose of the profile, including what type of user would be interested in using it. The long description should be suitable for displaying in the help text of an application.
`attr`	An optional list of key-value pairs separated by semicolons (;) that describe the security attributes to apply to the object upon execution. Zero or more keys can be specified. The four valid keys are `help`, `auths`, `privs`, and `profiles`. The keyword `help` identifies a help file in HTML. Help files can be accessed from the `index.html` file in the `/usr/lib/help/auths/locale/C` directory. `auths` specifies a comma-separated list of authorization names chosen from those names defined in the `auth_attr` database. Authorization names can be specified using the asterisk (`*`) character as a wildcard.

Perhaps the system administrator wants to create a new role account and delegate the task of printer management and backups. He could look through the `auth_attr` file for each authorization and assign each one to the new role account using the `roleadd` command, as described earlier. Or, he could use the `Operator` profile currently defined in the `prof_attr` database, which looks like this:

The `Operator` profile consists of three other profiles:

- Printer Management
- Media Backup
- All

Let's look at each of these profiles as defined in the `prof_attr` database:

```
Printer Management:::Manage printers, daemons, spooling:help=RtPrntAdmin.\
html;auths=solaris.admin.printer.read,solaris.admin.printer.modify,\
solaris.admin.printer.delete
Media Backup:::Backup files and file systems:help=RtMediaBkup.html
All:::Execute any command as the user or role:help=RtAll.html
```

`Printer Management` has the following authorizations assigned to it:

- `solaris.admin.printer.read`
- `solaris.admin.printer.modify`
- `solaris.admin.printer.delete`

When you look at these three authorizations in the `auth_attr` database, you see the following entries:

```
solaris.admin.printer.read:::View Printer Information::help=AuthPrinterRead.html
solaris.admin.printer.modify:::Update Printer Information::help=AuthPrinterModify.html
solaris.admin.printer.delete:::Delete Printer Information::help=AuthPrinterDelete.html
```

Assigning the `Printer Management` profile is the same as assigning the three authorizations for viewing, updating, and deleting printer information.

The `Media Backup` profile provides authorization for backing up data, but not restoring data. The `Media Backup` profile does not have authorizations associated with it like the `Printer Management` profile has. I'll describe how this profile is defined in the next section when I describe execution attributes.

The `All` profile grants the right for a role account to use any command when working in an administrator's shell. These shells can only execute commands that have been explicitly assigned to a role account through granted rights. We'll explore this concept further when I describe execution attributes in the next section.

To create a new role account named `admin2` specifying the `Operator` profile, use the `roleadd` command with the `-P` option:

```
# roleadd -m -d /export/home/admin2 -c "Admin Assistant" -P Operator admin2<cr>
```

The following entry is added to the `user_attr` database:

```
admin2::::type=role;profiles=Operator
```

At any time, users can check which profiles have been granted to them with the `profiles` command:

```
$ profiles<cr>
```

The system lists the profiles that have been granted to that particular user account.

Execution Attributes (`exec_attr`) Database

An execution attribute associated with a profile is a command (with any special security attributes) that can be run by those users or roles to which the profile is assigned. For example, in the previous section, we looked at the profile named `Media Backup` in the `prof_attr` database. Although no authorizations were assigned to this profile, the `Media Backup` profile was defined in the `exec_attr` database as follows:

```
Media Backup:solaris:act:::Tar;*;*;*;*:privs=all
Media Backup:solaris:act:::Tar;*;TAR,MAGTAPE;*;>0:privs=all
Media Backup:solaris:act:::TarList;*;*;*;*:
Media Backup:suser:cmd:::/usr/bin/mt:euid=0
Media Backup:suser:cmd:::/usr/lib/fs/ufs/ufsdump:euid=0;gid=sys
Media Backup:suser:cmd:::/usr/sbin/tar:euid=0
```

The fields in the `exec_attr` database are as follows and are separated by colons:

```
name:policy:type:res1:res2:id:attr
```

The fields are defined in Table 4.6.

Table 4.6 `exec_attr` Fields

Field Name	Description
`Name`	The name of the profile. Profile names are case-sensitive.
`policy`	The security policy associated with this entry. Currently, `suser` (the superuser policy model) and `solaris` are the only valid policy entries. The `solaris` policy recognizes privileges, whereas the `suser` policy does not.
`type`	The type of entity whose attributes are specified. The two valid types are `cmd` (command) and `act`. The `cmd` type specifies that the ID field is a command that would be executed by a shell. The `act` type is available only if the system is configured with Trusted Extensions.
`res1`	This field is reserved for future use.
`res2`	This field is reserved for future use.
`id`	A string identifying the entity; the asterisk (*) wildcard can be used. Commands should have the full path or a path with a wildcard. To specify arguments, write a script with the arguments and point the id to the script.
`attr`	An optional list of semicolon (;) separated key-value pairs that describe the security attributes to apply to the entity upon execution. Zero or more keys can be specified. The list of valid keywords depends on the policy being enforced. Six valid keys exist: `euid`, `uid`, `egid`, `gid`, `privs`, and `limitprivs`. `euid` and `uid` contain a single username or numeric user ID. Commands designated with `euid` run with the effective UID indicated, which is similar to setting the `setuid` bit on an executable file. Commands designated with `uid` run with both the real and effective UIDs. `egid` and `gid` contain a single group name or numeric group ID. Commands designated with `egid` run with the effective GID indicated, which is similar to setting the `setgid` bit on an executable file. Commands designated with `gid` run with both the real and effective GIDs.

NOTE

Trusted Solaris You will see an additional security policy if you are running Trusted Solaris, a special security-enhanced version of the operating environment. The policy `tsol` is the trusted solaris policy model.

Looking back to the `Media Backup` profile as defined in the `exec_attr` database, we see that the following commands have an effective UID of 0 (superuser):

```
/usr/bin/mt
/usr/sbin/tar
/usr/lib/fs/ufs/ufsdump
```

Therefore, any user that has been granted the `Media Backup` profile can execute the previous backup commands with an effective user ID of 0 (superuser).

In the `prof_attr` database, we also saw that the `Operator` profile consisted of a profile named `All`. Again, `All` did not have authorizations associated with it. When we look at the `exec_attr` database for a definition of the `All` profile, we get the following entry:

```
All:suser:cmd:::*:
```

Examining each field, we see that `All` is the profile name, the security policy is `suser`, and the type of entity is `cmd`. The attribute field has an `*`.

It's common to grant all users the `All` profile. The `*` is a wildcard entry that matches every command. In other words, the user has access to any command while working in the shell. Without the `All` profile, a user would have access to the privileged commands, but no access to normal commands such as `ls` and `cd`. Notice that no special process attributes are associated with the wildcard, so the effect is that all commands matching the wildcard run with the UID and GID of the current user (or role).

NOTE

The `All` profile Always assign the `All` profile last in the list of profiles. If it is listed first, no other rights are consulted when you look up command attributes.

syslog

Objective

- Explain `syslog` function fundamentals and configure and manage the `/etc/syslog.conf` file and `syslog` messaging.

A critical part of the system administrator's job is monitoring the system. Solaris uses the `syslog` message facility to do this. `syslogd` is the daemon responsible for capturing system messages. The messages can be warnings, alerts, or simply informational messages. As the system administrator, you customize `syslog` to specify where and how system messages are to be saved.

The `syslogd` daemon receives messages from applications on the local host or from remote hosts and then directs messages to a specified log file. To each message that `syslog` captures, it adds a timestamp, the message type keyword at the beginning of the message, and a newline at the end of the message. For example, the following messages were logged in the `/var/adm/messages` file:

```
July 15 23:06:39 sunfire ufs: [ID 845546 kern.notice] NOTICE: alloc: /var: \
file system full
Sep  1 04:57:06 docbert nfs: [ID 563706 kern.notice] NFS server saturn.east ok
```

`syslog` enables you to capture messages by facility (the part of the system that generated the message) and by level of importance. Facility is considered to be the service area generating the message or error (such as printing, email, or network), whereas the level can be considered the level of severity (such as notice, warning, error, or emergency). `syslog` also enables you to forward messages to another machine so that all your messages can be logged in one location. The `syslogd` daemon reads and logs messages into a set of files described by the configuration file `/etc/syslog.conf`. When the `syslogd` daemon starts up, it preprocesses the `/etc/syslog.conf` file through the `m4` macro processor to get the correct information for specific log files. `syslogd` does not read the `/etc/syslog.conf` file directly. `syslogd` starts `m4`, which parses the `/etc/syslog.conf` file for `ifdef` statements that can be interpreted by `m4`. The function `ifdef` is an integral part of `m4` and identifies the system designated as `LOGHOST`. The macro then can evaluate whether log files are to be held locally or on a remote system, or a combination of both.

If m4 doesn't recognize any m4 commands in the `syslog.conf` file, output is passed back to `syslogd`. `syslogd` then uses this output to route messages to appropriate destinations. When m4 encounters `ifdef` statements that it can process, the statement is evaluated for a true or false condition and the message is routed relative to the output of the test.

EXAM ALERT

/`etc/syslog.conf` and `ifdef` statements Make sure you become familiar with the facilities and values listed in the tables in this section. An exam question might provide a sample file and ask where a specific type of message, such as a failed login, will be logged. Also watch out for the `ifdef` statements to see if the logging is being carried out on a remote system.

An entry in the `/etc/syslog.conf` file is composed of two fields:

```
selector     action
```

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

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

The `action` field indicates where to forward the message. Many defined facilities exist.

EXAM ALERT

Separate with tabs The separator between the two fields must be a tab character. Spaces do not work and give unexpected results. This is a very common mistake.

The facilities are described in Table 4.7.

Table 4.7 Recognized Values for Facilities

Value	Description
`user`	Messages generated by user processes. This is the default priority for messages from programs or facilities not listed in this file.
`kern`	Messages generated by the kernel.
`mail`	The mail system.
`daemon`	System daemons, such as `in.ftpd`.
`auth`	The authorization system, such as `login`, `su`, `getty`, and others.
`lpr`	`lpr` is the `syslogd` facility responsible for generating messages from the line printer spooling system—`lpr` and `lpc`.
`news`	Reserved for the Usenet network news system.
`uucp`	Reserved for the UUCP system. It does not currently use the `syslog` mechanism.
`cron`	The `cron/at` facility, such as `crontab`, `at`, `cron`, and others.
`audit`	The `audit` facility, such as `auditd`.
`local0-7`	Reserved for local use.
`mark`	For timestamp messages produced internally by `syslogd`.
`*`	Indicates all facilities except the `mark` facility.

Table 4.8 lists recognized values for the `syslog level` field. They are listed in descending order of severity.

Table 4.8 Recognized Values for `level`

Value	Description
`emerg`	Panic conditions that would normally be broadcast to all users.
`alert`	Conditions that should be corrected immediately, such as a corrupted system database.
`crit`	Warnings about critical conditions, such as hard device errors.
`err`	Other errors.
`warning`	Warning messages.
`Notice`	Conditions that are not error conditions but that might require special handling, such as a failed login attempt. A failed login attempt is considered a notice and not an error.
`info`	Informational messages.
`debug`	Messages that are normally used only when debugging a program.
`none`	Does not send messages from the indicated facility to the selected file. For example, the entry `*.debug;mail.none` in `/etc/syslog.conf` sends all messages except mail messages to the selected file.

NOTE

Levels include all higher levels too When you specify a `syslog` level, it means that the specified level and all higher levels. For example, if you specify the `err` level, this includes `crit`, `alert`, and `emerg` levels as well.

Values for the `action` field can have one of four forms:

- A filename, beginning with a leading slash. This indicates that messages specified by the selector are to be written to the specified file. The file is opened in append mode and must already exist. `syslog` does not create the file if it doesn't already exist.
- The name of a remote host, prefixed with a `@`. An example is `@server`, which indicates that messages specified by the selector are to be forwarded to `syslogd` on the named host. The hostname `loghost` is the hostname given to the machine that will log `syslogd` messages. Every machine is its own `loghost` by default. This is specified in the local `/etc/hosts` file. It is also possible to specify one machine on a network to be `loghost` by making the appropriate host table entries. If the local machine is designated as `loghost`, `syslogd` messages are written to the appropriate files. Otherwise, they are sent to the machine `loghost` on the network.
- A comma-separated list of usernames, which indicates that messages specified by the selector are to be written to the named users if they are logged in.
- An asterisk, which indicates that messages specified by the selector are to be written to all logged-in users.

Blank lines are ignored. Lines in which the first nonwhitespace character is a `#` are treated as comments.

All of this becomes much clearer when you look at sample entries from an `/etc/syslog.conf` file:

```
*.err    /dev/console
*.err;daemon,auth.notice;mail.crit    /var/adm/messages
mail.debug   /var/log/syslog
*.alert    root
*.emerg   *
kern.err    @server
*.alert;auth.warning    /var/log/auth
```

In this example, the first line prints all errors on the console.

The second line sends all errors, daemon and authentication system notices, and critical errors from the mail system to the file `/var/adm/messages`.

The third line sends mail system debug messages to `/var/log/syslog`.

The fourth line sends all alert messages to user root.

The fifth line sends all emergency messages to all users.

The sixth line forwards kernel messages of `err` (error) severity or higher to the machine named server.

The last line logs all alert messages and messages of `warning` level or higher from the authorization system to the file `/var/log/auth`.

The level `none` may be used to disable a facility. This is usually done in the context of eliminating messages. For example:

```
*.debug;mail.none /var/adm/messages
```

This selects debug messages and above from all facilities except those from mail. In other words, mail messages are disabled. The mail system, sendmail, logs a number of messages. The mail system can produce a large amount of information, so some system administrators disable mail messages or send them to another file that they clean out frequently. Before disabling mail messages, however, remember that sendmail messages come in very handy when you're diagnosing mail problems or tracking mail forgeries.

As of Solaris 10, the mechanism for stopping, starting, and refreshing `syslogd` has changed. The `syslog` function is now under the control of the Service Management Facility (SMF), which is described in detail in the book *Solaris 10 System Administration Exam Prep: CX-310-200, Part I*.

To stop or start `syslogd`, use the `svcadm` command with the appropriate parameter, `enable` or `disable`:

```
# svcadm enable -t system-log<cr>
# svcadm disable -t system-log<cr>
```

The `syslog` facility reads its configuration information from `/etc/syslog.conf` whenever it receives a refresh command from the service administration command, `svcadm`, and when the system is booted. You can make your changes to `/etc/syslog.conf` and then run the following command to cause the file to be reread by the `syslogd` daemon:

```
# svcadm refresh system-log<cr>
```

EXAM ALERT

No more `kill -HUP` Make sure you remember that the `kill -HUP` facility should no longer be used to try to cause a daemon process to re-read its configuration file, even though it still works. The `svcadm refresh` command is now the recommended way of achieving this.

The first message in the log file is logged by the `syslog` daemon itself to show when the process was started.

`syslog` logs are automatically rotated on a regular basis. In previous Solaris releases, this was achieved by the program `newsyslog`. A new method of log rotation was introduced with Solaris 9—`logadm`, a program normally run as a root-owned cron job. A configuration file `/etc/logadm.conf` is now used to manage log rotation and allows a number of criteria to be specified. See the `logadm` and `logadm.conf` manual pages for further details.

Using the `logger` Command

The `logger` command provides the means of manually adding one-line entries to the system logs from the command line. This is especially useful in shell scripts.

The syntax for the `logger` command is as follows:

```
logger [-i] [-f file] [-p priority] [-t tag] [message] ...
```

Options to the `logger` command are described in Table 4.9.

Table 4.9 `logger` Options

Option	Description
`-i`	Logs the Process ID (PID) of the `logger` process with each line written to a log file.
`-f <file>`	Use the contents of *`file`* as the message to be logged.
`-p <priority>`	The message priority. This can be defined as a numeric value or as a `facility.level` pair, as described in Tables 4.7 and 4.8. The default priority is `user.notice`.
`-t <tag>`	Marks each line with the specified tag.
`message`	One or more string arguments, separated by a single space character comprising the text of the message to be logged.

For example, perhaps you have a simple shell script that backs up files:

```
#/bin/ksh
tar cvf /tmp/backup  .
logger -p user.alert "Backups Completed"
```

The last line of the script uses the `logger` command to send a `"Backups Completed"` message to the default system log (`/var/adm/messages`). After running the script, I see the following message appended to the log file:

```
Jan 23 14:02:52 sunfire root: [ID 702911 user.alert] Backups Completed
```

Summary

In this chapter you learned about Role-Based Access Control (RBAC), which allows the system administrator to delegate administrative responsibilities to users without having to divulge the root password. A number of profiles allow privileges to be grouped together so that a user can easily be granted a restricted set of additional privileges. Four main RBAC databases interact with each other to provide users with access to privileged operations:

- `/etc/security/auth_attr`: Defines authorizations and their attributes and identifies the associated help file.
- `/etc/security/exec_attr`: Defines the privileged operations assigned to a profile.
- `/etc/security/prof_attr`: Defines the profiles, lists the profile's assigned authorizations, and identifies the associated help file.
- `/etc/user_attr`: Associates users and roles with authorizations and execution profiles.

Also in this chapter, you learned about the system logging facility (`syslog`) and the configuration that facilitates routing of system messages according to specific criteria, as well as determining where the messages are logged. The `logger` command was covered, which allows the system administrator to enter ad-hoc messages into the system log files.

Key Terms

- Authorization
- Execution profile
- `logger`
- RBAC
- RBAC databases (know about all four)
- Rights profile
- Role
- `syslog`
- `svcadm` command

Apply Your Knowledge

Exercise

4.1 Creating a User and a Role

In this exercise, you'll create a new role named `admin1` and a profile called `Shutdown`. The `Shutdown` profile will be added to the role. A user account trng1 will be created and have the `admin1` role assigned to it. The user will then assume the role and execute a privileged command to shut down the system.

Estimated time: 20 minutes

To create a user and a role, follow these steps:

1. Create the role named `admin1`:

```
# roleadd -u 2000 -g 10 -d /export/home/admin1 -m admin1<cr>
# passwd admin1<cr>
```

   You are prompted to enter the password twice.

2. Create a profile to allow the user to shut down a system.

   Edit the `/etc/security/prof_attr` file and enter the following line:

```
Shutdown:::Permit system shutdown:
```

   Save and exit the file.

3. Add the `Shutdown` and `All` profiles to the role:

```
# rolemod -P Shutdown,All admin1<cr>
```

4. Verify that the changes have been made to the `user_attr` database:

```
# more /etc/user_attr<cr>
```

5. Create the user account and assign it access to the admin1 role:

```
# useradd -u 3000 -g 10 -d /export/home/trng1 -m -s /bin/ksh -R admin1 trng1<cr>
```

6. Assign a password to the new user account:

```
# passwd trng1<cr>
```

   You are prompted to enter the password twice.

7. Verify that the entry has been made to the passwd, shadow, and `user_attr` files:

```
# more /etc/passwd<cr>
# more /etc/shadow<cr>
# more /etc/user_attr<cr>
```

8. Assign commands to the `Shutdown` profile:

   Edit the `/etc/security/exec_attr` file and add the following line:

   ```
   Shutdown:suser:cmd:::/usr/sbin/shutdown:uid=0
   ```

   Save and exit the file.

9. Test the new role and user account as follows:

   a. Log in as trng1.

   b. List the roles that are granted to you by typing the following:

   ```
   $ roles<cr>
   ```

   c. Use the `su` command to assume the role admin1:

   ```
   $ su admin1<cr>
   ```

   You are prompted to enter the password for the role.

   d. List the profiles that are granted to you by typing the following:

   ```
   $ profiles<cr>
   ```

   e. Shut down the system:

   ```
   $ /usr/sbin/shutdown -i 0 -g 0<cr>
   ```

Exam Questions

1. Which of the following commands is used to create a role?

   - ❍ **A.** `useradd`
   - ❍ **B.** `makerole`
   - ❍ **C.** `roleadd`
   - ❍ **D.** `addrole`

2. In Role-Based Access Control, which file contains details of the user attributes?

   - ❍ **A.** `/etc/security/prof_attr`
   - ❍ **B.** `/etc/user_attr`
   - ❍ **C.** `/etc/security/user_attr`
   - ❍ **D.** `/etc/shadow`

3. Which two statements about the `roleadd` command are true? (Choose two.)

   - ❍ **A.** `roleadd` looks similar to the `useradd` command.
   - ❍ **B.** `roleadd` uses the profile shell (`pfsh`) as the default shell.
   - ❍ **C.** The `-A` option associates an account with a profile.
   - ❍ **D.** An account created with `roleadd` is the same as a normal login account.

4. Which component of RBAC associates users and roles with authorizations and profiles?

   - ❍ **A.** `user_attr`
   - ❍ **B.** `prof_attr`
   - ❍ **C.** `auth_attr`
   - ❍ **D.** `exec_attr`

5. Which component of RBAC defines the privileged operations assigned to a profile?

   - ❍ **A.** `user_attr`
   - ❍ **B.** `prof_attr`
   - ❍ **C.** `auth_attr`
   - ❍ **D.** `exec_attr`

6. In the execution attributes database, which of the following is not a valid value for the `attr` field?

   - ❍ **A.** `euid`
   - ❍ **B.** `uid`
   - ❍ **C.** `egid`
   - ❍ **D.** `suid`

7. After creating an RBAC role, you find that the only commands that can be executed within the role are the privileged commands that you have set up. Ordinary nonprivileged commands are not available. The RBAC setup has a problem. What is the cause of this problem?

   - ❍ **A.** The role is not associated with a correct profile.
   - ❍ **B.** The access mechanism to the role is not initializing properly.
   - ❍ **C.** The role's profile is not associated with the correct commands.
   - ❍ **D.** The file identifying the privileged commands has missing entries.
   - ❍ **E.** The role's profile is not associated with the correct authorizations.

8. Which of the following are valid RBAC databases? (Choose three.)

- ❍ **A.** `/etc/usr_attr`
- ❍ **B.** `/etc/user_attr`
- ❍ **C.** `/etc/security/exec_attr`
- ❍ **D.** `/etc/security/prof_attr`

9. You want to enable a user to administer all user `cron` tables. This includes amending entries in any user's `crontab`. Given due care to system security, what should you do to enable the user to carry out this duty?

- ❍ **A.** Give the user the root password.
- ❍ **B.** Set the `suid` on the `crontab` command.
- ❍ **C.** Use RBAC to authorize the user to administer `cron` tables.
- ❍ **D.** Use RBAC to give the user an ID of root when executing the `crontab` command.
- ❍ **E.** Use the ACL mechanism to give the user RW access to each `crontab` table.

10. Which command(s) grant a user access to a role account? (Choose two.)

- ❍ **A.** `roleadd`
- ❍ **B.** `rolemod`
- ❍ **C.** `useradd`
- ❍ **D.** `usermod`

11. Which option to the `rolemod` command appends an authorization to an exiting list of authorizations?

- ❍ **A.** `-A`
- ❍ **B.** `-P`
- ❍ **C.** `-a`
- ❍ **D.** `-o`
- ❍ **E.** None

12. In which files are profiles defined? Choose all that apply. (Choose two.)

- ❍ **A.** `/etc/security/prof_attr`
- ❍ **B.** `/etc/user_attr`
- ❍ **C.** `/etc/security/exec_attr`
- ❍ **D.** `/etc/security/auth_attr`

13. Which statements are true regarding the following line? (Choose all that apply.)

```
Media Restore:suser:cmd:::/usr/lib/fs/ufs/ufsrestore:euid=0
```

- ❍ **A.** It represents a profile in the `exec_attr` database.
- ❍ **B.** Any role that has Media Restore as a profile can execute the `ufsrestore` command with an effective UID of root.
- ❍ **C.** It represents a profile in the `prof_attr` database.
- ❍ **D.** It represents a role definition in the `user_attr` database.

14. In RBAC, which of the following is a bundling mechanism for grouping authorizations and commands with special attributes?

- ❍ **A.** Profile
- ❍ **B.** Role
- ❍ **C.** Authorization
- ❍ **D.** Group

Answers to Exam Questions

1. **C.** Use the `roleadd` command to create a role account. For more information, see the "Using RBAC" section.
2. **B.** `/etc/user_attr` contains details of the extended user attributes. For more information, see the "RBAC Components" section.
3. **A, B.** The `roleadd` command looks very similar to the `useradd` command, but it uses the profile shell as the default shell. For more information, see the "Using RBAC" section.
4. **A.** `user_attr` (extended user attributes database) associates users and roles with authorizations and profiles. For more information, see the "RBAC Components" section.
5. **D.** `exec_attr` (profile attributes database) defines the privileged operations assigned to a profile. For more information, see the "RBAC Components" section.
6. **D.** Six valid keys exist: `euid`, `uid`, `egid`, `gid`, `privs`, and `limitprivs`. For more information, see the "RBAC Components" section.
7. **A.** If a role is not associated with a correct profile, the only commands that can be executed within the role are the privileged commands that you have set up. Ordinary nonprivileged commands are unavailable. For more information, see the "RBAC Components" section.

8. **B, C, D.** The three valid RBAC databases are `/etc/user_attr`, `/etc/security/exec_attr`, and `/etc/security/prof_attr`. For more information, see the "RBAC Components" section.

9. **C.** To enable a user to administer all user `cron` tables, configure RBAC to authorize the user to administer `cron` tables. For more information, see the "Using RBAC" section.

10. **C, D.** Use the `roleadd` command to create a role account. Then, with the `usermod` command, assign the role to an existing user account using the `-R` option. If you are creating a new user account, use the `useradd` command with the `-R` option to assign the role to the new user account. For more information, see the "Using RBAC" section.

11. **E.** The `rolemod` command does not add to the existing authorizations; it replaces any existing authorization setting. For more information, see the "Using RBAC" section.

12. **A, C.** `/etc/security/prof_attr` (rights profile attributes database) defines profiles, lists the profile's assigned authorizations, and identifies the associated help file. `/etc/security/exec_attr` (profile attributes database) defines the privileged operations assigned to a profile. For more information, see the "RBAC Components" section.

13. **A, B.** The following entry in the `exec_attr` database represents a profile named Media Restore:

    ```
    Media Restore:suser:cmd:::/usr/lib/fs/ufs/ufsrestore:euid=0
    ```

    Any role that has Media Restore as a profile can execute the `ufsrestore` command with an effective UID of root. For more information, see the "RBAC Components" section.

14. **A.** Execution profiles are bundling mechanisms for grouping authorizations and commands with special attributes. For more information, see the "RBAC Components" section.

Suggested Reading and Resources

Solaris 10 Documentation CD: "Security Services" and "System Administration Guide: Advanced Administration" manuals.

http://docs.sun.com. Solaris 10 documentation set: "Security Services" and "System Administration Guide: Advanced Administration" books in the System Administration collection.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

Z