

Introduction to Solaris 10

Operating systems are the building blocks of computer systems, and provide the interface between user applications and computer hardware. Solaris 10 is a multiuser, multitasking, multithreading operating environment, developed and sold by Sun Microsystems (<http://www.sun.com/>). Solaris is one implementation of the UNIX operating system that draws on both the System V (AT&T) and Berkeley (BSD) traditions. It has risen from little more than a research project to become the dominant UNIX operating system in the international marketplace today. Solaris 10 is the latest in a long line of operating environment releases based around the SunOS operating system, which is currently in version 5.10. Solaris is commonly found in large corporations and educational institutions that require concurrent, multiuser access on individual hosts and between hosts connected via the Internet. However, it is also rapidly being adopted by small businesses and individual developers, through Sun's promotion of the "Free Solaris" program (<http://www.sun.com/software/solaris/binaries/>). In this book, many of the references to the commands and procedures of Solaris 10 apply equally to earlier versions of Solaris 9, 8, 7, and 2.x.

Many desktop computer users have never heard of the word "Sun" in the context of computing, nor are they usually familiar with the term "Solaris" as an operating environment. However, almost every time that an Internet user sends an e-mail message or opens a file from a networked server running Sun's Network File System (NFS) product, Solaris is transparently supporting many of today's existing Internet applications. In the enterprise computing industry, Sun is synonymous with highly available and reliable high-performance hardware, while Solaris 10 is often the operating environment of choice to support database servers, message queues, XML Web services, and Java 2 Enterprise Edition (J2EE) application servers. Sun's hardware solutions are based around the UltraSPARC integrated circuit technologies, which currently support more than 100 processors in a single StarFire 15K server system. Sun systems are typically used to run financial databases, large-scale scientific computing environments, such as genetic sequencing, and complex graphics rendering required by movie studios in post-production.

In recent times, two of Sun's innovations have moved the spotlight from the server room to the desktop. First, Sun's development of the Java programming language,

which promises “write once, read anywhere” application execution across any platform that supports the Java Virtual Machine (JVM), has revolutionized the development of networked applications. Java “applets” now appear on many Web pages, being small, encapsulated applications that execute on the client side. J2EE application servers and their associated distributed component models (Enterprise Java Beans) power the back end of many *n*-tier applications, such as CRM, ERP, and HR systems.

Second, Sun is promoting a “free” version of Solaris 10 for the SPARC and Intel hardware platforms (<http://www.sun.com/software/solaris/binaries/>). Sun has also made Solaris 10 more accessible for desktop users, offering the OpenOffice productivity suite for a relatively small cost. OpenOffice is a product that is competitive to Microsoft Office—it contains word processing, spreadsheet, presentation, and database components that are fully integrated. In addition, OpenOffice runs on many different platforms, and in eight languages, meaning that a user on an UltraSPARC system can share documents seamlessly with users on Linux and Microsoft Windows. The combination of a solid operating system with a best-of-breed productivity suite has given Solaris new exposure in the desktop market.

This book is a “complete reference” for the Solaris 10 operating environment, and for the SunOS 5.10 operating system, meaning that I will try to cover, in detail, the operational aspects of Solaris and SunOS. If you simply need to look up a command’s options, you can usually make use of Sun’s own online “manual pages,” which you can access by typing **man *command***, where *command* is the command for which you require help. You can also retrieve the text of man pages and user manuals online by using the search facility at <http://docs.sun.com/>. This reference will be most useful when you need to implement a specific solution, and you need practical, tried-and-tested solutions. Although Solaris 10 comes with a set of tools for process management, for example, there may be others that improve productivity. Thus, while `ps` and `psig` are supplied with Solaris 10, `lsof` is not. In outlining a solution to a problem, we generally introduce Sun-supplied software first, and then discuss the installation and configuration of third-party alternatives. You can also use this book as a reference for previous versions of Solaris, since much of the command syntax remains unchanged across operating system releases. Command syntax is typically identical across different platforms as well (SPARC and Intel), except where hardware differences come into play, such as disk configuration and layout.

If you’ve been keeping track of recent press releases, you may be wondering why Solaris has a version number of 10, while SunOS has a revision level of 5.10. Since the release of Solaris 7 (SunOS 5.7), Sun has opted to number its releases sequentially with a single version number, based on the old minor revision number, coinciding with the introduction of 64-bit CPU architectures. This means that the release sequence for Solaris has been 2.5.1, 2.6, 7, 8, 9, and now 10. Sun provides “jumbo patches” for previous operating system releases, which should always be installed when released, to ensure that bugs (particularly security bugs) are resolved as soon as possible. Some changes between releases may appear cosmetic; for example, Larry Wall’s Perl interpreter has been included since the Solaris 8 distribution, meaning that a new generation of system

administrators will no longer have the pleasure of carrying out their first post-installation task. However, other quite important developments in the area of networking (such as IPv6) and administration (Sun Management Console tools) may not directly affect users, but are particularly important for enterprise administration.

In this chapter, we cover the background to the Solaris 10 operating environment, which really begins with the invention and widespread adoption of the UNIX operating system. In addition, we also cover the means by which Solaris 10 can run cross-platform applications—Sun’s development of Java can be seen as a strong commitment to cross-platform interoperability. In addition, Solaris 10 uses Samba to allow a Solaris server to act as a Windows NT or 2000 domain controller. Thus, if you want the reliability of SPARC hardware coupled with the widespread adoption of Microsoft Windows as a desktop operating system, Solaris 10 running Samba is an ideal solution.

Finally, we review some of the many sites on the Internet that provide useful information, software packages, and further reading on many of the topics that we cover in this book.

What Is UNIX?

UNIX is not easily defined, since it is an “ideal” operating system that has been instantiated by different vendors over the years, in some quite nonstandard ways. It is also the subject of litigation, as vendors fight over the underlying intellectual property in the system. However, there are a number of features of UNIX and UNIX-like systems (such as Linux) that can be readily described. UNIX systems have a core kernel, which is responsible for managing core system operations, such as logical devices for input/output (such as */dev/pty*, for pseudo-terminals), and allocating resources to carry out user-specified and system-requisite tasks. In addition, UNIX systems have a hierarchical file system that allows both relative and absolute file path naming, and is extremely flexible. UNIX file systems can be mounted locally, or remotely from a central file server. All operations on a UNIX system are carried out by processes, which may spawn child processes or other lightweight processes to perform discrete tasks. Processes can be uniquely identified by their process ID (PID).

Originally designed as a text-processing system, UNIX systems share many tools that manipulate and filter text in various ways. In addition, small, discrete utilities can be easily combined to form complete applications in rather sophisticated ways. These applications are executed from a user shell, which defines the user interface to the kernel. Although GUI environments can be constructed around the shell, they are not mandatory.

UNIX is multiprocess, multiuser, and multithreaded. This means that more than one user can execute a shell and applications concurrently, and that each user can execute applications concurrently from within a single shell. Each of these applications can then create and remove lightweight processes as required.

Because UNIX was created by active developers, rather than operating system gurus, there was always a strong focus on creating an operating system that suited

programmers' needs. A *Bell System Technical Journal* article ("The Unix shell," by S.R. Bourne, 1978) lists the key guiding principles of UNIX development:

- *Create small, self-contained programs that perform a single task.* When a new task needs to be solved, either create a new program that performs it, or combine tools from the toolset that already exists, to arrive at a solution. This is a similar orientation to the current trend toward encapsulation and independent component building (such as Enterprise Java Beans), where complicated systems are built from smaller, interacting but logically independent modules.
- *Programs should accept data from standard input and write to standard output; thus, programs can be "chained" to process each other's output sequentially.* Avoid interactive input in favor of command-line options that specify a program's actions to be performed. Presentation should be separated from what a program is trying to achieve. These ideas are consistent with the concept of piping, which is still fundamental to the operation of user shells. For example, the output of the `ls` command to list all files in a directory can be "piped" using the `|` symbol to a program such as `grep` to perform pattern matching. The number of pipes on a single command-line instruction is not limited.
- *Creating a new operating system or program should be undertaken on a scale of weeks not years: the creative spirit that leads to cohesive design and implementation should be exploited.* If software doesn't work, don't be afraid to build something better. This process of iterative revisions of programs has resurfaced in recent years with the rise of object-oriented development.
- *Make best use of all the tools available, rather than asking for more help.* The motivation behind UNIX is to construct an operating system that supports the kinds of toolsets required for successful development.

This is not intended to be an exhaustive list of the characteristics that define UNIX; however, these features are central to understanding the importance that UNIX developers often ascribe to the operating system. It is designed to be a programmer-friendly system.

The History of UNIX

UNIX was originally developed at Bell Laboratories as a private research project by a small group of people, starting in the late 1960s. This group had experience with research efforts on a number of different operating systems in the previous decade, and its goals with the UNIX project were to design an operating system to satisfy the objectives of transparency, simplicity, and modifiability, with the use of a new third-generation programming language. At the time of conception, typical vendor-specific operating systems were extremely large, and all written in assembly language, making them difficult to maintain. Although the first attempts to write the

UNIX kernel were based on assembly language, later versions were written in a high-level language called C, which was developed during the same era. Even today, most modern operating system kernels, such as the Linux kernel, are written in C. After the kernel was developed using the first C compiler, a complete operating environment was developed, including the many utilities associated with UNIX today (e.g., the visual editor, `vi`). In this section, we examine the timeline leading to the development of UNIX, and the origins of the two main “flavors” of UNIX: AT&T (System V) and BSD.

Origins of UNIX

In 1969, Ken Thompson from AT&T’s Bell Telephone Labs wrote the first version of the UNIX operating system, on a DEC PDP-7. Disillusioned with the inefficiency of the Multics (Multiplexed Information and Computing Service) project, Thompson decided to create a programmer-friendly operating system that limited the functions contained within the kernel and allowed greater flexibility in the design and implementation of applications. The PDP-7 was a modest system on which to build a new operating system—it had only an assembler and a loader, and it would allow only a single user login at any one time. It didn’t even have a hard disk—the developers were forced to partition physical memory into an operating system segment and a RAM disk segment. Thus, the first UNIX file system was emulated entirely in RAM!

After successfully crafting a single-user version of UNIX on the PDP-7, Thompson and his colleague Dennis Ritchie ported the system to a much larger DEC PDP-11/20 system in 1970. This project was funded with the requirement of building a text-processing system for patents, the descendants of which still exist in text filters such as `troff`. The need to create application programs ultimately led to the development of the first C compiler by Ritchie, which was based on the B language. C was written with portability in mind—thus, platform-specific libraries could be addressed using the same function call from source code that would also compile on another hardware platform. Although the PDP-11 was better than the PDP-7, it was still very modest compared to today’s scientific calculators—it had 24KB of addressable memory, with 12KB reserved for the operating system. By 1972, the number of worldwide UNIX installations had grown to ten.

The next major milestone in the development of UNIX was the rewriting of the kernel in C, by Ritchie and Thompson, in 1973. This explains why C and UNIX are strongly related—even today, most UNIX applications are written in C, even though other programming languages have long been made available. Following the development of the C kernel, the owners of UNIX (being AT&T) began licensing the source code to educational institutions within the United States and abroad. However, these licenses were often restrictive, and the releases were not widely advertised. No support was offered, and no mechanism was available for officially fixing bugs. However, because users had access to the source code, the ingenuity in

hacking code—whose legacy exists today in community projects like Linux—gathered steam, particularly in the University of California at Berkeley (UCB). The issue of licensing and AT&T’s control over UNIX would determine the future fragmentation of the operating system in years to come.

In 1975, the first distribution of UNIX software was made by the Berkeley group, and was known as the BSD. Berkeley was Ken Thompson’s alma mater, and he teamed up with two graduate students (Bill Joy and Chuck Haley) who were later to become leading figures in the UNIX world. They worked on a UNIX Pascal compiler that was released as part of BSD, and Bill Joy also wrote the first version of *vi*, the visual editor, which continues to be popular even today.

In 1978, the seventh edition of the operating system was released, and it supported many different hardware architectures, including the IBM 360, Interdata 8/32, and Interdata 7/32. The version 7 kernel was a mere 40KB in size, and included the following system calls: `_exit`, `access`, `acct`, `alarm`, `brk`, `chdir`, `chmod`, `chown`, `chroot`, `close`, `creat`, `dup`, `dup2`, `exec*`, `exit`, `fork`, `fstat`, `ftime`, `getegid`, `geteuid`, `getgid`, `getpid`, `getuid`, `gtty`, `indir`, `ioctl`, `kill`, `link`, `lock`, `lseek`, `mknod`, `mount`, `mpxcall`, `nice`, `open`, `pause`, `phys`, `pipe`, `pkoff`, `pkon`, `profil`, `ptrace`, `read`, `sbrk`, `setgid`, `setuid`, `signal`, `stat`, `stime`, `stty`, `sync`, `tell`, `time`, `times`, `umask`, `umount`, `unlink`, `utime`, `wait`, `write`. Indeed, the full manual for version 7 is now available online at <http://plan9.bell-labs.com/7thEdMan/index.html>.

With the worldwide popularity of UNIX version 7, AT&T began to realize that UNIX might be a valuable commercial product, and attempted to restrict the teaching of UNIX from source code in university courses, thereby protecting valuable intellectual property. In addition, AT&T began to charge license fees for access to the UNIX source for the first time. This prompted the UCB group to create its own variant of UNIX—the BSD distribution now contained a full operating system in addition to the traditional applications that originally formed the distribution. As a result, version 7 forms the basis for all the UNIX versions currently available. This version of UNIX also contained a full Brian Kernighan and Ritchie C compiler, and the Bourne shell. The branching of UNIX into AT&T and BSD “flavors” continues even today, although many commercial systems—such as SunOS, which is derived from BSD—have now adopted many System V features, as discussed in the upcoming section, “Features of System V Release 4.” Mac OS X is the latest UNIX system to be based around a BSD kernel.

The most influential BSD versions of UNIX were 4.2, released in 1983, and 4.3, released in 1987. The DARPA-sponsored development of the Internet was largely undertaken on BSD UNIX, and most of the early commercial vendors of UNIX used BSD UNIX rather than pay license fees to AT&T. Indeed, many hardware platforms even today, right up to Cray supercomputers, can still run BSD out of the box. Other responses to the commercialization of UNIX included Andrew Tanenbaum’s independent solution, which was to write a new UNIX-like operating system from scratch that would be compatible with UNIX, but without even one line of AT&T code. Tanenbaum called it Minix, and Minix is still taught in operating systems courses today. Minix was also to

play a crucial role in Linus Torvalds' experiments with his UNIX-like operating system, known today as Linux.

Bill Joy left Berkeley prior to the release of 4.2BSD, and modified the 4.1c system to form SunOS. In the meantime, AT&T continued with its commercial development of the UNIX platform. In 1983, AT&T released the first System V Release 1 (SVR1), which had worked its way up to Release 3 by 1987. This is the release that several of the older generation of mainframe hardware vendors, such as HP and IBM, based their HP-UX and AIX systems upon, respectively. At this time, Sun and AT&T also began planning a future merging of the BSD and System V distributions. In 1990, AT&T released System V Release 4, which formed the basis for the SunOS 5.x release in 1992—this differed substantially from the previous SunOS 4.x systems, which were entirely based on BSD. Other vendors, such as IBM and DEC, eschewed this new spirit of cooperation and formed the Open Software Foundation (OSF).

In recent years, a number of threats have emerged to the market dominance of UNIX systems: Microsoft's enterprise computing products and frameworks, such as Windows 2003, 2000, and NT servers, and the .NET Framework. Together, these are designed to deliver price-competitive alternatives to UNIX on inexpensive Intel hardware. In the same way that UNIX outgunned the dominant mainframe vendors with a faster, leaner operating system, Microsoft's strategy has also been based on arguments concerning total cost of ownership (TCO), and a worldwide support scheme for an enormous installed base of desktop Microsoft Windows clients. With the development of XML Web services, providing platform-independent transports, data descriptions, and message-based Remote Procedure Call (RPC), there has been a strong push to move toward common standards for system integration. Thus, integrating .NET components with J2EE EJBs can now be performed with a few mouse clicks.

The greatest threat to UNIX is the increasing popularity of Linux, for which different vendors sell distributions based on a "free" kernel. Initially, these companies provided distributions for free, in the spirit of the "free software" movement, and only charged for support and services. Nowadays, the reverse is true: Linux vendors charge for distributions, while the Solaris distribution is free (see <http://www.sun.com/software/solaris/binaries/> for details)!

UNIX will still have an important role to play in the future; however, as desktop computing systems rapidly become connected to the Internet, they will require the kinds of services typically available under Solaris 10. As part of their territorial defense of the UNIX environment, many former adversaries in the enterprise computing market, such as IBM, HP, and Sun, have agreed to work toward a Common Open Software Environment (COSE), which is designed to capitalize on the common features of UNIX provided by these vendors. By distributing common operating system elements such as the common desktop environment (CDE), based on X11, these vendors will be looking to streamline their competing application APIs, and to support emerging enterprise data-processing standards, such as the Object Management Group's CORBA object management service, and XML Web services.

Features of BSD

Solaris was originally derived from the BSD distribution from the University of California. Thus, commands in SunOS 4.x were very similar to those found in other BSD distributions, although these changed significantly in SunOS 5.x when System V Release 4 was adopted. For example, many veteran system administrators would still find themselves typing **ps aux** to display a process list, which is BSD style, rather than the newer **ps -eaf**, which is correct for SVR4. Before AT&T commercialized UNIX, the BSD distribution required elements of the AT&T system to form a fully operational system. By the early 1990s, the UCB groups had removed all dependencies on the AT&T system. This led to the development of many of the existing BSD systems available today, including FreeBSD and NetBSD.

The innovations pioneered at UCB included the development of a virtual memory system for UNIX, a fast file system (which supported long filenames and symbolic links), and the basic elements of a TCP/IP networking system (including authentication with Kerberos). The TCP/IP package included support for services such as Telnet and FTP, and the Sendmail mail transport agent, which used the Simple Mail Transfer Protocol (SMTP). In addition, alternate shells to the default Bourne shell—such as the C shell, which uses C-like constructs to process commands within an interpreted framework—were also first seen in the BSD distribution, as were extensions to process management, such as job control. Standard terminal-management libraries, such as `termcap` and `curses`, also originated with BSD. Products from other vendors were also introduced into BSD, including NFS clients and servers from Sun Microsystems. Later releases also included support for symmetric multiprocessing (SMP), thread management, and shared libraries.

It is often said that the BSD group gave rise to the community-oriented free software movement, which underlies many successful software projects being conducted around the world today. However, BSD is not the only attempt to develop a “free” version UNIX. In 1984, Richard Stallman started developing the GNU (GNU’s Not UNIX) system, which was intended to be a completely free replacement for UNIX. The GNU C and C++ compilers were some of the first to fully support industry standards (ANSI), and the GNU Bourne Again Shell (BASH) has many more features than the original Bourne shell. You can find more information about the GNU project at <http://www.gnu.org/>. In addition, several versions of BSD are still freely distributed and available, such as FreeBSD.

Features of System V Release 4

Solaris 10 integrates many features from the AT&T System V releases, including support for interprocess communication, which were missing in the BSD distributions. As discussed earlier, many legal battles were fought over the UNIX name and source. System V was developed by the UNIX System Laboratories (USL), which was still majority-owned by AT&T in the early 1980s. However, Novell bought USL in early 1993. Eventually, USL sold UNIX to Novell, which ultimately sold it to X/Open. In 1991, the OSF-1 specification was released, and although DEC is the only major manufacturer to fully implement the standard, there is much useful cross-fertilization

between System V and other operating systems. Since Sun joined OSF in 1994, there has been new hope of standardizing UNIX services and APIs across platforms.

The major contributions of System V to the UNIX platform are as follows:

- Enhancement of the Bourne shell, including shell functions
- The STREAMS and TLI networking libraries
- Remote file sharing (RFS)
- Improved memory paging
- The Application Binary Interface (ABI)

The major differences between SVR4 and BSD UNIX can be summarized as follows:

Feature	System V	BSD
Boot scripts	<i>/etc/init.d</i>	<i>/etc/rc.d</i>
Default shell	Bourne shell	C shell
File system mount database	<i>/etc/mnttab</i>	<i>/etc/mtab</i>
Kernel name	<i>/unix</i>	<i>/vmunix</i>
Printing system	<i>lp</i>	<i>lpr</i>
String functions	<i>memcpy</i>	<i>bcopy</i>
Terminal initialization	<i>/etc/inittab</i>	<i>/etc/ttys</i>
Terminal control	<i>termio</i>	<i>termios</i>

The Solaris Advantage

Sun Microsystems was formed by former graduate students from Stanford and Berkeley, who used Stanford hardware and Berkeley software to develop the workstation market in the enterprise. Sun aimed to compete directly with the mainframe vendors by offering CPU speed and a mature operating system on the desktop, which was unprecedented. For a given price, greater performance could be obtained from the Sun workstations than was ever possible using mainframes. From one perspective, this success destroyed the traditional client/server market, which used very dumb terminals to communicate with very clever but horrendously expensive mainframe systems. The vendors of some proprietary systems, such as HP and DEC, saw their market share rapidly decline in the enterprise market because Sun delivered more “bang per buck” in performance. By 1986, UNIX was the dominant force at the expense of operating systems like VAX/VMS, although VMS would later come back to haunt UNIX installations in the form of Windows NT. When users could have a workstation with graphics instead of a dumb terminal, there were few arguments about adopting Sun.

However, Sun’s innovation enabled departments and workgroups to take control of their own computing environments and to develop productively with the C programming

language. Sun took BSD and transformed it into a commercial product, adding some useful innovations, such as NFS, along the way. This was similar in some ways to the approach of Linux companies that create distributions of useful software packages and bundle them with the Linux kernel. However, one significant difference between Sun and Red Hat Linux is that Sun has always been a company with a hardware focus—its systems were designed with the SPARC chipset, and more recently the UltraSPARC chipset, in mind. This has enabled Sun to create very fast workstations and servers with typically lower CPU speeds than Intel, but faster and more efficient bus performance. Sun invests heavily in hardware design and implementation for an expected commercial reward, all the more so now that Sun gives away the Solaris 10 operating system.

The major innovations of SunOS 4.x can be summarized as follows:

- Implementation of the network file system (NFS version 2.0, running over UDP)
- The OpenWindows 2.0 graphical user environment, based on X11
- The OpenBoot monitor
- The DeskSet utilities
- Multiprocessing support

The major innovations of SunOS 5.x can be summarized as follows:

- Support for SMP of more than 100 processors in a single server
- The OpenWindows graphical user environment and OpenLook
- Integration with MIT X11R5, Motif, PostScript, and the CDE
- Support for Gnome 2.0 as an alternative desktop, enhancing Linux integration
- The Network Information Service (NIS/NIS+)
- Kerberos integration for authentication
- Support for static and dynamic linking
- Full-moon clustering and N1 grid containers, ensuring high availability
- The ability to serve Windows 2003, 2000, and NT clients as a domain controller
- Tooltalk
- Java
- POSIX-compliant development environment, including single threads, multithreading, shared memory, and semaphores
- Real-time kernel processing
- X/OPEN-compliant command environment
- Compliance with UNIX 95 and UNIX 98 standards
- Support for very large (>2GB) files

- Microsoft Windows emulation on the desktop with Windows Application Binary Interface (WABI)
- Advanced volume management (`vold`)
- Standardized package administration and deployment tools
- Standardized patch management and integration
- Software-based power management
- Access control lists for resource authorization
- Support for centralized management of user home directories using the automounter
- Improvements to NFS (version 4), running over TCP
- Support for advanced networking, such as IPv6, ATM, frame relay, and Gigabit Ethernet
- JumpStart customization of local site installation and deployment
- 64-bit kernel architecture with Solaris 7 and later
- Simplified backup and restore procedures
- Simplified site administration with the AdminSuite toolkit and now the Solaris Management Console

Hardware Support (SPARC and x86)

The original CPU for Sun systems was the SPARC chip, with processor speeds of around 40–60 MHz. However, current systems use the UltraSPARC chipset, with processor speeds in the GHz range. The bus architectures of Sun systems are typically much faster than their PC counterparts, more than making up for their sometimes slower chip speeds.

With the introduction of Solaris 2.1 came support for the Intel platform, supporting ISA, EISA, MCA, and PCI bus types. Version 2.1 performed adequately on high-end 486 systems. Given the significant variation in types and manufacturers of PC hardware, not all devices are currently supported under Solaris 10, although newer innovations, such as the Universal Serial Bus (USB), are supported. Solaris 10 for Intel runs very fast on modern Pentium-IV systems, meaning that Intel devotees now have a wider choice of operating system, if they don't want to buy Sun hardware. There was also a single port of Solaris to the PowerPC platform (with version 2.5.1); however, this failed to impress MacOS users, and was deprecated in Solaris 2.6.

Solaris for Intel users will require the Hardware Compatibility List (HCL) to determine whether their particular system or their peripheral devices are supported. You can find this list at <http://access1.sun.com/drivers/hcl/hcl.html>. The HCL lists all tested systems, components, and peripherals that are known to work with Solaris for Intel. Chances are, if your hardware is not listed, it won't be supported. However, many Intel-based standards have been adopted by Sun, including the PCI bus, which is now integrated in the desktop Ultra workstations.

Cross-Platform Interoperability

The main focus of interoperability of different operating systems lies with the Java programming language, developed by Sun. Starting life as the “Oak” project, Java promises a “write once, run anywhere” platform, which means that an application compiled on Windows NT, for example, can be copied to Solaris 10 and executed without modification and without recompilation. Even in the 1970s, when C was being implemented far and wide across different hardware platforms, it was often possible to transfer source and recompile it without modification, but binary compatibility was never achieved. The secret to Java’s success is the two-stage compile and interpretation process, which differs from many other development environments. Java source is compiled on the source platform to an intermediary bytecode format, which can then be transferred to any other platform and interpreted by a JVM. Many software vendors, including SunSoft and Microsoft, have declared support for the Java platform, even though some vendors have failed to meet the specifications laid out by Sun. Until a standard is developed for Java, Sun will retain control over its direction, which is a risk for non-Solaris sites especially. However, organizations with Solaris 10 installations should have few qualms about integrating Java technology within their existing environments. With the release of free development tools, such as Borland’s JBuilder Foundation (<http://www.borland.com/>), development in Java is becoming easier for C and experienced UNIX developers. Java is the best attempt yet at complete binary compatibility between operating systems and architectures. More recently, XML Web services have emerged as an important set of technologies for system integration, and these are generally supported by Java.

Recent Solaris Innovations

Recent Solaris releases have contained many enhancements and new features compared to earlier versions, on both the client and server side, and specifically for administrators. For example, security has been overhauled with the inclusion of Kerberos version 5, and IPSec for both IPv4 and IPv6. This security makes it easy to create virtual private networks (VPNs) through improved tunneling and encryption technologies. In the next section, we review some of the recent Solaris innovations.

Server Tools

Many Solaris tools are targeted at implementing nonfunctional requirements, such as scalability, availability, security, integrity, and manageability. For example, Sun’s multiprocessing systems are highly available because of their hot-swapping capabilities, meaning that components can be replaced while the system is running—no need for a reboot! The systems are also highly scalable, because the CPU capacity of many systems can be combined using Sun’s Cluster product, which offers high system availability through management of hardware redundancy.

Clustering

Increased performance is often gained by the use of hardware redundancy, which can be achieved on a file system–by–file system basis, by using a software solution, such as volume management, or a hardware-based solution, such as a Redundant Array of Independent Disks (RAID) appliance. This allows partitions to be actively mirrored, so that in the event of a hardware failure, you can rapidly resume service and restore missing data.

This approach is fine for single-server systems that do not require close to 100 percent uptime. However, for mission-critical applications, where the integrity of the whole server is at stake, it makes sense to invest in clustering technology. Quite simply, clusters are what the name suggests: groups of similar servers (or “nodes”) that have similar function, and that share responsibility for providing system and application services. Clustering is commonly found in the financial world, where downtime is measured in hundreds of thousands of dollars, and not in minutes. Large organizations need to undertake a cost-benefit analysis to determine whether clustering is an effective technology for their needs. However, Sun has made the transition to clustering easier by integrating the Cluster product with Solaris 10, featuring a clustered virtual file system and cluster-wide load balancing.

Grids, Zones, and Resource Management

Clustering is generally restricted to a set of co-located servers with similar capabilities. The real innovation in Solaris 10 is the introduction of N1 grid containers, providing a framework for logically partitioning a single system. Advanced resource management features ensure that applications can be run in separate zones, with logical isolation ensuring true encapsulation. Applications can be allocated CPU capacity on demand, reducing overall waste.

The Resource Manager extends a number of existing tools that provide for monitoring and allocation of system resources to various tasks and services. This is particularly useful in high-end systems, where a large pool of resources can be allocated to specific processes. Although the existing `nice` command allows priorities to be set on specific processes, and `top` displays the resources used by each process, the Resource Manager is an integrated toolkit, featuring a scheduler, accounting, and billing. Again, although accounting tools are supplied as part of the standard Solaris toolkit, they have never been integrated with useful real-time monitoring tools. The Resource Manager also features a command-line interface and optional GUI for configuring and monitoring resource allocation and usage.

Volume Management

Although software RAID support has been previously provided in Solaris through Solstice Disk Suite (SDS), this product has now been superseded by Volume Manager (VM). VM supports RAID levels 0, 1, and 5, and allows a wide range of mirroring and striping facilities. Cross-grade and migration tools are also available to assist SDS users who are currently using metadevices as their primary virtual file systems for boot and nonboot disks and their associated slices.

Live Upgrade

Live Upgrade allows a Solaris system to continue running while components of its operating system are upgraded. This is particularly useful in production environments, where system downtime costs money and customers, particularly on shared platforms, like the StarFire 15K. A separate boot environment is constructed during run time, after which the system is rebooted with the new configuration, thereby minimizing downtime.

System Management

Solaris Management Tools 2.1 is the only supported GUI management environment for Solaris 10—`admintool`, AdminSuite 2.3 and 3.0, and Solaris Management Tools 1.0 and 2.1 have all been deprecated. For console-based system administration, new command sets are integrated into the Solaris Management Console command set. This makes system management tasks easier because there is now a single GUI or console interface. The Solaris Management Console Tools Suite includes the following tools:

- Computers and Networks Tool
- Diskless Client Support
- Disks Tool
- Enhanced Disk Tool (Solaris Volume Manager)
- Job Scheduler
- Log Viewer
- Mail Alias Support
- Mounts and Shares Tool
- Name Service Support
- Patch Tool
- Performance Tool
- Projects Tool
- RBAC Support
- RBAC Tool
- Serial Port Tool
- Software Package Tool
- System Information Tool
- User and Group Tool

Security Innovations

Security is a major concern for Solaris administrators. The Internet is rapidly expanding, with the new IPv6 protocol set to completely supercede IPv4 sometime in the next few years. This will make many more addresses available for Internet hosts than are currently available. It also means that the number of crackers, thieves, and rogue users will also

increase exponentially. Solaris 10 prepares your network for this “virtual onslaught” by embracing IPv6, not only for its autoconfiguration and network numbering features, but also because of the built-in security measures that form part of the protocol. In particular, authentication is a key issue, after many highly publicized IP-spoofing breaches reported in the popular press over the past few years. A second layer of authentication for internal networks and intranets is provided in Solaris 10 by the provision of Kerberos version 5 clients and daemons.

Kerberos Version 5

Kerberos is the primary means of network authentication employed by many organizations to centralize authentication services. As a protocol, it is designed to provide strong authentication for client/server applications by using secret-key cryptography. Recall that Kerberos is designed to provide authentication to hosts inside and outside a firewall, as long as the appropriate realms have been created. The protocol requires a certificate granting and validation system based around “tickets,” which are distributed between clients and the server. A connection request from a client to a server takes a convoluted but secure route from a centralized authentication server, before being forwarded to the target server. This ticket authorizes the client to request a specific service from a specific host, generally for a specific time period. A common analogy is a parking ticket machine that grants the drivers of motor vehicles permission to park in a specific street for one or two hours only.

Kerberos version 5 contains many enhancements, including ticket renewal, removing some of the overhead involved in repetitive network requests. In addition, there is a pluggable authentication module, featuring support for RPC. The new version of Kerberos also provides both server- and user-level authentication, featuring a role-based access-control feature that assigns access rights and permissions more stringently, ensuring system integrity. In addition to advances on the software front, Solaris 10 also provides integrated support for Kerberos and smart card technology using the Open Card Framework (OCF) 1.1. More information concerning Kerberos is available from MIT at <http://web.mit.edu/network/kerberos-form.html>.

IPv6 and IPsec

IPv6, described in RFC 2471, is the replacement IP protocol for IPv4, which is currently deployed worldwide. The Internet relies on IP for negotiating many transport-related transactions on the Internet, including routing and the Domain Name Service. This means that host information is often stored locally (and inefficiently) at each network node. It is clearly important to establish a protocol that is more general in function, but more centralized for administration, and that can deal with the expanding requirements of the Internet.

One of the growing areas of the Internet is obviously the number of hosts that need to be addressed; many subnets are already exhausted, and the situation is likely to get worse. In addition, every IP address needs to be manually allocated to each individual machine on the Internet, which makes the usage of addresses within a subnet sparse and less than optimal. Clearly, there is a need for a degree of centralization when

organizing IP addresses that can be handled through local administration, and through protocols like Dynamic Host Configuration Protocol (DHCP). However, one of the key improvements of IPv6 over IPv4 is its autoconfiguration capabilities, which make it easier to configure entire subnets and to renumber existing hosts. In addition, security is now included at the IP level, making host-to-host authentication more efficient and reliable, even allowing for data encryption.

One way security is achieved is by authentication header extensions: this allows a target host to determine whether a packet actually originates from a source host. This prevents common attacks, such as IP spoofing and denial of service (DoS), and reduces reliance on a third-party firewall by locking in security at the packet level. Tools are also included with Solaris 10 to assist with IPv4 to IPv6 migration.

VPN technology is also provided with Solaris 10, using IPSec. IPSec is compatible with both IPv4 and IPv6, making it easier to connect hosts using both new and existing networking protocols. IPSec consists of a combination of IP tunneling and encryption technologies to create sessions across the Internet that are as secure as possible. IP tunneling makes it difficult for unauthorized users (such as intruders) to access data being transmitted between two hosts on different sites. This is supported by encryption technologies, and an improved method for exchanging keys, using the Internet key exchange (IKE) method. IKE facilitates inter-protocol negotiation and selection during host-to-host transactions, ensuring data integrity. Implementing encryption at the IP layer will make it even more difficult for rogue users to “pretend” to be a target host, intercepting data with authorization.

What's New in Solaris 10

Each new release of Solaris brings about changes at the client, server, and system levels. These changes affect users, administrators, and developers in different ways. The following features have been released for the first time with Solaris 10:

- N1 containers, allowing systems to be logically partitioned into zones with specific functions. Containers can be “booted” within a few seconds, ensuring high availability.
- Resource management changes, ensuring that specific limits can be set on resource usage by applications, preventing “runaway” applications from bringing a system to its knees.
- Integrated firewall technology, not requiring a separate install.
- Support for smart card authentication.
- Kernel instrumentation through dynamic tracing, allowing system fine-tuning and problem identification.
- Binary compatibility between different Solaris versions and Linux, and source compatibility between different Solaris platforms.
- Failure prediction of hardware components, ensuring that they can be replaced before impacting on system performance.

Sources for Additional Information

In this chapter, we have so far examined the history of UNIX, and what distinguishes UNIX systems from other operating systems. We have also traced the integration of both “flavors” of UNIX into the current Solaris 10 release. With the ever-rising popularity of Solaris 10, there are many Web sites, mailing lists, and documentation sets that new and experienced users will find useful when trying to capitalize on an investment in Sun equipment or the latest Solaris 10 operating environment. In this section, we present some pointers to the main Internet sites on which you can find reliable information about Solaris 10.

Sun Documentation/Sun Sites

Unlike some operating systems, Solaris 10 comes with a complete set of online reference manuals and user guides on the Documentation CD-ROM, which is distributed with all Solaris 10 releases (Intel and SPARC). The manuals, which are in PDF format and cover a wide range of system administration topics, include the following:

- *System Administration Guide: Basic Administration*
- *System Administration Guide: Advanced Administration*
- *System Administration Guide: Devices and File Systems*
- *System Administration Guide: IP Services*
- *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*
- *System Administration Guide: Naming and Directory Services (NIS+)*
- *System Administration Guide: Network Services*
- *System Administration Guide: Security Services*

The best thing about the manuals is that they are available for download and interactive searching through <http://docs.sun.com/>. This means that if you are working in the field and need to consult a guide, you don’t need to carry around a CD-ROM or a printed manual. Just connect through the Internet and read the guide in HTML, or download and retrieve a PDF format chapter or two.

The two main Sun sites for Solaris 10 are at <http://www.sun.com/solaris> (for SPARC users) and <http://www.sun.com/intel> (for Intel users). Both of these pages contain internal and external links that are useful in finding out more information about Solaris 10 and any current offerings. The Sun Developer Connection (<http://developers.sun.com/>) is a useful resource that users can join to obtain special pricing and to download many software components for free.

Web Sites

Many third-party Web sites are also available that deal exclusively with Sun and Solaris 10. For example, if you are looking for a Solaris 10 FAQ, or pointer to Sun information, try the Sun Help site (<http://www.sunhelp.org/>). If it's free, precompiled software that you're after, check the Sun Freeware site (<http://www.sunfreeware.com/>) or one of the many mirrors. Here you can find the GNU C compiler in a precompiled package. For Solaris for Intel users, there is also an archive of precompiled binaries available at <ftp://x86.cs.duke.edu/pub/solaris-x86/bins/>.

In case you are interested in seeing what the pioneers of UNIX are doing these days, check out the home pages of these famous UNIX developers:

- Brian Kernighan <http://cm.bell-labs.com/cm/cs/who/bwk/index.html>
- Dennis Ritchie <http://cm.bell-labs.com/cm/cs/who/dmr/index.html>
- Ken Thompson <http://cm.bell-labs.com/who/ken/>

USENET

USENET is a great resource for asking questions, finding answers, and contributing your skills and expertise to help others in need. This is not necessarily a selfless act—there will always be a Solaris 10 question that you can't answer, and if you've helped others before, they will remember you. The **comp.unix.solaris** forum is the best USENET group for Solaris 10 information and discussion. The best source of practical Solaris 10 information is contained in the Solaris FAQ, maintained by the legendary Casper Dik. You can always find the latest version at <http://www.wins.uva.nl/pub/solaris/solaris2/>. For Solaris for Intel users, there is the less formal **alt.solaris.x86** forum, where you won't be flamed for asking questions about dual-booting with Microsoft Windows, or mentioning non-SPARC hardware. For Solaris Intel, the best FAQ is at <http://sun.pmbc.com/faq/>. For both SPARC and Intel platforms, there is a **comp.sys.sun.admin** group that deals with system administration issues, which also has a FAQ available at <ftp://thor.ece.uc.edu/pub/sun-faq/FAQs>.

Mailing Lists

Mailing lists are a good way of meeting colleagues and engaging in discussions in a threaded format. The Sun Manager's List is the most famous Sun list, and contains questions, answers, and, most importantly, summaries of previous queries. All Solaris-related topics are covered. Details are available at <ftp://ftp.cs.toronto.edu/pub/jdd/sun-managers/faq>. In addition, there is a Solaris for x86 mailing list archived at <http://www.egroups.com/group/solarisonintel/>, which has some great tips, tricks, and advice for those who are new to Solaris 10, or who are having difficulties with specific hardware configurations.

Summary

Solaris 10 is an exciting, innovative operating environment. It can provide more functionality than existing desktop operating systems; however, there is an increased administrative overhead that you must consider. In this book, we hope to convey sound management practices and divulge practical techniques for solving many Solaris-related problems, and to implement the best-of-breed methods for all enterprise-level installations. By the end of this book, you should feel confident in managing all aspects of Solaris 10 system administration, and feel confident in transferring those skills to the management of related operating systems, such as Linux.

How to Find Out More

The main site for all Sun technologies in <http://www.sun.com/>. For further information on Java technologies, users should browse Sun's Java site at <http://java.sun.com/>.