PREFACE

I hate getting my hair cut. It's not because I have problem hair or feel like I'm giving in to peer pressure, or because 10 bucks seems like a lot for the maintenance of my Marine Corps-style crew cut. It's because I dread the inevitable question: "So, what do you do?"

How can a UNIX sysadmin describe what he or she does to a non-IT person? Or, even more important, just what the heck *is* a UNIX sysadmin, anyway?

If I were a brain surgeon, there would be no confused looks. My line of work would be obvious: I'd fix brains. If I were a plumber, there'd be no further questions. Sure, I'd make water run faster. If I were a cop, there'd be no feigned understanding. Of course, I'd enforce the laws.

A UNIX sysadmin is all of those things and much more.

I'm a brain surgeon because a computer is, at a very basic level, a manmade attempt at simulating the functions of the human brain. When there's a problem with that manmade brain, it can manifest itself in any number of symptoms. Our first job is to detect the problem, hopefully before it causes permanent damage, and fix it before it gets worse. However, I say a sysadmin's job is much more difficult than a brain surgeon's. How would a brain surgeon like it if a new version of the human brain was released every 18 months?

I'm a plumber, too. You will rarely work on a UNIX computer that isn't connected to a network of some kind, just like your house plumbing. In fact, Sun's motto, "The Network is the Computer," is one of the truest statements about IT today. Sysadmins and plumbers are both responsible for making sure that everything gets where it's going as quickly as possible. But compared to being a sysadmin, a plumber's got it easy. After all, have you ever been in an office where every employee had his or her own personal bathroom? Don't even get me started on the need for taking their bathrooms on business trips with them!

I'm also a cop. (Okay, I'm not about to say that being a sysadmin is more dangerous than being a police officer, but it's definitely tougher than being a mall security guard.) One of the fastest-growing areas in the IT field is security. There's a fine line between locking something down and locking it up. We are required to protect resources while allowing proper access to them. Making something more complicated does not automatically make it more secure. We—both law enforcement personnel and sysadmins—have an obligation to make sure that we protect everyone equally. This means looking for and fixing areas of weakness that could be exploited, reviewing and refining our own procedures and catching law breakers.

We have written this guide based on the exam objectives published by Sun Microsystems for the Solaris 8 SCSA Exams, Parts I and II. This book was a group effort. The authors are all Sun Certified System Administrators and Solaris sysadmins with valuable experience. We think we've provided a unique, powerful, and reliable resource to help anyone who wants to learn.

It's what we do. We're UNIX sysadmins.

—Randy Cook, SCSA, MCSE
http://www.rcook.com

Our Objective

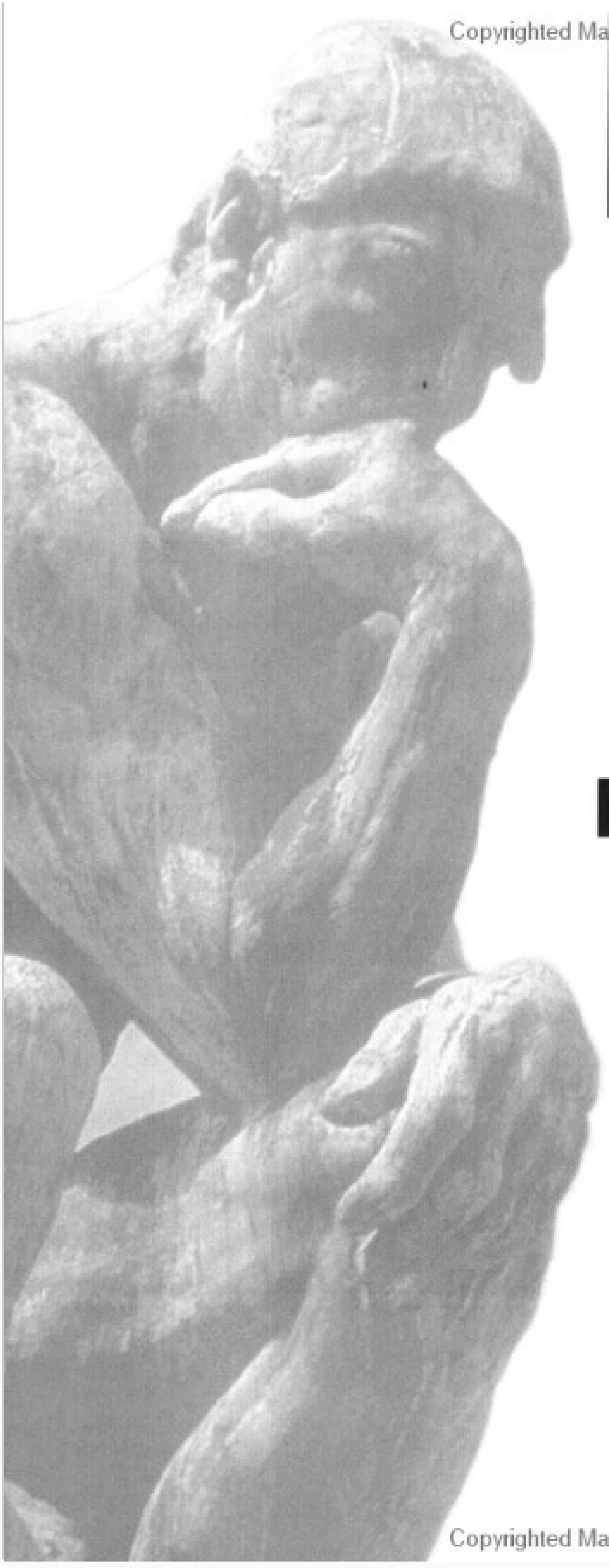
This book's primary objective is to help you prepare for the Sun Certified System Administrator for Solaris 8 Exams, Parts I and II.

At the time of this book's publication, all the exam objectives were posted on the Sun Web site and the beta exam process had been completed. Sun announced its commitment to measuring real-world skills. This book is designed with that premise in mind; its authors have practical experience in the field, using the Solaris 8 operating environment in hands-on situations, and have followed the development of the product since early beta versions.

Because the focus of the exams is on application and understanding, as opposed to memorization of facts, no book by itself can fully prepare you to obtain a passing score on these exams. It is essential that you work with the operating environment to enhance your proficiency. Toward that end, this book includes many practical, step-by-step exercises in each chapter that are designed to give you hands-on practice as well as guide you in truly learning the Solaris 8 operating environment, not just learning *about* it.

In This Book

This book is organized in such a way as to serve as an in-depth review for the Sun Certified System Administrator for Solaris 8 Exams, Parts I and II, for both



Solaris

SUN® CERTIFIED SYSTEM ADMINISTRATOR

1 Understanding System Concepts

CERTIFICATION OBJECTIVES

1.01	Defining system administration terms
1.02	Defining the effect of various man command options in viewing online manual pages
✓	Two-Minute Drill
Q&A	Self Test

In order to truly understand any machine, you must first understand its function and parts. Then you must understand how those parts work together in order to perform the machine's function.

In this chapter, we touch on all three topics—function, parts, and how the parts work together—in very general terms. The rest of the book goes into very specific detail, but first we need to define some common terms. This chapter covers the basic terminology in system administration and the use of the man command. This command is a starting point designed to give you the foundation on which the rest of the book builds.

CERTIFICATION OBJECTIVE 1.01

Defining System Administration Terms

In this section, we want to introduce you to the basic terminology that you'll need to understand in your role as system administrator.

Daemons

Many sysadmins are quick to point out that a *daemon* is not a *demon*. Even though it might sometimes seem like it, your server is not possessed by a devil. (Some of your users might be, but not your server.) We define a *daemon* as a program that runs in the background, disconected from a terminal or a particular login session. It is often used to manage system functions. The word literally means *divinity*. The poet William Butler Yeats had a fascination with the concept of daemons and their representation of the duality of existence. He described them as beings that represent the opposing sides of human nature. You can't know what good is with out evil's existence. How could you recognize a great piece of pie, for example, unless you've experienced a not-so-great one? Actually, the term *daemon* is derived from Greek mythology. The ancient Greeks called a supernatural being that acted as an intermediary between the gods and man a daemon. This is a very accurate description of what a UNIX daemon does for you. Basically, a daemon is always there, waiting to be called on to perform some action or service. There are daemons running on a system for every service the system provides. However, don't confuse the terms *daemon* and *process*.

A *process* is an instance of a running program. So, although a daemon is a process, a process isn't necessarily a daemon. You can see all the processes that are running on your system by using the ps command. You can use the ps command to display all the processes at once. These processes can comprise one big list, even on a little-used system. Naturally, with the use of command options, you can use the ps command to display only the information you're looking for. We cover the ps command in more detail in later chapters, but for now, let's take a look at some of the daemons that are running on your system right now.

EXERCISE 1-1

Viewing the Processes on a System

Let's first take a look at what is going on behind the scenes on our Solaris system. For this exercise, our system's hostname is *sol*.

1. Log on to the system:

```
Trying 192.168.1.10...
Connected to sol.
Escape character is '^]'.
SunOS 5.8
login:ra
password: **********
Last login: Tue Apr  3 19:12:13 from :0
Sun Microsystems Inc.   SunOS 5.8
Welcome to Sol - This system will be down for 4 hours this
Sunday for upgrades.  Call ext. 3649 if you have any
questions.
[sol: ra] $
```

2. List the processes.

Now we've logged in to *sol* as the user *ra* and seen a message from the very conscientious system administrator regarding a planned outage. Next, let's look at the processes, including the daemons, that are currently running on *sol* by using the following ps command.

```
[sol: ra] $ ps -ef  | more
     UID   PID  PPID  C    STIME TTY        TIME CMD
root      0      0  0   Mar 20 ?          0:03 sched
root    196      1  0   Mar 20 ?          0:16 /usr/sbin/nscd
root    206      1  0   Mar 20 ?          0:00 /usr/lib/lpsched
root    177      1  0   Mar 20 ?          0:01 /usr/sbin/syslogd
root    268      1  0   Mar 20 console  0:00 /usr/lib/saf/ttymon
-g -h -p cc9972-b console login:  -T sun-color -d /dev/cons
root    158      1  0   Mar 20 ?          0:01 /usr/sbin/inetd -s
root    157      1  0   Mar 20 ?          0:00 /usr/lib/nfs/lockd
daemon    160     1  0   Mar 20 ?          0:00 /usr/lib/nfs/statd
root    186      1  0   Mar 20 ?          0:01 /usr/sbin/cron
--More--
```

We used the ps command with the *e* and *f* options. This command displayed every process in a full listing. Since the list would easily cover more than one screen, we piped (|) the display into the more command. This way we can see one screen at a time, and by pressing the SPACEBAR, we see more of the display. We cover the ps command in more detail later. For now, let's go daemon hunting!

The output of the ps command includes the following line, which is highlighted in the preceding code:

```
root    206      1  0   Mar 20 ?          0:00 /usr/lib/lpsched
```

This is one of the daemons running on the system named *sol*. It's the printer spooling daemon, lpsched. This daemon waits for print requests, then sends the print job to the requested printer. lpsched is always running, waiting for a print job to come in.

exam Watch

Remember that most daemons end with the letter d. Most are started by initialization scripts at boot time, but this is configurable. Daemons can be stopped or restarted as needed. You often have to restart a daemon after making a configuration change to its service or, to fix a stuck print queue, you might have to stop and start the print spooler daemon.

Shells

A *shell* is the interface between the user and the kernel. It's the means by which you communicate commands. There are many, many kinds of shells. We focus on three of the most popular: Bourne, C, and Korn Shells.