Building a Secure Backup Server for the Solaris 9 Operating Environment

Shaun McAdams
August 21, 2003

GCUX Practical v1.9
Administrivia v2.6

Abstract

This paper will present an attempt to build a secure Veritas NetBackup 4.5 server running in the Solaris 9 Operating Environment. Solaris 9 was chosen because it is a widely used commercial operating system and is familiar to many system administrators. There are documented approaches to improving the security of a Solaris 9 server, which we will employ. Veritas NetBackup is a commercial client-server backup system that is both powerful and easy to use once installed. However, it suffers from a number of security failings that we will do our best to overcome. In addition to making selected operating system configuration changes, IP Filter, AIDE, SSH and TCP Wrappers will be used to improve security.

This paper is submitted in partial fulfillment of the GCUX certification requirements under Practical assignment v1.9, Option 1, Administrivia 2.6.

Table of Contents

Introduction

Any well-written procedure on securing a critical server will highlight the importance of making regular backups, both for disaster recovery and as a fallback position when files are deleted or modified by a malicious or careless individual (Garfinkel, p. 197;Pomeranz[1], p. 27;Parmar, p. 68). However, there are few guides to building a backup server in a secure way (Christensen). This may be because backups are not seen as an exciting part of a system administrator's job. It may be because backup servers are usually shielded from direct access to the Internet, or even "interior" company networks, by firewalls and private networking. Or it may be because many backup solutions are inherently insecure.

Such reasons aside, an insecure central backup server fundamentally undermines an otherwise secure network. A compromised backup server not only allows for the theft of data from any of the clients it has backed up and for the potential destruction of those important backups. It may also provide a conduit to compromising other computers due to the trust client machines often give backup hosts.

Description of System

The system we will build is a central backup server for approximately 100 clients located in a software development facility. These backup clients act as servers for a number of different development projects and are dispersed throughout many rooms and floors of a large building. The backup clients are addressed on multiple subnets, and many of them have only a single ethernet interface. This layout precludes the use of a dedicated private network for backups, which would significantly improve both our backup performance and our security options. The server and all clients will be located inside a firewall to prevent external access.

The hardware we will use consists of a Sun Microsystems Sunfire 280R rack mounted server with dual 1.2 GHz UltraSPARC-III+ processors, 4.0 GB of RAM, two 72 GB SCSI disks, a DVD/CDROM drive and dual power supplies. Additionally, we have installed a Low Voltage Differential (LVD) SCSI card in one PCI slot. Our backups will be written to LTO tapes housed in a StorageTek L80 robotic library equipped with four Seagate LTO Ultrium tape drives. The tape library can hold 80 LTO tapes for approximately 16 terabytes of storage and is connected via LVD SCSI to the server.

Solaris is one of the most widespread commercial Unix operating systems (Sun[1]). Because of this market share advantage, management feels comfortable buying Solaris based systems, system administrators are familiar with the configuration of those systems, and security experts have developed recipes for Solaris hardening. In the environment where our backup server is to be

deployed, Solaris is the most common OS for other infrastructure servers and for the clients to be backed up.

Solaris 9 is the latest major release from Sun Microsystems and features a number of improvements over earlier releases. Of specific interest are the enhancements to security that include (Sun[2]):

- integrated SSH server and client software
- the presence of a native /dev/random and /dev/urandom to ease the generation of encryption keys
- enhanced Pluggable Authentication Module (PAM) support
- included SunScreen firewall software
- integrated TCP Wrappers

Using Solaris on the backup server will allow us to leverage previous Solaris knowledge to streamline the installation and configuration process. For these reasons, Solaris 9 has been selected as the operating system for our server.

The backup software we will use is Veritas NetBackup DataCenter 4.5 Feature Pack 3. NetBackup is a leading backup and restore solution that features support for a large number of operating systems and media types, database options, and scalability for large enterprises. This flexibility and scalability come at a price – NetBackup requires numerous network connections and gives little thought to security.

Risk Analysis

While our backup server (and all its clients) will be protected by a firewall from external Internet access, we will still attempt to harden it. As part of a "defense in depth" strategy we will not assume that our firewall is perfect. We will prepare for the eventuality that some malicious person may gain a foothold on the internal network and from there initiate a network attack against our server. To this end we will limit active listening ports to our remote connection protocol (SSH) and the NetBackup software.

Since we will have an IP address on a general purpose network instead of a private backup network, a large number of employees will have visibility to the backup server. One of them may be an industrial spy, disgruntled over their treatment, or just have too much time on their hands and want to try out an exploit they have learned about. We will trust no one on the internal network any more than we have to.

A final network attack vector may come from the backup client machines themselves. The backup server will be regularly exchanging volumes of information with these clients that could mask network probes or exploit attempts.

Additionally, trusted or senior programmers on a given project may have been given root access on the servers we will be backing up. This is a questionable security practice in itself, but is outside the scope of this paper. A user with root access on the client could insert malicious code into the backup client software or otherwise take advantage of the fact that the server will be contacting the client at scheduled intervals.

Another class of attack to be concerned with is local privilege escalation. We will attempt to mitigate this risk by allowing only system administrators responsible for backups and security to log into this machine. Such persons will already have undergone extensive background checks and will have earned a position of trust. They will also already have privileged access. Nonetheless, we will use local intrusion detection and auditing to check for unexpected changes in the system state and to provide another layer of security.

We must also be concerned about physical security. While the physical security of any critical server is fundamentally important, it is especially so when the complete images of a hundred other machines are ripe for the picking merely by opening a tape library door and walking off with them. The server and the tape library must be physically secured to avoid tape theft, disk removal, console access, and the myriad of denials of service that can be propagated when you can actually lay hands on a machine.

Finally, we must be cognizant that Information Assurance is not only defending against malicious persons. The availability and integrity of the information that our backup server will hold are as important as its confidentiality. Care must be taken to minimize the likelihood of data loss due to failed hardware and the likelihood of hardware loss (and associated down time) due to electrical spikes, fire and water damage, overheating and other environmental factors. And some thought should be given to rapid and effective recovery methods to be implemented when such unfortunate events do occur.

Building the Server

The steps below have been developed from the author's personal experience, learned from other system administrators, and gained from reading a number excellent resources on hardening servers (Garfinkel, Noordergraaf, Pomeranz[2]). It is hoped that they will be effective, easy to understand and easy to reproduce.

Initial Setup

We will begin by rack mounting our 280R server (the Low Voltage Differential SCSI card is assumed already to have been installed) in a rack with locking front and rear doors. The tape library will be mounted in the same locking rack. The server's two power supplies will be serviced by feeds from separate uninterruptible power supplies (UPS). The primary ethernet interface (eri0) will

be connected via Category 5 cable to a standalone hub. No other computers will be connected to the hub at this time, so we have an isolated, “air gap” network. This setup will allow us to build the server without fear of a network attack against an unsecured OS. It will also avoid the annoying *link down* messages caused by having no network connection. Later we will have the opportunity to connect other computers to this isolated network to test our hardening from the outside.

Communication with the 280R will be by means of a DEC VT220 terminal attached to serial port A. The advantage to using a “dumb terminal” as our interface is that there is no chance the machine we are connecting to our console port can have been compromised previously. We can be sure that no one will SSH to our terminal machine and use our serial connection to access the new server, and that no one can capture our keystrokes as we configure the root password or other sensitive information. These are among the risks faced if you use a networked or “smart” host to connect to the new server’s console port. Minor disadvantages to this approach include the use of a keyboard that may lack certain function keys and the inability to make screen captures for documenting the process.

All components for our system are located inside a server room with redundant air handling, UPS support, and combination-locked doors. The server room is located in a building that is equipped with card readers for entry access, monitored video cameras, and 24-hour security guard staffing.

Operating System Install

For all subsequent steps it is assumed that the reader has a functional familiarity with the basics of Unix configuration, the Bourne shell and the **vi** editor. Information output by the computer will be denoted by use of a `fixed-width font`. Information to be input will be shown as **`fixed-width bold`**. Variable input that is a placeholder for an actual value will be shown in ***`fixed-width bold italics`***.

In addition to the Solaris 9 and the Veritas NetBackup installation media and a knowledge of the local network topology, you will also need a CDROM containing operating system patches, SCSI card drivers and third party tools such as IP Filter, AIDE, and fix-modes. If you wish to have this CD ready before beginning the OS install, skip ahead to the section titled “Additional Software and Patches” for a complete listing of software needed and sources for it.

Sun Microsystems generally releases an updated version of its current OS several times a year. You should make an attempt to use the latest release available. In this particular case we will be using the May 2002 release of Solaris 9.

1. Begin by powering on the server. Turn the key to the on position (12 noon) and press the power button to the right. While the computer is working its way through the LOM initialization sequence, you will have time to open the CDROM drive and insert the "Solaris 9 Software 1 of 2" CD. Once the OpenBoot banner has appeared and the memory test has begun you may send a "break" to bring the system to the OK prompt. (The precise method of sending a break on your console will vary from terminal to terminal.) We will issue the following commands:

```
{0} ok setenv security-mode command
security-mode =          command
{0} ok setenv security-password somepasswd
security-password =
```

This has the effect of requiring knowledge of the eeprom password to change any eeprom settings at the OK prompt or to boot from other than the default location (usually an internal hard disk.) Booting from a CDROM is an easy way to gain root access on a system, and this will make such an action harder. While this machine will be inside a locked cabinet and inside a locked room, we still take this step for the additional security it adds for very little effort and to keep this server's configuration consistent with other servers and workstations (which are at greater risk) in our environment. Because even the case of the 280R requires a key to open, using other physical contact methods to gain access to private information will likely be more destructive, and thus more noticeable.

Note that *somepasswd*, above, is merely a variable, and you should use an easy to remember password of your own. The restrictions on selecting a hard to crack password may be slightly relaxed for an eeprom password. Manually typing in guesses at the OK prompt is slow enough to make it untenable, and if you are root on the system, you may set the eeprom password without knowledge of the current value. WARNING: If you lose both the eeprom password and the root password on a system, recovery can be time consuming.

2. Type **`boot cdrom`** at the OK prompt. If you completed Step 1 above, you will be prompted for your password at this time.

3. The next three screens will ask you for information on how to present the installer menus. The first option you will be presented with is a choice of languages in which to proceed. I chose "0" for English.

4. Next you will select a locale. Again, "0" for English (C – 7-bit ASCII) was the right choice for this environment.

5. Now you will be asked to choose your terminal type. If the terminal you are using does not appear in the list, option “3”, the DEC VT100, is usually a safe choice.

6. The following screen describes the Solaris installation program and options for making selections if you don’t have function keys. If you have function keys on your keyboard, you may hit F2 to continue. Otherwise, follow the instructions for finding alternate key sequences.

7. We will now begin configuring the identity of our server on the Network Connectivity screen. Using the arrow keys to move the cursor, move to the “Yes” checkbox under the Networked menu. Mark the box with the <enter> key (if it is not already selected) and then hit F2 to continue. While the server cannot see any network at the time, it is easier to make these settings now.

8. In similar fashion, select “No” from the Use DHCP menu and continue.

9. The primary network interface for this machine will be eri0. Select that option and continue.

10. The Hostname screen allows you to type in your choice for this computer. You may use a fully qualified name here. Follow the on-screen instructions.

11. Next you will select the IP address of this host using dotted-quad notation. Enter your pre-selected address and continue.

12. The Solaris installer will now ask if the system is part of a subnet. Tell it “Yes.”

13. The following screen asks you to select a netmask. The value you choose will be dependent on your specific network. The default of 255.255.255.0 is often correct.

14. At this point you will have the option to enable IPv6 on the server. While IPv6 may provide a number of security enhancements to networking, at this time our facility has not developed the infrastructure and “best practices” to deploy it. Therefore, we will select “No” under the Enable IPv6 menu. Having an unneeded communication protocol on your system is an unwise choice.

15. The Default Route query will give you a choice of how this machine’s default route will be discovered. Chose the “Specify one” option and continue. On the next screen type in the IP address of your router and continue with F2.

16. You will now be presented with a confirmation screen. It will list the options you have chosen and should look like this:

```
----Confirm Information-------------------------------------
> Confirm the following information.  If it is correct,
press F2; to change any information, press F4.

                    Networked: Yes
                     Use DHCP: No
    Primary network interface: eri0
                    Host name: backup1.example.com
                   IP address: 192.168.1.200
      System part of a subnet: Yes
                      Netmask: 255.255.255.0
                  Enable IPv6: No
               Default route: Specify one
            Router IP Address: 192.168.1.1

-------------------------------------------------------------
```

Press the F2 key to accept these choices.

17. The next screen gives us the option to configure Kerberos security. Select "No". Press F2 to confirm your choice.

18. The Name Service screen: Select "None" as your name service and confirm. We may optionally configure a name service at a later time.

19. This brings us to the Time Zone screen. Select your time zone using the <enter> key to expand and collapse menu branches. This machine was configured as Americas > United States > Central. Then you may set the current date and time and confirm it.

20. Now that we have configured the identity of this computer, we will begin the options for the OS software installation. You will be asked if this is an upgrade or an initial install. Select "Initial" by pressing F4 to chose a pristine operating system and remove any older OS components that may be on the disk.

21. Next we must choose to do a standard install or a "flash" install. The Flash install option allows a pre-created image that includes both OS and applications to be rapidly installed on the machine over the network. Currently we are not connected to the network. Further, this system will likely be unique on the network, and the overhead of creating a flash image for it may not be cost effective. We will choose to do a standard install from CD by pressing F2.

22. The Select Geographic Region screen lets us add language support for various parts of the globe. The server will perform fine with none of these packages installed, and we will select no options at this screen. Continue with F2.

23. We have now reached the interesting part of the install – choosing what software to place on our disk. Begin by selecting “Core System Support 64-bit” with the arrow and <enter> keys. This will give us the minimum set of software that Sun considers a viable operating system. However, as brave and paranoid system administrators, we will further customize our package list by hitting F4.

The installer will present an ASCII menu containing four columns. [Our highly secure but primitive terminal connection prevents us from placing an informative screen capture here.] It will look something like this:

```
    [X]   Audio Drivers (64-bit)                    0.51 MB
 >  [/]   Audio Drivers and Applications            0.77 MB
    [ ]   AuditService Implementation               0.00 MB
    [ ]   AuditService Implementation (64-bit)      0.00 MB
    [X]   AutoFS, (Root)                            0.08 MB
    [X]   AutoFS, (Usr)                             0.27 MB
```

The first column will contain a “**>**”, a “**V**”, or nothing. The “**>**” will indicate the line represents a cluster of packages. Using the arrow keys to place the cursor on this symbol and pressing <enter> will cause the cluster to be expanded into a list of individual packages. It will also change the symbol to a “**V**”. Pressing the <enter> key again while over this symbol will hide the cluster’s package list and return the “greater than” symbol. A blank first column indicates the line represents a single package. An expanded cluster will look similar to the following:

```
    [X]   Audio Drivers (64-bit)                    0.51 MB
 V  [/]   Audio Drivers and Applications            0.77 MB
    [X]       Audio Applications                    0.38 MB
    [X]       Audio Drivers                         0.39 MB
    [ ]       Audio Header Files                    0.00 MB
    [ ]       Audio Sound Files                     0.00 MB
```

The second column will contain either a space, an **X**, an **!**, or a **/** enclosed in square brackets. A space means the package is unselected. You may select it by using the <enter> key while the cursor is over the brackets. An **X** means the package (or cluster) is selected. The **/** means the line is a cluster and that an incomplete set of its packages has been selected. An exclamation point indicates that you may not remove the package from the installation list.

The third column contains the descriptive name of the package or cluster. You may use the arrow keys to highlight this name. Hitting <enter> will then cause a more detailed description of the package to be displayed on its own screen. It is here than you will find the actual package name, such as SUNWauda. You may hit <enter> again to return to the menu.

The fourth column shows the size of a selected package and will be 0.00 MB if the package is not to be installed. The fourth column is not selectable.

24. Now that we have mastered the package selection tool, we will work our way down the list making changes.

Table 1: Initial Modifications to Core System Install

Action	Package	Description
Remove	SUNWauda	SunOS audio applications
Remove	SUNWatfsr	AutoFS, (Root)
Remove	SUNWatfsu	AutoFS , (Usr)
Remove	SUNWbsr	Boot server daemons (Root)
Remove	SUNWbsr	Boot server daemons (Usr)
Remove	SUNWftpr	FTP server configuration files
Remove	SUNWftpu	FTP server and utilities
Add	SUNWxwfnt	X Window system fonts {1}
Add	SUNWzlib	Zip compression library {2}
Add	SUNWj3rt	J2SDK 1.4 Java Runtime Environment {1}
Remove	SUNWkrbr	Kerberos version 5 support (Root)
Remove	SUNWkrbu	Kerberos version 5 support (Usr)
Remove	SUNWlldap	LDAP libraries for development
Add	SUNWmdb	Modular Debugger {3}
Add	SUNWmdbx	Modular Debugger (64-bit) {3}
Add	SUNWmfrun	Motif Runtime Kit {4}
Remove	SUNWnfscr	NFS client support (Root)
Remove	SUNWnfscu	NFS client support (Usr)
Remove	SUNWnfssr	NFS server support (Root)
Remove	SUNWnfssu	NFS server support (Usr)
Remove	SUNWnisr	NIS client (Root)
Remove	SUNWnisu	NIS client (Usr)
Add	SUNWntpr	NTP v3 daemon and utilities (Root)
Add	SUNWntpu	NTP (Var)
Add	SUNWctplx	Portable Layout service for CTL (64-bit) {5}
Add	SUNWctpls	Portable Layout service for CTL {5}
Remove	SUNWinamd	Internet Domain Name Server
Remove	SUNWrcmdr	Remote network server commands (Root)
Remove	SUNWrcmdu	Remote network server commands (Usr)
Remove	SUNWtnetr	Telnet server daemon (Root)
Remove	SUNWtnetu	Telnet server daemon (Usr)
Remove	SUNWtftp	Trivial File Transfer server
Remove	SUNWtftpr	Trivial File Transfer server (Root)
Remove	SUNWtnamr	Trivial Name Server (Root)
Remove	SUNWtnamd	Trivial Name Server (Usr)
Add	SUNWmdr	Solaris Volume Manager (Root)
Add	SUNWmdu	Solaris Volume Manager (Usr)
Add	SUNWmdx	Solaris Volume Manager Drivers (64-bit)

Action	Package	Description
Add	SUNWscpu	Source Compatability, (Usr) {6}
Add	SUNWlibC	SunWorkshop compilers libC {4}
Add	SUNWadmfw	System & Network Admin Framework
Add	SUNWadmc	System Administration Core Libraries
Add	SUNWxwrtl	X Windows system & graphics runtime {1}
Add	SUNWxwice	X Windows ICE components {1}
Add	SUNWxwplt	X Windows platform software {1}
Add	SUNWxwicx	X Windows ICE components (64-bit) {1}
Add	SUNWxwrtx	X Windows runtime package (64-bit) {1}
Add	SUNWxwplx	X Windows system library (64-bit) {1}
Add	SUNWtcpd	TCPD access control {2}

{1} Required for Veritas NetBackup Graphical User Interface
{2} Required for SSH cluster
{3} Sun requires for 280R platform (Sun[3])
{4} Required for Java Runtime Environment
{5} Required for Motif Runtime Kit
{6} Desired for NetBackup logging

Notes on package selection:

This package list was arrived at by exhaustive trial and error. It should provide the needed functionality but may lack some of the “bells and whistles” of a more fully featured OS. If you are tempted to remove anything from this list, be forewarned that the dependencies of the NetBackup software are not always easily determined.

We have attempted to remove as many packages as we can without entirely disabling the machine. Packages such as Telnet, FTP and TFTP have long presented security holes, as do many network daemons. Sun allows us to remove them easily in Solaris 9 – so we shall. There is little chance of the software being exploited to gain access to the system if it is not even on the disk. Similarly, we will not be using NFS on this server, so it is removed to avoid any chance of providing an avenue of attack. Packages such as the audio applications probably present a small risk. But we do not need them on this particular machine, so removing them reduces the number of files we have to audit and the number of places for “malware” to hide.

We have also added a couple of tools to help us improve the security and reliability of our server. NTP not only keeps our clock up to date, it allows for meaningful timestamps on log data (both local and remote) for assistance in performing post-mortems. TCPD will allow us to control more precisely who may talk to our server. The Solaris Volume Manager (formerly known as Solstice DiskSuite) cluster will allow us to mirror our two disks so that no data will be lost if and when one fails.

Most of the configuration and control tools for Veritas NetBackup are only available through an X Windows interface written in Java. This means we are forced to include a number of large packages (noted above) we might otherwise do without – nearly doubling our install size. The other packages added for install at this time are prerequisites for packages to be added later.

Finally, you may optionally choose to include the SUNWtoo and SUNWtoox packages from this disk if you think you will need tools like **truss** and **ldd** to debug your system. They can be removed once things are functioning correctly.

25. Once you have fully exited the package selection process, a screen prompting for disk layout will be presented. Use the arrow keys to select your primary boot device (in this case c1t0d0) and the hit F4 to customize and select slice 0 as your root partition. You will be asked if you wish to change the eeprom setting to always boot from your new choice. Do so. (You will need your prom password to do this.)

26. At the Preserve Data screen, merely continue. Do not choose to preserve any data.

27. Using the Manual Disk Layout option, allocate and size your partitions. Veritas NetBackup uses a large amount of disk space to store data about its clients, backup schedule information, and its catalogs of meta-data listing what images are stored on each tape. NetBackup prefers to be installed in /opt, so we will make a large /opt partition. We will also make separate root, /var, and /usr partitions. This will allow us to treat them differently, including mounting /usr as a read only device. With my 72 GB disk, I chose the following partition scheme:

Slice	**Partition**	**Size (MB)**
0	/	516
1	swap	1028
2	Overlap	69991
3	/var	2057
4		0
5	/opt	65339
6	/usr	1028
7		24

Note that a few megabytes have been left in slice 7. This gives us a location to store metadb replicas for our mirrored disks in a future step. Choose not to mount any remote files systems when asked. Review your install options and hit F2 to continue.

28. You may receive a warning screen at this point if you changed your boot device in the eeprom. You may ignore it.

29. The final installer screen will ask if you want a manual or automatic reboot after the install is complete. If you choose automatic reboot, do not go anywhere. Recall that you have not selected a root password yet. When the machine comes up, you will be able to log in as root with no password. Our stripped down OS will install in just a few minutes, so it won't be long.

30. Once the machine has been rebooted log in and set a root password. Choose a strong password, preferably one of at least 8 characters consisting of upper- and lower-case letters, numbers and other printable symbols. Make sure it is not based on information that could be easily guessed by knowing about you or your facility, but make sure it is one you can remember.

31. We are not quite done with our package selection yet. In theory, we could have chosen all the packages at the same time and let the installer ask for the "Solaris 9 Software 2 of 2" disk when it was ready for it. But in practice the installer depends on several parts of the Java Virtual Machine and expects a relatively modern output device, so it is cleaner to add the second disk's packages by hand.

Eject the "Solaris 9 Software 1 of 2" disk and load "Solaris 9 Software 2 of 2" into the CDROM drive. Mount the disk with the following command:

```
# mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /mnt
```

Change directory to `/mnt/Solaris_9/Product` and use the **pkgadd** command to add the packages in the following table. The syntax is:

```
# pkgadd -d . pkgname1 [pkgname2 pkgname3. . .]
```

Table 2: Additional Solaris Packages

Package	Description	Comment
SUNWaccr	System Accounting (Root)	
SUNWaccu	System Accounting (Usr)	
SUNWbash	GNU Bourne-again shell	Required for SUNWgzip
SUNWcpp	C preprocessor	Required for X Windows
SUNWgcmn	Common GNU packages	Required for SUNWgtar
SUNWgtar	GNU tar utility	Tar used by NetBackup
SUNWgzip	Gzip compression utility	Used by NetBackup
SUNWhea	C header files	Required for 280R
SUNWj3rtx	J2SDK 1.4 JRE (64-bit)	Req'd for NetBackup GUI
SUNWpstl.u	Apptrace support	Required for 280R
SUNWpstlx.u	Apptrace support (64-bit)	Required for 280R
SUNWsshcu	SSH common utilities (Usr)	Our communication tool
SUNWsshr	SSH client and utilities (Root)	
SUNWsshu	SSH client and utilities (Usr)	

Package	Description	Comment
SUNWsshdr	SSH server (Root)	
SUNWsshdu	SSH server (Usr)	
SUNWzip	Zip compression utility	

We add SSH to be our replacement communication protocol for Telnet and FTP. The advantages of SSH are well known (OpenSSH, Pomeranz[3]), but cannot be overstated. End-to-end encryption, public/private key identification, and automatic X forwarding are just some of the "must have" features of SSH.

We have also added tools like gzip and GNU tar, which most administrators use regularly and will be valuable in handling data produced on our backup server. While many people acquire third party packages for these tools, or compile their own from source, using a version which is "part" of the operating system increases the likelihood that it will be patched for security holes discovered in the future during regular patch cluster updates. While we may instantly rebuild sshd when a vulnerability is announced, many administrators are less likely to fix a buffer overflow in gzip. Sun will now fix it for us.

32. Once all packages have been added we will remove the "Solaris 9 Software 2 of 2" CD and reboot the system. This will allow us to check that services such as SSH come up correctly.

```
# cd / ; umount /mnt
# reboot
```

Additional Software and Patches

Now that we are up and running, we will need some additional software to make our server fully operational. We will also need the latest patches for the operating system. Since we do not have a network connection, we will acquire the software on another host and burn it to CDROM there.

First, we have to add drivers for the specific SCSI card used in this system. These drivers are not included in the Solaris 9 package. The packages we will need to add are SUNWqus, SUNWqusu, SUNWqusx, and SUNWqusux. This software is obtainable from Sun Microsystems by following the "Sun StorEdge PCI Dual Ultra3 SCSI Host Adapter v1.0" link at http://wwws.sun.com/software/download/allproducts.html. Unfortunately, you must have a SunDownload account to access the drivers on the following page. The SunDownload account is free and you will be prompted to create an account if you do not already have one (or another Sun online account such as SunSolve.) Note that there are packages for both Solaris 8 and Solaris 9 in ".tar.Z" format – be sure to get the correct ones. There does not seem to be any

way to verify the files downloaded through this procedure through md5 checksum or PGP signature. We will have to trust that the files made available by Sun via this SSL wrappered web site are what they claim to be.

We will also need both the latest recommended and security patch cluster from Sun and a special patch for our SCSI card. The Recommended and Security patches will clean up bugs and security problems that have been fixed since our operating system CD's were released. Patches may be obtained from Sun at http://sunsolve.sun.com. For the patch cluster, follow the links to the Recommended & Security Patches page and download the current Solaris 9 cluster. Verify the md5 checksum on the resulting 9_Recommended.zip file to be sure your file downloaded cleanly. While here, we will collect patch 112706 for our SCSI card. On the initial patch page is a "Find Patch" tool that will take you to the latest version of the patch (112706-02 as of this writing.)

A Note on Third Party Applications:
If you are truly paranoid (like this author) you will have a dedicated "build" host that has been appropriately hardened, but contains all the tools necessary for building software from source. When able, you will acquire software packages as source code from the official repository for that package. PGP signatures or MD5 checksums from multiple trusted sources other than the one you received the source code from will be checked against those of the package. (Don't place too much trust in a checksum from the same server as the source package. If one has been replaced with a modified version, the other may have been as well.) Make a quick review of the source for obvious backdoors or security holes. A thorough code review is not usually feasible, but often even a cursory check will often reveal interesting information. Next, configure the package (either with "configure" or by modifying the make file) to build into a non-standard location and compile. In addition to a number of non-security related package management benefits, this avoids being caught by automated exploit scripts that search the standard paths for known exploitable software. It will do little against an observant hacker, however. If you have done this right, all libraries and modules needed to run the software will have been placed into the same application directory and can easily be **scp**'ed as a tar file to another machine. Optionally, for a Solaris OS you can create a package that may be added to other systems by use of the **pkgadd** command. Likewise, you can build an **rpm** for a Linux based system.

Since this document is meant to be a straight forward guide to building a secure NetBackup server that can be used by a wide range of system administrators, we will not take a detour into software dependencies, editing makefiles, statically vs. dynamically linked binaries and source code auditing. Instead, we will build one of the third party packages in a relatively vanilla fashion and acquire the other as a precompiled binary to illustrate two approaches.

To assist us in monitoring the configuration of our server, we will use AIDE. According to the author, "AIDE (Advanced intrusion detection environment) is an intrusion detection program. More specifically a file integrity checker."(Lehti) AIDE is a successor to the venerable Tripwire program. The source code is available on the AIDE home page via http://www.cs.tut.fi/~rammer/aide-0.9.tar.gz.

For this package we will take the expedient shortcut of downloading a precompiled binary from www.sunfreeware.com. Those of you with the knowledge and necessary level of paranoia may take the approach outlined earlier.

At the time of writing the latest version of AIDE is v0.9. Obtain the Solaris 9/SPARC version of this package from sunfreeware.com. In order for the md5 checksum and file size to match with the values published on the web site, you may need to connect via FTP and download the "aide-0.9-sol9-sparc-local.gz" file. (Some browsers may automatically unzip the file on the fly, changing the checksums.)

To provide for additional network security we will want to have the IP Filter package. This handy software written by Darren Reed can provide "network address translation (NAT) or firewall services."(Reed) If you prefer a precompiled binary, one is available at http://www.maraudingpirates.org/ipfilter/. To provide a simple example, I chose to build this package myself from source (available at http://coombs.anu.edu.au/~avalon/ip-fil3.4.33pre1.tar.gz). If you do this you will need a compiler capable of creating 64-bit binaries and you should not use the GNU **make**. Sun's **make** works fine.

I built the package on the secure build engine alluded to above. A bare outline of the required commands follows.

```
# gunzip -c ip-fil3.4.33pre1.tar.gz | tar xvf -
# cd ip_fil3.4.33pre1
# make solaris
# cd SunOS5
# make package
```

Inside the architecture specific subdirectory (sparc-5.9) will be a file named `ipf.pkg`. This is the file we will burn to CD and take to our new server.

In addition to the above software, there are a couple of tools that will be handy. First, the FixModes script by Caspar Dik will check for permissions, SUID bits, and other interesting issues. A version precompiled for Solaris may be obtained from www.sun.com/solutions/blueprints/tools. Again you will need the Sun Download account to collect this. Also, the Center for Internet Security has a

scoring tool for Solaris to point out potential security flaws in your configuration. This may be acquired from www.cisecurity.com.

You should now have:
112706-02.zip
9_Recommended.zip
FixModes.tar.Z
SUNWqus.tar.Z
SUNWqusu.tar.Z
SUNWqusux.tar.Z
SUNWqusx.tar.Z
aide-0.9-sol9-sparc-local.gz
cis.tar.Z
ipf.pkg

Burn these files to a CDROM and mount it on the new backup server.

```
# mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /mnt
```

Copy the files onto the local disk and begin by installing the driver packages for the SCSI card. You should always install all needed Solaris packages before adding patches.

```
# mkdir /opt/cdrom
# cp /mnt/* /opt/cdrom
# cd /opt/cdrom
# uncompress *.Z
# for i in *.tar
> do
> tar xf $i
> done
# pkgadd -d . SUNWqus SUNWqusu SUNWqusx SUNWqusux
```

Now that the SCSI drivers are installed, patch them.

```
# unzip 112706-02.zip
# patchadd 112706-02
```

At this point we may install the Solaris 9 Recommended & Security patch cluster.

```
# unzip 9_Recommended.zip
# cd 9_Recommended
# ./install_cluster
```

You will see a number of errors during the patch process. Patches that fail with “return code 2” have previously been applied. Patches that fail with “return code 8” apply to packages you do not have installed. You will likely see several of

these. For my particular combination of OS release and patch cluster, I also had a couple of patches fail with "return code 25." This error means a prerequisite patch has not been applied. In my case, these patches were destined for the Kerberos software that had not been installed, so they could safely be ignored. (The prerequisite patch had failed with return code 8 earlier in the cluster.) If you have failure codes other than 2 or 8, you will need to track them down and figure out what went wrong. Note that you can find a complete listing of the return codes inside the `/usr/sbin/patchadd` script itself.

Now the AIDE package will be installed.

```
# gunzip aide-0.9-sol9-sparc-local.gz
# pkgadd -d aide-0.9-sol9-sparc-local
```

We will leave the configuration of the AIDE software for a later step and install the IP Filter package:

```
# pkgadd -d ipf.pkg
```

There are two packages: `ipf` and `ipfx`. Install `ipfx` first, and then `ipf`. Your file name in the example will differ if you used a precompiled binary.

Finally, install the CISscan package.

```
# cd /opt/cdrom/cis
# pkgadd -d CISscan
```

Once all software from the CDROM has been installed, reboot and remove the CD from the drive.

Veritas NetBackup Installation

There are two small adjustments we will make to our OS configuration before installing the backup software. First, we will completely replace our `/etc/inet/inetd.conf` file. [Note that for backward compatibility, there is a symbolic link from `/etc/inetd.conf` to `/etc/inet/inetd.conf`. We will use this shorter path in most `inetd.conf` modifications to follow.] There are a number of services listed there initially, but none that we need. The NetBackup installer will modify this file by adding some lines at the end for new listeners. Rather than clean it up later, we'll just fix it in advance.

```
# mv /etc/inet/inetd.conf /etc/inet/inetd.conf.orig
# echo '# Only 4 NetBackup daemons' > /etc/inetd.conf
```

Second, you should **vi** your `/etc/hosts` file so that both the fully qualified host name and the short node name appear on the appropriate line. It should look similar to this:

```
# cat /etc/hosts
#
# Internet host table
#
127.0.0.1       localhost
192.168.1.200 backup1.example.com backup1 loghost
```

The installer will attempt to resolve the local machine during the install process and different calls may use either version of the hostname.

The backup software we are using is Veritas NetBackup 4.5 Feature Pack 3. The software comes on several CDs. Insert the disk labeled "VERITAS NetBackup DataCenter 4.5 Feature Pack 3 for Solaris" into the drive and kick off the installer.

```
# mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /mnt
# /mnt/install
```

1. The installer script will ask which install option you want. Answer "1" for NetBackup.

2. When asked if it is OK to install the software in `/opt`, answer in the affirmative.

3. Next you will be prompted to install any additional copies of the client software for operating systems other than the one on which you are currently installing. NetBackup has an option to push client software to its clients. Note that it will not push to Windows clients, and that pushing to Unix requires the use of a `/.rhosts` file on the client during the install. You always have the option to install the client software on each client from CD. Since we will shortly be breaking many of the rpc calls that NetBackup uses to push the client software out, there is no point in installing clients for other operating systems.

4. Once the package has been installed, you will be asked to enter your license key. You will need a valid NetBackup license key to continue. If you have additional keys for optional features, enter them all at this time following the directions on screen.

5. The installer will ask if this machine (by name) is the Master Server. This will be our Master server, so answer "yes."

6. When asked for a media server, enter this machine's hostname again.

7. In this step we will also identify this server as the Global Device Database server. If you were building this server as part of a large backup cluster, this answer and the previous two might be different – but we are considering a minimal case for purposes of this document.

8. When prompted to start the NetBackup daemon **bpdbm**, do so.

9. You will be asked if you want to create example policies and schedules. If you have little experience with NetBackup, you may want to do this. Otherwise, say "no."

10. Answer "yes" when asked if the Media Manager should be started.

11. Answer "yes" when asked if the **bprd** daemon should be started.

12. Quit when the installer script returns to the initial menu.

You will have noticed that the `/etc/services` file and `/etc/inetd.conf` were edited by the installer. A number of services were added. We will look at these more closely later. Our Veritas NetBackup software is now installed. Its configuration will be discussed in a separate section.

System Configuration

TCP Wrappers

We have already begun to improve the security of our operating system by the removal of services from `/etc/inetd.conf` mentioned in the previous section. We will continue this task by enabling the TCP Wrappers functionality that we installed with the SUNWtcpd package. TCP Wrappers is a package originally developed by Wietse Venema for finer grained control over network services launched from **inetd** and now included in Solaris 9.

First we will create the `/etc/hosts.allow` file, which defines which IP addresses are allowed to connect to services started from **inetd**.

```
# echo 'ALL: 127.0.0.1 192.168.1.200' > /etc/hosts.allow
# echo 'sshd: 192.168.1. 192.168.2.' >>/etc/hosts.allow
```

Our first line gives permission to the local host to connect to himself. It is a good practice to explicitly allow this at the top of your configuration. The second line allows SSH connections from our server network (192.168.1.0/24) and the network where our administrators live (192.168.2.0/24.)

Next we will create the `/etc/hosts.deny` file.

```
# echo 'ALL: ALL' > /etc/hosts.deny
```

Finally, we will enable **tcpd**. This is easily done in Solaris 9:

```
# echo 'ENABLE_TCPWRAPPERS=YES' >>/etc/default/inetd
# kill -HUP <inetd process ID>
```

You may be asking yourself how we expect to back up any hosts when we haven't allowed them in `/etc/hosts.allow`. There is good news and bad news. The good news is that the sevices listed in `/etc/inetd.conf` are used to call up the NetBackup graphical user interface (GUI) and to handle communication between the master server and the media server. In this case the master and media server are the same machine, so our first line solves that problem. And the GUI only needs to be launched on this server as well.

The bad news is that NetBackup starts several additional daemons at boot time, but not out of **inetd**, so we can't wrapper them. Further, during the actual backup or restore process, a large number of connections are made on ephemeral ports. Again, TCP Wrappers won't help us. A more powerful tool will be needed to fully secure network connections.

SSH Configuration

We already have SSH installed. Conveniently, since both it and TCP Wrappers are part of Solaris 9, SSH is already using the hosts.allow and hosts.deny directives. Further, the default configuration of `/etc/ssh/sshd_config` is actually pretty good. It does not allow root logins or logins to passwordless accounts. It does not honor `.rhosts` files or speak the version 1 protocol. We will make only a few changes.

First, we need to enable X11 forwarding so we can display the NetBackup GUI through the SSH tunnel to our workstation. Edit the /etc/ssh/sshd_config and change "`X11Forwarding no`" to "`X11Forwarding yes`". Next, uncomment the "`Banner /etc/issue`" line and change it to read "`Banner /etc/motd`". Finally, lower the number of login attempts before hanging up and turn on logging for all failed attempts – change "`MaxAuthTries 6`" to "`MaxAuthTries 3`" and "`MaxAuthTriesLog 3`" to "`MaxAuthTriesLog 0`". In addition to being a fairly standard setting, this last change is recommended by the Center for Internet Security whose Solaris Benchmark document even provides a nice awk script to perform it for you (CIS).

Last, we'll get rid of our default message of the day (which reveals we are running Solaris 9) and replace it with a more generic warning.

```
# echo 'This machine for authorized use only.  Activity is
monitored and reported' > /etc/motd
```

We don't expect to be using the SSH client much on this server. But we'll take a look at its configuration file as well. Note that the defaults are listed as comments in `/etc/ssh/ssh_config`. They are pretty reasonable. There is one change we will make – we don't want to fall back to SSH protocol version 1 while making an outgoing call.

```
# echo 'Host *' >> /etc/ssh/ssh_config
# echo 'Protocol 2' >> /etc/ssh/ssh_config
```

Sendmail

We will want to receive messages from our backup server about cron jobs, backup failures, and other information. But we don't want listeners running on exposed ports. Unfortunately, the sendmail that ships with Solaris 9 is a bit different from earlier versions and will require a different approach to lock it down. Not only does it listen on tcp port 25, it also listens on tcp port 587. Outgoing mail goes through an extra "submission" hop, delivering to *localhost*. If the MODE option were set in `/etc/default/sendmail` to stop sendmail from listening on port 25 (as in Solaris 8), all outbound mail would stop.

The procedure below is a modified version of that outlined in the December 2002 edition of *Solaris Operating Environment Security* by Alex Noordergraaf and Keith Watson (Noordergraaf).

```
# cd /usr/lib/mail/cf
# cp subsidiary.mc backup1.mc
```

Edit `backup1.mc`. Remove the two MAILER() lines and add the following three lines in their place.

```
DAEMON_OPTIONS(`Name=NoMTA4, Family=inet, Addr=127.0.0.1')dnl
FEATURE(`no_default_msa')dnl
FEATURE(`nullclient',`mailhub.example.com')dnl
```

The mailhub above will be your local primary mail server. Then:

```
# /usr/ccs/bin/m4 ../m4/cf.m4 backup1.mc > backup1.cf
# cp backup1.cf /etc/mail/sendmail.cf
# /etc/init.d/sendmail restart
```

The result of this is a sendmail that still runs in "-bd" mode but does not listen on ports 25 or 587 and happily sends off all mail (even unqualified usernames) to your mail hub.

NTP Configuration

As previously mentioned, running NTP on your server is beneficial for allowing precise log correlation with other systems. Additionally, since our backup server performs most of its functions based on time of day, we would like it to stay synchronized with the rest of our network. We included NTP in our package selection and a script is already in place at `/etc/rc2.d/S74xntpd` to start NTP at boot time. Since that script checks for the presence of a configuration file to determine if it is to start, we merely have to put our configuration in place and kick it off.

```
# cat <<EOF > /etc/inet/ntp.conf
# NTP configuration
restrict default ignore
restrict timeserverIP1
restrict timeserverIP2
restrict timeserverIP3
restrict 127.0.0.1
restrict 127.127.1.1

server timeserverIP1
server timeserverIP2
server timeserverIP3

driftfile /etc/ntp.drift
EOF
# /etc/init.d/xntpd start
```

We have told NTP to ignore anyone but our three timeservers. Then we remove the restrictions for these servers and the localhost. It should be noted that these three timeservers are all inside our firewall, reside on our server network, and are synchronized to a local GPS clock. They will be trusted to provide accurate time information. Finally, we tell NTP where to store its drift information. This information is regularly updated and used by NTP to converge to the correct time quickly on boot up. The `/etc/ntp.drift` file will be created by NTP if it is not already present.

Boot scripts

Although we installed only a small part of the full Solaris operating system, there are still some processes and boot scripts installed that we don't really need. A quick perusal will show that `/etc/rc2.d` is the only "rc" directory that we really have to clean up.

We aren't using network file systems or any rpc services, so we can get rid of `S71rpc`, `S73cachefs.daemon`, `S76nscd` and `S93cacheos.finish`. We are not using LDAP, so `S71ldap.client` can be removed. The autoconfiguration tools `S30sysid.net`, `S71sysid.sys` and `S72autoinstall` can be dispensed with. Finally, the **vi** preserve tool has been known to have security issues in the past, and we don't really need its functionality. So `S89PRESERVE` can be removed.

In keeping with the good administration practice of making changes reversible, we will simply rename these files to something that will not run at boot time, rather than remove them entirely. This approach is one I have used for years, and is also "blessed" by the SANS Institute in *Solaris Security Step by Step v2.0* (Pomeranz[1]).

```
# cd /etc/rc2.d
# for file in S30sysid.net S71ldap.client S71rpc \
S71sysid.sys S72autoinstall S73cachefs.daemon S76nscd \
S89PRESERVE S93cacheos.finish
> do
> mv $file .NOT$file
> done
#
```

While we are in `/etc/rc2.d`, take a look at `S72inetsvc`. Most of this file is for dealing with DHCP, multicast, starting nameserver daemons and other tasks. We are going to replace it with a two-line file that does only what we need.

```
# cd /etc/rc2.d
# mv S72inetsvc .NOTS72inetsvc
# cat <<EOF >/etc/init.d/secinetsvc
> /usr/sbin/ifconfig -au netmask + broadcast +
> /usr/sbin/inetd -s -t
> EOF
# chown root:root /etc/init.d/secinetsvc
# chmod 744 /etc/init.d/secinetsvc
# ln -s /etc/init.d/secinetsvc S72inetsvc
```

Note that we have added the "–t" to our invocation of **inetd** to enable connection logging. This `secinetsvc` script is a stripped down version of the init script outlined in the SANS 6.5 Unix Practicum coursebook (Pomeranz[2], p. 43).

Finally, while we are thinking about networking and boot scripts, we will adjust some parameters of the tcp stack. Persons familiar with Solaris may have used the **ndd** command to solve problems with computers and switches that do not autonegotiate correctly. Here we will use it to enhance the security and performance of the server's network stack. While we will be changing a number of parameters, we will not be as aggressive as some experts. The rate-limiting step in our backups is likely to be the squeezing of multiple backup streams through the network interface. Thus, certain changes that increase the expense or number of network calls may not be implemented (Guru).

Briefly, this script will turn off IP forwarding, drop source routed packets and ICMP redirects, and ignore a number of broadcast packets. It will also attempt to avoid denials of service by increasing the number of tcp connections allowed and decreasing timeouts.

```
# cat <<EOF > /etc/init.d/setndd
#!/bin/sh
ndd -set /dev/ip ip_forwarding 0
ndd -set /dev/ip ip_strict_dst_multihoming 1
ndd -set /dev/ip ip_forward_src_routed 0
ndd -set /dev/ip ip_ignore_redirect 1
ndd -set /dev/ip ip_send_redirects 0
ndd -set /dev/ip ip_forward_directed_broadcasts 0
ndd -set /dev/ip ip_respond_to_address_mask_broadcast 0
ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0
ndd -set /dev/tcp tcp_conn_req_max_q0 8192
ndd -set /dev/tcp tcp_ip_abort_cinterval 60000
EOF
# chown root:sys /etc/init.d/setndd
# chmod 744 /etc/init.d/setndd
# ln -s /etc/init.d/setndd /etc/rc2.d/S69setndd
```

As another improvement to the network behavior of this host, change the default ISS setting in `/etc/default/inetinit` to read "TCP_STRONG_ISS=2". This improves the initial tcp sequence number selection algorithm.

This is probably a good time to reboot and make sure all these changes are working.

Logging

In the previous section we added logging to `inetd` with the "-t" flag. Now we will turn our attention to logging. First, we will adjust the behavior of syslog so that it does not listen for external messages on udp port 514. In Solaris 9 this can be done quite easily:

```
# echo 'LOG_FROM_REMOTE=NO' >> /etc/default/syslogd
# /etc/init.d/syslog stop
# /etc/init.d/syslog start
```

A quick `netstat -a` at this point should show only NTP bound to a udp port.

Now enable logging of AUTH events and our **inetd** connections with:

```
# echo \
'auth.info          /var/log/authlog'>>/etc/syslog.conf
# echo 'daemon.debug          /var/log/connlog'>> \
/etc/syslog.conf
# touch /var/log/connlog
# chmod 600 /var/log/connlog
```

Next enable logging of failed login attempts:

```
# echo 'SYSLOG_FAILED_LOGINS=0' >> /etc/default/login
# touch /var/adm/loginlog
# chmod 600 /var/adm/loginlog
```

Finally, the system accounting tools will be put to work to log the hardware activity of our system. This is a step recommended by the Center for Internet Security, but in our case it will be more useful as a gauge of bottlenecks in our backup system and for future server sizing requirements. Note, however, that enabling system accounting can cause a bottleneck. If our backup server is highly loaded, we may be better off disabling system accounting after a review of the tradeoffs.

In `/etc/init.d`, copy the `perf` file to `ourperf`. Edit `ourperf` and uncomment the line:

```
/usr/bin/su sys -c "/usr/lib/sa/sadc \
 /var/adm/sa/sa`date +%d`"
```

Then:

```
# chown root:sys /etc/init.d/ourperf
```

```
# rm /etc/rc2.d/S21perf
# ln -s /etc/init.d/ourperf /etc/rc2.d/S21perf
```

and add the following lines to the sys crontab:

```
0,20,40 * * * * /usr/lib/sa/sa1
45 23 * * * /usr/lib/sa/sa2 -s 0:00 -e 23:59 -i 1200 -A
```

Daily reports will be placed in `/var/adm/sa/`. As with all of the log files we have just activated, they should be regularly reviewed.

To help us rotate our log files, we will use the Solaris 9 **logadm** command. It is already configured and running out of `cron`, so we just need to add a few lines to `/etc/logadm.conf`. Lines like:

```
/var/log/authlog -C 8 -a \
'kill -HUP `cat /var/run/syslog.pid`'
```

should be sufficient. After some trial and error you will probably want to add more options to tune your log rotations.

Configuring the File System

The two disks that this system shipped with will be placed in a mirrored configuration using the Solaris Volume Manager (SVM) packages that were installed earlier. Currently, we are booting off the disk located at `c1t0d0`. The second disk will be partitioned to match it with the following command.

```
# prtvtoc -h /dev/rdsk/c1t0d0s2 | \
fmthard -s - /dev/rdsk/c1t1d0s2
```

Now that the two disks are partitioned the same, begin by creating a database in which to store the metadevice information. We will actually create three replicas on each of two slices.

```
# metadb -a -c 3 -f /dev/dsk/c1t0d0s7 /dev/dsk/c1t1d0s7
```

Next create the "stripes" that will be mirrored. Each stripe here will only contain one disk slice, but stripes can be made of concatenations of multiple slices. The "–f" option is used because we are working with mounted file systems. For an explanation of the finer points of Solaris Volume Manager see the man pages or the Solaris Volume Manager Administration Guide at http://docs.sun.com/db/doc/806-6111.

```
# metainit -f d10 1 1 c1t0d0s0
```

```
d10: Concat/Stripe is setup
# metainit -f d20 1 1 c1t1d0s0
d20: Concat/Stripe is setup
# metainit -f d13 1 1 c1t0d0s3
d13: Concat/Stripe is setup
# metainit -f d23 1 1 c1t1d0s3
d23: Concat/Stripe is setup
# metainit -f d15 1 1 c1t0d0s5
d15: Concat/Stripe is setup
# metainit -f d25 1 1 c1t1d0s5
d25: Concat/Stripe is setup
# metainit -f d16 1 1 c1t0d0s6
d16: Concat/Stripe is setup
# metainit -f d26 1 1 c1t1d0s6
d26: Concat/Stripe is setup
```

Now that the slices are under the control of SVM, the mirrors can be created. Initially, only one "side" will be attached to each mirror.

```
# metainit d0 -m d10
d0: Mirror is setup
# metainit d3 -m d13
d3: Mirror is setup
# metainit d5 -m d15
d5: Mirror is setup
# metainit d6 -m d16
d6: Mirror is setup
```

You will note that *swap* was not mirrored. There will be an extra partition on the second disk for some future use. Once the mirrors are set up and have the data containing slices in them, issue the **metaroot** command to update `/etc/system`, `/kernel/drv/md.conf` and `/etc/vfstab`.

```
# metaroot d0
```

To complete the updating of `/etc/vfstab`, **vi** the file and change the device information in the same way that the root partition information has been changed. Namely, "`/dev/dsk/c1t0d0s6`" will become "`/dev/md/dsk/d6`" and "`/dev/rdsk/c1t0d0s6`" will become "`/dev/md/rdsk/d6`". Other partitions follow suit. When you are done, reboot the system.

Do not be concerned when you see a warning about failed "forceloads." The mirrors are still working. All there is left to do is attach the other slices so that the mirrors are actually mirrored.

```
# metattach d0 d20
d0: submirror d20 is attached
```

```
# metattach d3 d23
d3: submirror d23 is attached
# metattach d5 d25
d5: submirror d25 is attached
# metattach d6 d26
d6: submirror d26 is attached
```

At this point the mirrors will start to sync up. Their progress can by watched with the **metastat** command. WARNING: Do not reboot the server until the mirrors are done syncing. In this case there is a 63 GB partition to manage, so it will take some time.

Admittedly, this configuration would be better with separate SCSI controllers for each disk. The SCSI controller is still a single point of failure. But controllers fail less often than disks, so we have improved our reliability quite a bit.

While the disks are syncing, return to the `/etc/vfstab` and add mount options to the partitions. In the seventh column add the options "`nosuid,logging`" to the `/var` and `/opt` partitions in place of the dash. Add the options "`remount,logging`" to the root file system. The `/usr` file system will receive the "`ro`" option. The result should be something like this:

```
# cat /etc/vfstab
#device     device    mount     FS    fsck    mount   mount
#to mount   to fsck   point    type   pass    at boot options
#
fd      -       /dev/fd fd     -       no      -
/proc   -       /proc   proc   -       no      -
/dev/dsk/c1t0d0s1    -   -       swap    -    no      -
/dev/md/dsk/d0  /dev/md/rdsk/d0 /     ufs  1  no remount,logging
/dev/md/dsk/d6  /dev/md/rdsk/d6 /usr  ufs  1  no      ro
/dev/md/dsk/d3  /dev/md/rdsk/d3 /var  ufs  1  no  nosuid,logging
/dev/md/dsk/d5  /dev/md/rdsk/d5 /opt  ufs  2  yes nosuid,logging
swap             -              /tmp  tmpfs   -   yes     -
```

The logging option creates a journaling file system. Disk writes are logged before being committed, and in the event of an ungraceful shutdown, can be replayed. This avoids file system corruption and manual **fsck** runs. For the root partition this improves both reliability and security. For the `/opt` partition, this can save a lot of **fsck** time. No "nosuid" option disables set UID programs from running on those partitions. Finally, the `/usr` partition will be mounted "read only." This increases the difficulty of placing compromised software in the trusted `/usr/bin` and `/usr/sbin` directories. Note that if we have changes to make in `/usr`, we will have to remount that partition read-write. This can be done with the command:

```
# mount -o remount,rw /usr
```

Finally, although not really a file system related command, there are four lines to add to `/etc/system`. The first two are recommended by Veritas for the performance of NetBackup. The second two are for stopping and logging "stack" attacks. According to the Sun Blueprints, "The SPARCv9 64-bit API prohibits the execute flag on stack pages (Noordergraaf, p. 26)." But we will add these lines to deal with 32-bit binaries.

```
# cat <<EOF >> /etc/system
> forceload: drv/st
> forceload: drv/sg
> set noexec_user_stack = 1
> set noexec_user_stack_log = 1
> EOF
#
```

Like the changes to the vfstab above, these will not take effect until a reboot. But remember not to reboot until your mirrors are done syncing.

User and Network Configuration

It is time to prepare the system for joining a network. Step one is:

```
# touch /etc/notrouter
```

At this time the server only has one network interface, and we have already told the server not to act as a router in our `/etc/init.d/setndd` script, but it is good to get in the habit of adding this file.

Next, we will configure DNS as our name service. Note that important hosts should be explicitly listed in `/etc/inet/hosts`, but there will be occasions when we need to talk to some "unknown" systems. Create `/etc/resolv.conf` with the correct search order and nameservers listed. Then edit `/etc/nsswitch.conf` and change the "hosts" line to:

```
hosts: files dns
```

There are twelve entries in `/etc/passwd` besides root. Make sure that none of them has UID 0 and that each of them is locked. Also, give each one `/dev/null` as a login shell.

```
# for acc in daemon bin sys adm lp uucp nuucp smmsp \
listen nobody noaccess nobody4
> do
> passwd -l $acc
> passmgmt -m -d /dev/null $acc
```

```
> done
#
```

Edit `/etc/default/passwd` and set `PASSLENGTH=8`. Other sysadmins may not be as conscientious as you are. Likewise, choose a value for `MAXWEEKS` of 13 or less and enter it and set `MINWEEKS=1`. Remember that only sysadmins will have accounts on this server and will be able to circumvent these restrictions if they are made burdensome. Create an entry `WARNWEEKS=3`, to remind you to change your password before it expires.

Create an account for yourself with the **useradd** command. You will need this to log in over the network because remote root logins are not allowed. Make sure to create a home directory and set a password. Other users may be added later.

```
# useradd -u 1101 -g 14 -d /opt/home/username \
-s /bin/csh username
# mkdir -p /opt/home/username
# chown username:14 /opt/home/username
# chmod 750 /opt/home/username
# passwd username
New Password:
Re-enter new Password:
passwd: password successfully changed for username
```

There will be no users on this system that aren't experienced, trusted, and privileged. However, it is possible that an administrator's password might be compromised, say for example it was the same as a password used on another domain that is not as secure – a bad practice, of course. Someone logging in as said user will not necessarily have the root password and may look for a way to gain elevated privileges. So we will still take some steps to restrict user access by changing the default file permissions on newly created files.

Begin by editing `/etc/default/login` and changing the line "`#UMASK=022`" to "`UMASK=077`". Likewise, edit `/etc/profile` and change the line "`umask 022`" to "`umask 077`". Then to fix `/etc/.login`:

```
# echo 'umask 077' >> /etc/.login
```

Last we will clean up the **cron** facility. There should be no additional crontabs at this point, but you can check in `/var/spool/cron/crontabs` and `/var/spool/cron/atjobs` to make sure. Then limit access to the files and to the **crontab** command to root:

```
# chown -R root:root /var/spool/cron
# chmod -R go-rwx /var/spool/cron
# echo 'root' > /etc/cron.d/cron.allow
```

```
# echo 'root' > /etc/cron.d/at.allow
```

We will run the **fix-modes** program at this time to help clean up permissions and ownership of files. In an earlier step we placed this software in `/opt/cdrom` and unpacked it.

```
# /opt/cdrom/FixModes/fix-modes
```

There will be a few warnings about being unable to change files and directories we have not installed. This is not a problem. The changes that **fix-modes** made can be found in `/var/sadm/install/contents.mods`. There are very few changes, and only to third party packages. Apparently Sun has improved its security awareness quite a bit with Solaris 9.

NetBackup Configuration

We now turn to the configuration of the backup software. In order to configure tape drives, backup policies, volume catalogs and other backup related items, we will have to attach an X Windows capable host to our "air gap" network. [If you do not wish to connect a client machine at this time, and you have faith that the backup server will actually perform in its current configuration, you may skip ahead to NetBackup Security Enhancements.] This machine will be used to ssh to the server and launch the NetBackup GUI. Use of this tool is the only reasonable way to perform most NetBackup functions. Our client machine should be clean (preferably a new build) and definitely NOT connected to another network.

An in depth recipe for configuring Veritas NetBackup is outside the scope of this paper. Veritas will be happy provide you with a one week course on configuring and managing your new backup server. We will lightly skim over the regular configuration (which would vary considerably based on your hardware in any case) and then spend some time looking at the potential methods of securing the installation.

Assuming all disk mirrors are synced up, verify that your tape library is connected to the server and powered on. Reboot the server with "`reboot -- -r`". Once the server is up, ssh from the client machine as yourself. Launch the GUI with the command:

```
% /usr/openv/netbackup/bin/jnbSA &
```

At the pop-up login screen enter "root" as the username and the root password in the password field and log in. In the right hand panel of the Administration Console, click on the Getting Started button. This will launch an applet that will walk you through the steps of configuring storage devices, configuring volumes,

configuring catalog backups, and creating a backup policy. These steps are specific to your environment, but should be easy to perform.

To test your NetBackup installation, you may optionally configure a backup policy for the client, install the client software on the client, and perform a test backup and restore. If you have followed each step in this document correctly, both should succeed.

While there are a number of backup related functions available in the GUI, there are no security options here. Close the Administration Console.

NetBackup Security Enhancements

From a network security standpoint, NetBackup is a nightmare. Not only does it spawn its own listening daemons from outside of **inetd** at boot time, it also opens multiple ephemeral tcp ports in both reserved and unreserved ranges during a backup or restore job. A look at the `/etc/services` file that was modified during the NetBackup install shows 22 new named tcp ports. A standard backup job may require 11 network connections (more if media multiplexing is used) in some combination between the Master server, the Media server, and the client. A standard restore requires 13 connections (again, more if media multiplexing is involved.) To briefly outline what we are up against, here is a breakdown of a backup as seen from the network perspective.

1. [Note: all port numbers are tcp ports.] The client issues a backup request from a non-reserved port (> 1023, let's call it Client-N1) to the **bprd** daemon on the master server (port 13720, not started from **inetd**.)

2. If NetBackup authentication is being used, the master server calls back from a non-reserved port (Master-N1) to **vopied** (port 13783) on the client. An exchange will continue over this connection.

3. When the master server is satisfied, it will contact the **bpcd** daemon on the media server (port 13782) from a reserved port (Master-R1) to initiate the backup. The master server will also send a port number for the media server to call back on.

4. The media server duly calls back from a reserved port (Media-R1) to the reserved port number given it in step 3 (Master-R2). Each backup job maintains its own connection in this way.

5. The media server connects to the client from a port which may be reserved or non-reserved depending on client configuration. The connection is media server "Media-U1" to client port 13782, the **bpcd** daemon. The media server will issue a port number for call back (Media-U2.)

6. The client connects back using the same class of port that was used to contact it in step 5 (call it Client-U1) to Media-U2. The actual data to be backed up begins to flow.

7. If media multiplexing is being performed (a method of spooling multiple backups to the same tape at the same time), an additional connection is created for each data stream. Client-U2 to Media-U3 is born, and possibly some siblings.

8. During the backup, the media server will query the Volume Manager Daemon (**vmd**, not started from **inetd**) on the master server at port 13701 to select the appropriate tape(s). The connection will initiate from a non-reserved port (Media-N1.)

9. The media server will also talk to the master server's **bpdbm** process on port 13721 (not started from **inetd**) from another non-reserved port (Media-N2.)

10. Log entries to the client will be sent from the media server using the **bpcd** approach – Media-U3 to client 13782.

11. And of course the client will call back from Client-U2 to Media-U4 to establish a permanent channel.

This gives a grand total of 11 different connections using 21 different tcp ports on three different servers (Veritas[1], App. C). The restore case has two additional connections, one on a non-reserved port. Only one of these ports is actually started from **inetd** (the **bpcd** port on the media server) and we have already placed it under the control of TCP Wrappers. The other full time daemons are available for connection from anyone all the time. A number of the other connections are to ephemeral ports that are only opened during a backup. But a backup may continue for hours, and there might be multiple backups occuring at any given time. We would prefer not to have all of this visible to anyone who can "see" our server.

What can we do to tighten this up? NetBackup has the ability to allow any client to perform a user directed backup or restore. As a matter of policy, we will only allow backups or restores to be initiated from the backup server. Not only is this a good idea from a philosophical standpoint, it will also allow us to close off direct calls to the Master server's unwrappered **bprd** port from the client. Steps 1 and 2 are disposable.

WARNING: Closing off <u>all</u> communication between the Master server and the client will break the following:

- backups using the NetBackup database extensions
- progress logs
- user directed backups and restores

- e-mail notification from clients
- automatic discovery of multi-streaming information from clients
- automatic software updates to clients

If you need any of these features you will need to enable this channel.

Also to our advantage is that we have only a single Master/Media server in our current configuration. This means that by allowing the server to talk to itself (although it does not use the loopback address to do this) we have implicitly allowed all Master to Media server communication. Steps 3, 4, 8 and 9 are now accounted for.

This leaves at issue only the ports that the client needs to talk to the media server and vice versa. The media server needs to contact the clients on port 13782 (**bpcd**) from some port range and to be called back. While NetBackup can use both reserved and non-reserved ports for this, it turns out that the default setting is to use reserved ports, specifically tcp ports 512-1023. Further, there is a sparsely documented way to narrow this window (Veritas[2]).

In the NetBackup configuration file `/usr/openv/netbackup/bp.conf` place the following directives:

```
CLIENT_RESERVED_PORT_WINDOW=923 1023
```

This setting specifies the range of ports the server will call out from.

```
SERVER_RESERVED_PORT_WINDOW=923 1023
SERVER_PORT_WINDOW=4950 5000
```

This setting specifies the range of ports the server will accept incoming connections on. These are the ports it will tell clients to call back on.

Should you need to enable master server to client communications, you can add

```
CLIENT_PORT_WINDOW=4950 5000
```

This is the range of non-reserved ports the master server would call back from in step two of our connection outline.

```
# cat <<EOF >>/usr/openv/netbackup/bp.conf
> CLIENT_RESERVED_PORT_WINDOW=923 1023
> SERVER_RESERVED_PORT_WINDOW=923 1023
> SERVER_PORT_WINDOW=4950 5000
> #CLIENT_PORT_WINDOW=4950 5000
> EOF
#
```

Please note that the `CLIENT_RESERVED_PORT_WINDOW` and `SERVER_RESERVED_PORT_WINDOW` directives are also required in the `/usr/openv/netbackup/bp.conf` file of all clients. Having identified exactly which ports we need to pass traffic through, we can move on to our next task.

IP Filter Configuration

IP Filter is a powerful and lightweight packet filtering tool which can be used to build a production firewall. Here we will just be using it to lock up our backup server. For those not familiar with IP Filter, there is a fine tutorial available at www.obfuscation.org/ipf/ipf-howto.html.

The first thing to do is decide what we want to let through our firewall. Everything not on the list will be dropped. SSH is needed, so traffic inbound to tcp port 22 and outbound to tcp port 22 will be allowed. Sendmail traffic outbound to tcp port 25 will be permitted. Outbound DNS queries on tcp/udp 53 are allowed. NTP on upd port 123 is required. We may want ICMP for trouble shooting.

Finally the backup software needs to do its job. Connections in the tcp port range 923-1023 and 4950-5000 to the backup clients' net block are needed as are outbound connections to port 13782. We will take the approach of limiting outbound traffic as well as inbound.

We installed the IP Filter software earlier and all we have to do now is create a configuration file and tell **ipf** to read it. The configuration file is in `/etc/opt/ipf/ipf.conf` and is empty. A small but functional `ipf.conf` file is listed in Appendix A. It is primarily focused on the specific task of allowing NetBackup to do its job because this is the concern of this document. There are more steps one could take to limit exposure to network attack, log suspicious activity, and hide the fact that filter software is running.

There are a number of **ipf** tools and cryptic command line arguments. But if you are new to IP Filter, the easiest way to have your configuration file read is to issue the command:

```
# /etc/init.d/ipfboot reload
```

If you haven't connected your client host to the stand-alone network, now would be a good time. It can be used to test that the `ipf.conf` file is actually doing what you want.

AIDE Configuration

AIDE will tell us what files have changed since we last updated its database. This is both useful for discovering intruder fingerprints and handy for figuring out what configuration change just broke the backups or what files that last patch cluster changed.

Although we have installed the AIDE package in `/usr/local/`, we won't be running it from there. Copy the aide binary to `/opt` and create a configuration file for it.

```
# cd /opt
# cp /usr/local/bin/aide .
# vi aide.conf
```

An `aide.conf` file to get you started is located in Appendix B. It lists the built-in definitions for the file and a few scanning rules. It checks all of `/usr`, `/sbin`, `/lib`, most of `/etc`, and the NetBackup binaries (which will be regularly invoked as root) located in `/opt/openv/volmgr/bin` and `/opt/openv/netbackup/bin`. Note that NetBackup has a bad habit of placing lock files in its bin directory, so there are a few rules that you may want to update. Finding an optimal `aide.conf` is a personal thing and will require some experimentation, but this example should get you started.

Once the configuration file is created, make a database of the current state of the machine:

```
# ./aide -c aide.conf -init
# mv aide.db.new aide.db
```

You might want to run a backup and restore against your client and reboot the server just to see how the database stands up against those events. A large number of false positives can lead to complacency when reading report output. Then see if anything has changed.

```
# ./aide -c aide.conf --check
```

Copy these files off the server to your client machine. Later you can burn them to a CDROM. AIDE will happily run from the CD and having a read only copy of your binary, configuration, and database means you won't have to worry about them being replaced by a clever hacker.

Some people leave the AIDE CD in the drive and regularly call the binary from **cron**. While this has the benefit of being automated, it also means that once someone has accessed your machine they have a complete list of the files you

are (and are NOT) looking at. If they can tell you never check `/dev/md/shared/1/rdsk`, that is where they are going to hide the rootkit. If you are disciplined, running a periodic check by hand is not that much trouble. You will be inside this rack to change tapes regularly anyway.

Once the AIDE files have been copied off the server, remove them from `/opt`, along with the driver and patch files that have already been used. Leave the CISscan tool in place.

System Backup

The last step to building the backup server is to make an archival backup in case should we ever need to restore to this state. While the purpose of this server is to perform backups, and it will regularly make backups of its disks, we will not use NetBackup for our archival backup. The number of elements that must be working properly to restore from a NetBackup image is significantly higher than the number needed to do a simpler restore. We will use **ufsdump** and **ufsrestore** for this special backup.

Manually insert a tape in the first robotic drive or use the NetBackup command **/usr/openv/volmgr/bin/robtest** to move a tape into a suitable tape drive. Reboot the system to single user mode

```
# reboot -- -s
```

and use **ufsdump** to backup each partition to a new tape. Make sure to use the correct tape device.

```
# mount -a
# mt /dev/rmt/0 rewind
# for sys in / /usr /var /opt
> do
> ufsdump 0f /dev/rmt/0n $sys
> done
# mt /dev/rmt/0 rewind
```

As a check of your backup, perform an interactive restore. Check several directory trees and recover of some random files to test that your image is good.

```
# ufsrestore i /dev/rmt/0
. . .
# mt /dev/rmt/0 offline
```

Remove and write protect the tape and store it in a safe and secure location. For extra insurance against local disaster, make a second tape in the same manner

and store it offsite. Since you should be storing some of your "regular" backups offsite already, it should be easy to add this tape to that storage.

System Maintenance

This server is functional, secure, and ready to do its job. How can it be kept that way? First, take the steps you would take for any server. Regularly check the log files for interesting information related to performance and security. Take advantage of the **sar** data to look for ways to improve system performance and watch for unexpected system behavior. Fine tune your `logadm` settings and consider adding some automation to your log watching. Both swatch and logcheck are widely used for this purpose.

AIDE checks should be run at least weekly. They can be run any time there is a question about the integrity of the system in addition to the regularly scheduled checks. The time of the scan and the result should be recorded in the server logbook. (Hard copy records about your critical systems are a good idea.) Continue to improve the `aide.conf` file and regularly update the read-only media with the `aide.conf`, `aide.db` and **aide** binary. A well-tuned configuration file is a great starting point for the next server you build, no matter what its function will be.

Check for operating system patch updates at http://sunsolve.sun.com at least once every two weeks, and make sure to patch immediately if major security releases are made. In any event, the server should be patched with the latest patch cluster on a monthly basis. Take advantage of any patch automation system you are currently using in your environment to be apprised of patches, even if they are not automatically installed.

Visit the support site for Veritas (http://support.veritas.com) at least once a month for updates on the application software. These updates will not be as regular as the OS patches, nor usually as critical from a security standpoint. But one never knows when or where the fatal vulnerability will come from.

As a security conscious system administrator you should stay abreast of current information regarding known vulnerabilities and solutions to them. The BugTraq mailing list (http://www.securityfocus.com/subscribe/) is a full disclosure mailing list that is often the first to publish new vulnerabilities. The CERT (originally Computer Emergency Response Team) home page also provides a valuable resource at www.cert.org, as does the CERT mailing list (http://www.cert.org/contact_cert/certmaillist.html.) And SANS provides a number of weekly digests that provide useful overviews of the current state of the computer security world. They can be subscribed to at http://www.sans.org/newsletters.

Finally, the server should be viewed by friendly eyes the same way it will be by unfriendly eyes. Regularly portscan your server from the outside to make sure that the face it shows to the network is the one you want it to. NMAP is a great tool for this and will be used in our next section for an initial check. Regular NMAP scans of your entire server network and comparisons with previous scans are a good way to discover problems before they happen and to keep a handle on just what services you are providing (wittingly or unwittingly.)

Many site operating procedures require the periodic movement of some backup media to an off-site location to reduce data loss in a catastrophic event. While collecting tapes for shipment, take the time to check the physical condition of the server and tape library to make sure access is still restricted and that there are no signs of tampering. Also look for loose SCSI cables, pinched network cabling, and other potential sources of future service interruptions. Check the UPSes to which the server is connected for correct functioning and battery charge level.

Checking the Configuration

Before this secure server is placed on the network, it needs to be tested to make sure the hardening procedures have had the desired effect. The first tests will be simple ones, but ones that are often overlooked by new security administrators.

Physical checks

Begin by checking the physical accessibility of the server. Try the doors to the server room. Are all of them really locked? Are the doors on the server rack closed and locked? Is the key sitting in the lock or secured in a location known only to the administrators? Check the case on the server and the door to the tape library as well to see that they are locked.

Open the rear doors on the rack and unplug one of the power cables. The server should continue to run, but a warning message should be sent to the console and logged to `/var/adm/messages`. Feel free to try the other power cable. But not before reattaching the first…

Check the load and battery condition on the UPSes and check the temperature in the server room to make sure it is within specified limits. Once the physical security of the machine has been verified, lock the rack and return the key to its safe location.

NetBackup Check

If backups don't work on this server, all this effort has been for naught. Reboot the server and check that all desired backup daemons start up correctly. The **vmd**, **ltid**, **bprd**, and **bpdbm** processes should all be running. From your client machine, ssh in (verifying ssh access) and perform a test backup and restore on the client. Launch the GUI with "`/usr/openv/netbackup/bin/jnbSA &`" and chose a Manual Backup for the client from the appropriate test policy. Restore a file back to the client after the successful backup.

Remember that the client cannot initiate jobs. Backups and restores must be started from the server. Once the backup and restore functionality has been verified, the security features of the system can be tested.

Although details of NetBackup Policy configuration are outside the scope of this paper, it is worth reminding the reader that the server should be backing up its own disks on a daily basis. Additionally, the "catalogs" should be configured to back up to two alternating tapes after each regularly scheduled backup. The GUI setup wizard will walk you through this process, but while the NetBackup configuration is being checked for completeness, verify that these backups are scheduled.

TCP Wrappers Testing

To test TCP Wrappers, modify `/etc/hosts.allow` such that it does not include the IP address of the test client. Attempt to ssh from the client to the server. The connection will fail with a "`Connection closed by remote host`" message. Correct the `/etc/hosts.allow` file and ssh in. Note that the changes to the configuration of TCP Wrappers take place immediately.

SSH Check

SSH has already been shown to be correctly using the TCP Wrappers configuration. There is another test we would like to perform. Adjust the `/etc/ssh/ssh_config` on the client machine so that it will only use SSH protocol version 1. Make another attempt to ssh into the server. If the server is configured correctly, the connection will fail with a return message of:

```
Protocol major versions differ: 1 vs. 2
```

Additionally, there will be an entry made in the server's `/var/log/authlog` by sshd stating:

```
Did not receive ident string from 192.168.1.201
```

It looks like our SSH service is performing as advertised.

Checking the Logs

Open another ssh connection to the server from the client, but intentionally enter an incorrect password. View `/var/log/authlog` to see the failed (and previously successful) ssh connections. View `/var/log/connlog` to see that processes launched by **inetd** are logging connections. The master server will connect to the media server (itself) on 10-minute intervals, so there is a nice connection "heartbeat."

The File System

Changes were made to the mount options for the various partitions on our server. View the contents of `/etc/mnttab` to see that our options have been used when mounting `/`, `/usr`, `/var`, and `/opt`. Verify that the `/opt` and `/var` partitions are logging and "nosuid." The root partition should be logging and the `/usr` partition is "ro", or read only.

Verify the read-only nature of `/usr`. As the super-user:

```
# touch /usr/testfile
touch: /usr/testfile cannot create
```

Testing AIDE

Insert the AIDE CD with the `aide` binary, `aide.conf` and `aide.db` into the drive and mount. Then check the current state of the system.

```
# mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /mnt
# cd /mnt
# ./aide -c aide.conf --check
```

Now change your password with the **passwd** command, and run the check again. There should be changes in both `/etc/shadow` and `/etc/.pwd.lock`. Note that if you later see changes in `/etc/shadow` but not `/etc/.pwd.lock`, someone has been hand editing your shadow file. If your check does not reveal any changes to the system, go back and adjust your `aide.conf` and verify that aide is actually checking something.

IP Filter and NMAP

The server is almost ready to be placed on the network. We will check its network appearance by running an NMAP scan from the client machine. NMAP is an excellent network scanning tool available from www.insecure.org. A precompiled version is available from www.sunfreeware.com. There are a number of configuration, input, and output options for NMAP which are explained in the man pages (or at http://www.insecure.org/nmap/data/nmap_manpage.html if you prefer.) We will scan both for both TCP and UDP ports.

```
# ./nmap -v -sS -O '192.168.1.200'
```

The full output of the NMAP tcp scan is shown in Appendix C. The only port that NMAP finds is SSH on port 22. Note that the scan correctly identifies the operating system as Solaris 9 with TCP_STRONG_ISS set to 2.

```
# ./nmap -v -sU '192.168.1.200'
```

The udp scan initiated above should return even less information that would be useful to a hacker. NMAP determines that all ports are filtered. Technically, 123/udp (NTP) is not filtered by IP Filter, but by its own rules in `/etc/inet/ntp.conf`.

```
(no udp responses received -- assuming all ports filtered)
```

For completeness, Appendix C also contains NMAP tcp and udp scans run against the host with IP Filter disabled. This shows us what ports are really listening and what our IP Filter setup has done for us. Of primary interest are the seven additional NetBackup daemons that listen on tcp ports 13701, 13711, 13720-13722, 13782-13783. Note, however, that there are no other services visible to the network – no sendmail, no telnet, no rpc services. Without IP Filter, we still have a reasonably secure network face, but with it things are much improved.

On the udp front, an NMAP scan of the unfiltered machine shows only NTP active on port 123.

A final comment: the client from which we have launched the network scans is inside the privileged IP range. This host has the greatest access to the server of any that will be on the network. Hosts outside the privileged IP range will only see SSH under any circumstances, while backup clients may connect to other ports. If it is possible to restrict the IP addresses of the various backup clients into a smaller range than the rest of the internal network, the `ipf.conf` file should be adjusted to take advantage of the smaller privileged range.

CIS scan

Before moving the new backup server onto the production network, the CIS benchmarking tool that was installed earlier will be run.

```
# cd /opt/cdrom/CIS
# ./cis-scan
```

Do not be too concerned about the final score of the scan. Rather, look carefully at the output file to determine if any additional steps should be taken to secure the server. The CIS guidelines are intelligent suggestions for securing a server, but understanding why they are made is more important than a numerical score.

The scan will reveal things about both the server and about the scanner's logic. The particular scanner version used by this author complained about several things being insecure which either were secure or were irrelevant. Particularly, although testing proves that TCP Wrappers is successfully handling connections made via **inetd**, the string that invokes **tcpd** is not present in `/etc/inetd.conf`. (This paper has taken the approach of modifying `/etc/default/inetd`.) The benchmark scanner assumes this means we are not using **tcpd**. Likewise, syslog is configured to listen only to local traffic. But because it was not done via the "-t" flag, the scanner misses it.

Points are also lost for not restricting NFS clients to reserved ports and for lacking `/etc/ftpd/ftpusers`. Given that there are no binaries to run those services and that there is a restrictive IP Filter configuration, our server probably has little to worry about on that front.

Nevertheless, the scanner does provide a useful portrait of the server's configuration. For every item, both positive and negative, the administrator should know why the tool reports as it does.

The last several checks are for functionality, and can only be performed on the network. Take this opportunity to remove any unnecessary files from `/opt`. Be sure not to leave behind the CIS scan results, or the AIDE database or configuration information.

Final Checks

Reboot and place the server on the appropriate network. Check that sendmail is working by sending a mail message. Mail should be correctly routed through the local mail hub and delivered accordingly. After the sever has been up a few minutes, check the NTP service by running **ntptrace**:

```
# ntptrace
localhost: stratum 3, offset 0.000011, synch distance \
0.03529
time.example.com: stratum 2, offset -0.000296, synch \
distance 0.02505
gps.example.com: stratum 1, offset -0.001724, synch \
distance 0.00000, refid 'GPS'
#
```

Also check to see that `/etc/ntp.drift` has been recently updated.

Check that DNS is working by resolving a host that is not listed in the local hosts file. Remember that you may want to list important machines in `/etc/hosts` to avoid potential DNS spoofing attacks. In theory, all clients and administrator workstations could be entered this way, although this step, like all security, comes at the expense of some convenience.

```
# nslookup blackhole
Server:  ns2.example.com
Address:  192.168.1.22

Name:    blackhole.example.com
Address:  192.168.1.111
```

If the above tests work, you are ready to start installing the client software and running backups. Remember to modify the `bp.conf` on all clients and be sure to continue the with the maintenance steps previously listed.

Conclusion

A step by step outline for building a reasonably secure and functional Veritas NetBackup server on Solaris 9 has been presented. By following this recipe, one can create a server that is resistant to loss of availability due to power failure or disk failure. The server will provide recourse when valuable data is lost or

altered, for both itself and for many clients. The use of off-site tape storage will prevent a complete loss of data in the event of a server room catastrophe.

Physical access to the server has been restricted to the responsible administrators. Login accounts have likewise been restricted. This should help reduce both local privilege escalation attacks and local denials of service.

The confidentiality of the information on the server has been improved by the use of TCP Wrappers and IP Filter to make network attacks difficult. Even the backup clients are only trusted to a certain degree.

Finally, additional logging has been enabled and AIDE has been configured on the server to provide information in the event of assaults against the server (either successful or unsuccessful.)

While there are always further steps one can take to make a computer system more secure (and less usable), this procedure should provide a solid starting point for those additional refinements.

References

Christensen, Jason. "Building a Secure Solaris 8 Backup Server." October 2001. http://www.giac.org/practical/Jason_Christensen_GCUX.doc (18 August 2003).

CIS, Center for Internet Security. "Solaris Benchmark v.1.2.0" March 2003. http://www.cisecurity.org (18 August 2003)

Conoboy, Brendan and Fichtner, Erik. "IP Filter Based Firewalls HOWTO." 11 December 2002 www.obfuscation.org/ipf/ipf-howto.html (18 August 2003)

Fyodor. "NMAP Home Page." 25 June 2003 www.insecure.org (18 August 2003)

Garfinkel, Simson, and Spafford, Gene. Practical UNIX and Internet Security, 2nd Edition. Sebastopol: O'Reilly and Associates, 1996.

Guru Labs LC. Solaris Network Intrusion Detection (Course Book). Sun Microsystems, 2001.

Lehti, Rami. "AIDE Manual." http://www.cs.tut.fi/~rammer/aide/manual.html (18 August 2003)

Noordergraaf, Alex and Watson, Keith. "Solaris Operating Environment Security: Updated for Solaris 9 Operating Environment." December 2002. http://www.sun.com/solutions/blueprints/1202/816-5242.pdf (18 August 2003)

OpenSSH Team, "OpenSSH Features." 13 November 2002. http://www.openssh.org/features.html (18 Aug 2003)

Parmar, Harpal. "Building a Cost Effective Syslog Server using Solaris for Intel and Sunscreen Lite." January 2003. http://www.giac.org/practical/GCUX/Harpal_Parmar_GCUX.pdf (18 August 2003)

Pomeranz[1], Hal. Solaris Security Step by Step, Version 2.0. The SANS Institute, 2001.

Pomeranz[2], Hal. SANS Course Book 6.5: UNIX Practicum. The SANS Institute, 2003.

Pomeranz[3], Hal. SANS Course Book 6.2: UNIX Security Tools. The SANS Institute, 2003.

Reed, Darren. "IP Filter home page." http://coombs.anu.edu.au/~avalon/ip-filter.html (18 August 2003)

Sun Microsystems[1]. "Solaris 9 Presskit: Frequently Asked Questions." http://www.sun.com/aboutsun/media/presskits/solaris9/sol9-faq.html (18 August 2003)

Sun Microsystems[2]. "Solaris 9 OE Features and Benefits – Security." http://wwws.sun.com/software/solaris/sparc/solaris9_features_security.html (18 August 2003)

Sun Microsystems[3], "Solaris 9 Sun Hardware Platform Guide" http://docs.sun.com/db/doc/816-1664-05/6m82ltoqt?a=view (18 August 2003)

Sun Microsystems[4]. "Solaris Volume Manager Administration Guide" http://docs.sun.com/db/doc/806-6111 (18 August 2003)

Venema, Wietse. "Wietse's Home Page." http://www.porcupine.org/wietse/ (18 August 2003)

Veritas Software Corporation[1]. Veritas Advanced NetBackup for Solaris (Course Book). Veritas, 2001.

Veritas Software Corporation[2]. "Veritas TechNote 187321." 26 July 2000 http://seer.support.veritas.com/docs/187321.htm (18 August 2003)

Windl, Ulrich. "The NTP FAQ and HOWTO." 25 January 2003 http://www.ntp.org/ntpfaq/NTP-a-faq.htm (18 August 2003)

Appendix A

Example /etc/opt/ipf/ipf.conf file for Veritas NetBackup.

```
#
# IP filter config file
# I use "quick" because I prefer first match and exit.
# Ordered for desired performance.
#
# Don't mess with loopback
#
pass in quick on lo0 all
pass out quick on lo0 all
#
# Netbackup
# (first, let Master talk to Media server *shrug*)
pass in quick from 192.168.1.200/32 to 192.168.1.200/32 keep state
pass out quick from 192.168.1.200/32 to 192.168.1.200/32 keep state
#
# NB client stuff
pass in quick on eri0 proto tcp from 192.168.0.0/16 port 922 >< 1024 to
192.168.1.200/32 port 922 >< 1024 flags S keep state
pass in quick on eri0 proto tcp from 192.168.0.0/16 port 922 >< 1024 to
192.168.1.200/32 port 4949 >< 5001 flags S keep state
pass out quick on eri0 proto tcp from 192.168.1.200/32 port 922 >< 1024
to 192.168.0.0/16 port 922 >< 1024 flags S keep state
pass out quick on eri0 proto tcp from 192.168.1.200/32 port 922 >< 1024
to 192.168.0.0/16 port = 13782 flags S keep state
#
# Uncomment next lines if you need NB Master to client communication
#
#pass in quick on eri0 proto tcp from 192.168.0.0/16 to
192.168.1.200/32 port 13720 flags S keep state
#pass out quick on eri0 proto tcp from 192.168.1.200/32 port 4949 ><
5001 to 192.168.0.0/16 port 13783 flags S keep state
#
# NTP
#
pass out quick on eri0 proto udp from 192.168.1.200/32 to any port =
123 keep state keep frags
pass in quick on eri0 proto udp from any to 192.168.1.200/32 port = 123
keep state keep frags
#
# SSH
#
pass in quick on eri0 proto tcp from any to 192.168.1.200/32 port = 22
flags S keep state
pass out quick on eri0 proto tcp from 192.168.1.200/32 to any port = 22
flags S keep state
#
# Sendmail (outbound only)
#
pass out quick on eri0 proto tcp from 192.168.1.200/32 to any port = 25
flags S keep state keep frags
#
```

```
# DNS
#
pass out quick on eri0 proto tcp from 192.168.1.200/32 to any port = 53
flags S keep state keep frags
pass out quick on eri0 proto udp from 192.168.1.200/32 to any port = 53
keep state keep frags
#
# ICMP (could tighten this up much more)
#
pass out quick proto icmp from any to any keep state
pass in quick proto icmp from any to any keep state
#
# Default deny stance at end
#
block in all
block out all
#
```

END ipf.conf

Appendix B

```
# Evolving AIDE config file
#
# Where is the database we read?
database=file:aide.db

# Where is the database we write?
database_out=file:aide.db.new

#
# Build in definitions
#
#p:     permissions
#i:     inode
#n:     number of links
#u:     user
#g:     group
#s:     size
#b:     block count
#m:     mtime
#a:     atime
#c:     ctime
#S:     check for growing size
#md5:   md5 checksum
#sha1:  sha1 checksum
#rmd160:        rmd160 checksum
#tiger: tiger checksum
#R:     p+i+n+u+g+s+m+c+md5
#L:     p+i+n+u+g
#E:     empty group
#>:     growing logfile p+u+g+i+n+S
#

#file list
/usr R
/lib R
/sbin R
/etc R
=/etc$ L
=/etc/logadm.conf$ L
!/etc/.syslog_door$
!/etc/coreadm.conf$
!/etc/cron.d/FIFO$
!/etc/dumpadm.conf$
!/etc/initpipe$
!/etc/lvm
```

```
!/etc/mnttab$
!/etc/ntp.drift$
!/etc/opt/ipf/ipmon.pid$
!/etc/saf
!/etc/sysevent
!/etc/syslog.pid$
!/etc/utmppipe$
/opt/openv/volmgr/bin R
/opt/openv/netbackup/bin R
=/opt/openv/netbackup/bin$ L
!/opt/openv/netbackup/bin/bpdbm.lock
!/opt/openv/netbackup/bin/bpjobd.lock
!/opt/openv/netbackup/bin/bprd.lock
!/opt/openv/netbackup/bin/*_CALLED
!/opt/openv/netbackup/bin/bprd.d
!/opt/openv/netbackup/bin/bpsched.d
```

END aide.conf

Appendix C

TCP NMAP scan, IP Filter on:

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Host backup1.example.com (192.168.1.200) appears to be up
... good.
Initiating SYN Stealth Scan against backup1.example.com
(192.168.1.200)
Adding open port 22/tcp
The SYN Stealth Scan took 557 seconds to scan 1601 ports.
Warning:  OS detection will be MUCH less reliable because
we did not find at least 1 open and 1 closed TCP port
For OSScan assuming that port 22 is open and port 42131 is
closed and neither are firewalled
Interesting ports on backup1.example.com (192.168.1.200):
(The 1600 ports scanned but not shown below are in state:
filtered)
Port       State       Service
22/tcp     open        ssh
Remote operating system guess: Solaris 9 with
TCP_STRONG_ISS set to 2
Uptime 0.079 days (since Tue Aug 5 14:17:21 2003)
TCP Sequence Prediction: Class=truly random
                         Difficulty=9999999 (Good luck!)
IPID Sequence Generation: Incremental

Nmap run completed -- 1 IP address (1 host up) scanned in
562 seconds
```

TCP NMAP scan, IP Filter off:

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Host backup1.example.com (192.168.1.200) appears to be up
... good.
Initiating SYN Stealth Scan against backup1.example.com
(192.168.1.200)
Adding open port 13722/tcp
Adding open port 13783/tcp
Adding open port 13782/tcp
Adding open port 13720/tcp
Adding open port 13721/tcp
Adding open port 22/tcp
Adding open port 13711/tcp
Adding open port 13701/tcp
The SYN Stealth Scan took 351 seconds to scan 1601 ports.
```

```
For OSScan assuming that port 22 is open and port 1 is
closed and neither are firewalled
Interesting ports on backup1.example.com (192.168.1.200):
(The 1593 ports scanned but not shown below are in state:
closed)
Port       State       Service
22/tcp     open        ssh
13701/tcp  open        VeritasNetbackup
13711/tcp  open        VeritasNetbackup
13720/tcp  open        VeritasNetbackup
13721/tcp  open        VeritasNetbackup
13722/tcp  open        VeritasNetbackup
13782/tcp  open        VeritasNetbackup
13783/tcp  open        VeritasNetbackup
Remote operating system guess: Solaris 9 with
TCP_STRONG_ISS set to 2
Uptime 0.005 days (since Tue Aug 5 10:51:51 2003)
TCP Sequence Prediction: Class=truly random
                         Difficulty=9999999 (Good luck!)
IPID Sequence Generation: Incremental

Nmap run completed -- 1 IP address (1 host up) scanned in
361 seconds
```

UDP NMAP scan, IP Filter off:

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Host backup1.example.com (192.168.1.200) appears to be up
... good.
Initiating UDP Scan against backup1.example.com
(192.168.1.200)
Too many drops ... increasing senddelay to 50000
The UDP Scan took 298 seconds to scan 1468 ports.
Adding open port 123/udp
Interesting ports on backup1.example.com (192.168.1.200):
(The 1467 ports scanned but not shown below are in state:
closed)
Port       State       Service
123/udp    open        ntp

Nmap run completed -- 1 IP address (1 host up) scanned in
298 seconds
```