Global Information Assurance Certification Paper

Global Information Assurance Certification
Certified UNIX Security Administrator

Practical Assignment
Version 1.9 – Revised April 8, 2002

Hewlett Packard UNIX Security Server Lockdown

By Daimian Woznick

Date Submitted: January 20, 2004

Table of Contents

ABSTRACT ..... **I**

SECTION I - INTRODUCTION **1**

SECTION II – SCOPE **1**

SECTION III – SYSTEM SPECIFICATION **1**

HARDWARE SPECS 1
SOFTWARE USED 1
SYSTEM MISSION 2
DATA SENSITIVITY 2

SECTION IV – RISK ANALYSIS **1**

INTRODUCTION 1
OPERATIONAL DESCRIPTION 1
SAFEGUARDS AND COUNTERMEASURES 2
THREAT SUMMARY 2
VULNERABILITY SUMMARY 3
CONCLUSION 4

SECTION V – HPUX INSTALLATION **1**

PREREQUISITES 1
SYSTEM BOOT 1
OPERATING SYSTEM CONFIGURATION 3
Basic Tab *4*
Software Tab *5*
System Tab *6*
File System Tab *8*
Advanced Tab *10*
INSTALLATION TIME 10

SECTION VI – STEPS AFTER INSTALLATION **1**

PATCH INSTALLATION 1
SOFTWARE REMOVAL 2
ADDITIONAL SOFTWARE TO INSTALL 3
SHADOW PASSWORD FILES 5
CURRENT ACCOUNTS 6
NEW ACCOUNTS 7
NETWORK SERVICES 7
TIME ZONE SETTINGS 12
SYSTEM STARTUP 13
/sbin/rc0.d *13*
/sbin/rc1.d *14*
/sbin/rc2.d *14*
/sbin/rc3.d *14*
/sbin/rc4.d *15*
SYSTEM LOG FILES 15
HOST RESOLUTION SETUP 15
SUID AND SGID FILES 16
SYSTEM PROFILE 18

SECTION VII – ETRUST ACCESS CONTROL **1**

INTRODUCTION 1
INSTALLATION 1
ACCESS RULES 3
EXIT SCRIPTS 4
RELATED PROCESSES 5
REPORTING 6

SECTION VIII – UNIX SECURITY AUTOMATED PROCESSES 1

INTRODUCTION 1
INSTALLATION 1
AUTOMATED SECURITY REVIEW 1
AUTOMATED FILE TRANSFER 2
CRON SETTINGS CHECK 2
GROUP AND PASSWORD FILE BACKUP 3
TCB AND PASSWORD FILE COLLECTOR 3
Table 8.1. TCB System Settings 5
Table 8.2. TCB Account Settings 6
SOFTWARE CONFIGURATION REPORTING 9
VALID LOGON ACTIVITY ARCHIVAL 9
INVALID LOGON ACTIVITY REPORTING AND ARCHIVAL 10
SURROGATE ACTIVITY REPORTING AND ARCHIVAL 10
TRUST RELATIONSHIP REPORTING 10
FTP ACCESS RECONCILIATION 11
ROOT ACCOUNT ACCESS REPORTING 11

SECTION IX – AUTOMATED SECURITY REVIEW 1

INTRODUCTION 1
SECTION CONVENTIONS 1
AUTOMATED ACCOUNT ADMINISTRATION CHECKS 1
UNIX Password File 1
UNIX Group File 3
Shadow Password Files 5
Account Home Directories 6
Account Configuration Settings 7
Account Security 8
Login Auditing 11
Surrogate Auditing 11
Security Configuration File 12
AUTOMATED FILE SYSTEM SECURITY CHECKS 12
System Backups 13
SUID and SGID Programs 13
Files Without Ownership 14
Binary and Library Directories 15
Text Editors 15
System Log Daemon 16
Disk Usage 16
World Writable Directories 17
AUTOMATED NETWORK SECURITY CHECKS 17
Banners 17
Modems 18
UNIX to UNIX Copy 18
Trusted Hosts 19
Services File 20
Network Services Daemon Configuration File 21

File Transfer Protocol Service *21*
Telnet Service *23*
Trivial File Transfer Protocol Service *23*
AUTOMATED ROOT ACCOUNT SECURITY CHECKS 23
Root UID *24*
Root Account Access *24*
Login Auditing *25*
Job Scheduling *25*
AUTOMATED JOB SCHEDULING SECURITY CHECKS 26
at *26*
batch *27*
cron *27*

SECTION X – APACHE WEB SERVER CONFIGURATION 1

INTRODUCTION 1
CONFIGURATION 1
httpd.conf *1*
ssl.conf *4*
SSL Certificates *6*
APACHE WEB SERVER STARTUP 8
MOD_PERL PROBLEMS 9
PASSWORD SYNCHRONIZATION 9
LOG MONITORING 9
REPORTS ON THE WEB SITE 10
HTML Formatted Reports *10*
Implementation on the Group Web Site *11*

SECTION XI – FILE SYSTEM INTEGRITY CHECKS 1

INTRODUCTION 1
PURPOSE OF APPLICATION 1
WHY IN-HOUSE DEVELOPMENT? 2
DATA REPOSITORY 2
APPLICATION USAGE 3
METHODS OF ENSURING INTEGRITY 3
FILE MONITORING 4
WHAT CHANGED? 4
REMOTE CONNECTIVITY 5
Application Account *5*
Secure Shell *6*
eTrust Access Control *6*
ADMINISTRATION 7
Manual Modifications *8*
Command Line Menu System *8*
Automated Check Process *9*
Web Site Reporting *9*
IMPORTANT NOTES 10

SECTION XII – OTHER PROCESSES 1

SECURITY PATCH CHECKS 1
ACCOUNT ADDITIONS 2
ACCOUNT DELETIONS 2
INACTIVITY CHECKER 2
QUICKER WAY TO FIND FILES 3

SECTION XIII – CHECKING CONFIGURATION 1

ROOT ACCOUNT ACCESS 1
FILE INTEGRITY CHECKING 1
PASSWORD HISTORY 1
DISABLING OF THE TELNET SERVICE 2
FILES WITHOUT OWNERSHIP 2

ATTACHMENT 1 – REFERENCES 1

Abstract

The Hewlett Packard version of UNIX, named HPUX, has been growing in market share and is one of the best variants of UNIX when it comes to security. In this document we will review what it takes to secure a HPUX server that contains some very sensitive data. This document also delves into the securing of an additional tens to hundreds of servers while saving yourself time. So come on in, kick off your shoes, and have a good read.

Section I - Introduction

This document was written for the securing of an HPUX server that will be used by the UNIX security group responsible for the security of two hundred UNIX servers on the network. The document details the steps necessary on installation and then goes on to what processes to setup to perform the initial security setup of the system and to routinely check the configuration settings.

The target audience for this document is UNIX security or system administrators that are looking for assistance with the securing of servers. Although the subject of this document is a central security server, there are many security tips and tricks included for those that are on a tight budget.

This document encompasses the used of one commercial product other than the operating system. This product is Computer Associate's eTrust Access Control. There are some outstanding uses of this product included in this document but mostly everything can be done without it.

Section II – Scope

This document was created to document the needed steps to secure an HPUX server. Application of the items found in this document will allow the server to obtain an effective security level. This document details security from a HPUX viewpoint. This document details known vulnerabilities and steps to resolve them or limit the impact in the event the vulnerability is exploited.

The compilation of information included in this document is compiled from the experience of this UNIX security analyst and is on the necessary steps to secure a HPUX server. This document also compiles information found in many sources, which are listed in the References section of this document. For more detailed information consult these references.

This document is meant to be a living document that is to be updated when new information on exposures and vulnerabilities are available.

Section III – System Specification

The system specification is identified to allow the focus of what the system is and what its uses will be.

Hardware Specs

The server that you decide to use for the function of administering the security of your environment is entirely dependent upon the size of your company, how many servers you support, and of course how fast you want your processes to complete.

Although you probably could use an old server that is out of use, I would suggest a server with at least two processors and plenty of disk space so you are not always archiving files.

To get a better feel for a system, the following system parameters will be used for this paper:

Model	9000/800/L3000-6x
CPU Type	PA-RISC 2.0
CPU Speed	650 MHz
Number of CPUs	2
Physical Memory	2048 MB
Number of Physical Disks	4
Total Disk Space	138928 MB

As you can see by the above table, this system is very far from the top of the line but will work very well for our purposes. If this server is too large for a specific environment and the cost can not be justified, a smaller server will work as well. The main consideration for a server will be the disk space because most of the processing will occur in off hours. So you can go to one processor and half the disk space listed above. As I said earlier this is dependent on the environment.

Software Used

More important than the hardware used, is the software that is installed on the server. If you decide on getting the most "beef" available in the server it will amount to an extreme waste of resources if you are unable to use it. The following is the software will be need to purchase to make the system used for this paper.

OS Version	Hewlett Packard UNIX (HPUX) B.11.11 (11i) 64 Bit
Security Application	Computer Associate's eTrust Access Control

The eTrust Access Control software was used for this paper because I have been working with it for a while now. There are plenty of third party applications on the market and you could make a case for many of them. Like all the other applications, there are good and bad things about all third party applications and of course, none will solve everything for you (that is of course why we get paid to do what we do).

System Mission

The UNIX Security group for a big company already has a central server to handle the security processing of the approximate two hundred UNIX servers on the network. Some process or another is constantly utilizing this primary server and it provides the ability for other groups, such as the Help Desk, to perform security changes like password changes and resetting of accounts that have been disabled. Security check processing is also accomplished on this system so the processing is not done on the remote servers where it may affect the applications that allow the company to function. This central server also provides the central administration point for our third party security application, Computer Associate's eTrust Access Control.

The UNIX Security group now requires a server to allow them to run a web site that will allow the reporting and analysis of system information for all the UNIX servers in the environment. This server will also be utilized for additional security processing but the main use will be the hosting of the web site. The web site will ease the review of the reports and log files while also making an archive of the files that will make it easier to review old events.

Data Sensitivity

Since the mission of this system has been clearly defined, we can state that the sensitivity of the data on this server is extremely high. Since the data for the security of all the other systems in the environment is collected on this server you have to build a really big fence to keep the neighbors out. This data won't let an attacker get in right away, but with the right analysis of the data, a security vulnerability may present itself while allowing the attacker to get the "keys to the kingdom".

Now that we know that the data on this server is very important we now must look at the next section to analyze the risks.

Section IV – Risk Analysis

This section is used to analyze the risks to the system so we can determine the proper security functions to complete so all the risks will be mitigated.

Introduction

A risk analysis report is used to identify the risks, the countermeasures leveraged against those risks, and the residual risks encountered on the system. This analysis is conducted to consider the necessity and sufficiency of current security measures.

All assessments are based on processing history and subjective judgement.

Operational Description

The following is a description of the operation of this system.

- No customer data will be processed on this server.
- No employee data will be processed on this server with the exception of UNIX account information.
- The server resides in the computer operations room, which is a controlled area that is manned 24 hours a day, 7 days a week.
- The Local Area Network (LAN) software contains security utilities, which help prevent unauthorized users from gaining access to systems.
- Power protection devices, including a generator, are installed.
- Preventive maintenance is performed on all system components on an as needed basis.
- System software is thoroughly tested before installation.
- Checks of system operation to ensure reliability are performed periodically.
- The appropriate number of support staff has been hired, are properly trained, and have undergone an extensive background check.

- All buildings with connections to the LAN are controlled areas that are either locked or manned 24 hours a day, 7 days a week.

Safeguards and Countermeasures

The following are some of the safeguards and countermeasures that have been implemented for the system.

- Entry to the computer operations room is tightly controlled by building security.
- All main buildings for the company are protected by the building security departments.
- All backups are stored in controlled areas.
- Contingency plans are in place and well documented in the event of a disaster destroying the main processing center.
- UNIX Security analysts are hired to implement necessary safeguards on the Operating System.
- UNIX Security analysts are trained in Information Security controls and auditing.
- Before hiring, the company performs a standard background check.
- All available security patches and upgrades are implemented directly following testing.

Threat Summary

The system is located within an office environment and is subject to a range of normal threats that fall into the four categories below.

- **Natural Threats** – These threats include damage caused by earthquakes, high winds, tornadoes, floods, etc.
- **System Threats** – These threats include damage from fire and water, and malfunctions due to structural and electrical problems.
- **Unintentional Personnel Threats** – These threats include data destroyed inadvertently or lost due to inadequate file controls, data destruction due

to the mishandling of media or lack of regular maintenance, and unintentional introduction of a malicious program.

- **Intentional Personnel Threats** – System or data vandalism by disgruntled employees, theft or disclosure of sensitive data, and intentional introduction of a malicious program.

Vulnerability Summary

The following is a summary of the controls in place and the assessment of risk for each of the four categories of threats defined above.

- **Natural Threats** – The historical trend of natural disasters affecting the processing center is minimal.

  Assessment – Low

- **System Threats** – Since the system resides in a controlled atmosphere designed for computers and the building itself is powered by a generator in the event power is lost, this threat is minimal.

  Assessment – Low

- **Unintentional Personnel Threats** – Since employees are trained and this system will only have administrator level user accounts, this threat is minimal.

  Assessment – Low

- **Intentional Personnel Threats** – The company performs background checks on employees before hiring which reduces this risk a little.

  Due to the data processed on this system an employee of the company may attack the system to acquire additional access to other systems on the network. This risk is mitigated by disabling network services, limiting the access to the system, and security analysts reviewing logs for abnormal activity on a regular basis.

  Attacks from the outside of the LAN (i.e. outside the firewall) are minimal because of the firewalls and network intrusion detection systems in place.

  Assessment – Low

Conclusion

The residual risk of processing sensitive information on the system described in this analysis is low.

Section V – HPUX Installation

This section details the installation of the operating system on the server. The installation procedures for installing an operating system have become much easier over time. This is evident in the options you can select in the installation program. You can install with a guided installation that will walk you through many steps or go through the more advanced options. Even though the advanced options still do not give full control, we will use the options in the following instructions.

Prerequisites

The following information should be kept available for the installation. Take note that there are more options that could be used but these instructions are limited to what will be used during this installation.

1. **Server Name** – Know what your server will be named.

2. **IP Address** – Know the IP address that will be assigned to this server. Another option is to use your Dynamic Host Configuration Protocol (DHCP) to get this data but for these instructions we will already obtain the IP address.

3. **Subnet Mask** – Know the subnet mask that will be used.

4. **Gateway** – Know the IP address of the gateway that will be used.

5. **DNS Server** – Know one or two IP addresses that will be used for Domain Name Server (DNS).

6. **File System Configuration** – Know the amount of disk space you have available and map the mount points that you wish to create. Take note that all the disk space does not have to be utilized.

System Boot

The following installation instructions are accomplished at the system console.

1. Insert disc one of the installation discs into the compact disc (CD) drive. This disc is labeled with the OS version, HPUX 11i version 1, the disc contents, core os installation and recovery version B.11.11, and the published date, June 2003.

2. Reboot the server.

3. During the boot sequence the system will ask for interaction. There will be ten seconds for you to press any key to interrupt the boot. Press any key when the following message appears:

   Processor is booting from first available device.
   To discontinue, press any key within 10 seconds.

4. The main menu will now appear on the screen. We need to search the available devices on the server for any devices that have bootable media, i.e. media that contains the necessary instructions to perform an Initial Program Load (IPL). Enter the SEA command for searching and the IPL option.

   Main Menu: Enter command > SEA IPL

   Depending on the system speed and the number of devices connected, this may take a few minutes.

5. Another menu such as the following will now appear showing the devices that can be used.

Path Number	Device Path	Device Type and Utilities
P0	IDE	CD-532E-B IPL
P1	FWSCSI1.6.0	Quantum Atlas 10K-18LVD IPL

   Main Menu: Enter command >

   At this command prompt we select the CD drive that is connected to the server. In the above example we will select P0 along with the boot command.

   Main Menu: Enter command > BO P0

6. The server will now read the contents of this disc and will prompt if there is to be any interaction with the IPL. We will select no.

   Interact with IPL (Y, N, Q)?> N

7. The server will load all the necessary files to perform the installation.

8. If there is a keyboard connected to the Universal Serial Bus (USB) the system will prompt for a language selection to be used. We will enter in the option for English at this prompt. The option that will be used for the selection is: 26) USB_PS2_DIN_US_English.

   Enter the number of the language you want: 26

9. The menu for the installation will now appear which takes us to the next set of instructions.

Operating System Configuration

Use the tab to move the cursor and the space bar to select the option at this set of menus.

1. The main menu will allow the selection of a system recovery or an installation of HPUX. We will use the installation selection.

   Install HP-UX

2. At the User Interface and Media Options menu we get the options to select where the installation files are and the interface for installation. We will use the CD for installation and will select the advanced installation so we can configure more of the OS. The menu will look like the following and then we will select OK.

   Source Location Options:
   [*] Media only installation
   [ ] Media with Network enabled (allows use of SD depots)
   [ ] Ignite-UX Server based installation

   User Interface Options:
   [ ] Guided Installation (recommended for basic installs)
   [*] Advanced Installation (recommended for disk and filesystem management)
   [ ] No user interface – use all defaults and go

3. The Media Installation Selection menu will now appear and will give the option to perform a complete installation off the CD or allow the recovery off of a tape. We will select the complete installation off the CD. The second option available is useful if you need to boot from a tape but your system will not let you. The menu will look like the following and the OK will be selected.

   [*] CD/DVD Installation
   [ ] Boot from CD/DVD, Recover from Tape

4. The main configuration window will now appear. The heading of the window will show the command being used, /opt/ignite/bin/itool. This window is divided into tabs because there are a lot of configuration settings that can be changed. The tabs are Basic, Software, System, File System, and Advanced. The following set of instructions will go into detail on each tab.

Basic Tab

1. **Configurations** – Leave at default.

   HP-UX B.11.11 Default

2. **Environment** – Leave at default.

   HP-UX 11i Base OS – 64 Bit

3. **Root Disk** – Leave as default unless you want change which hard drive will be used as the primary disk.

   QUANTUM ATLAS 10R-18LVD, 10/0/15/1.6.0

4. **File System** – The file system type that you wish to use. For this installation we change the default of Logical Volume Manager (LVM) with VxFS to use the Veritas Volume Manager (VxVM) with VxFS. This is the base journaled file system for HP-UX and is free with version 11i of the OS. We use this file system type to get higher performance and a higher level of availability.

   VERITAS Volume Manager (VxVM) with VxFS

5. **Root Swap** – This option can be increased but for this installation we will keep the default.

   1024

6. **Physical Memory** – This option contains the amount of physical memory on the server.

   2048 MB

7. **Languages** – This option will open the Languages window. This window allows the selection of the languages that will be installed. All the options will be kept as the default with the exception of the following:

Global à **change to NO**
English à **change to YES**

8. **Additional** – This option brings up the Additional Configuration Controls window that displays further settings that can be changed. We will leave the configuration settings at the default. The default settings are as follows:

   Create /export volume à **NO**
   Create separate volumes (/usr, /var, …) à **YES**
   Secondary Swap Space (KB) à **0**
   # of discs in root VG… à **1**
   Force Ignite-UX autoboot?… à **YES**
   Disable DHCP?… à **NO**
   Save patched files?… à **YES**

Software Tab

This tab allows the selection of software that will or will not be installed along with the OS. The selections on the left are based on the category. The category All displays all the other categories. We will not use this category, but instead will go to each category listed separately.

1. **Ordered Apps**

   The following software selections will be **left on**:

Base-VXVM	Base VERITAS Volume Manager Bundle 3.5 for HP-UX
FDDI-00	PCI FDDI
FibrChanl-00	PCI/HSC FibreChannel
GigEther-00	PCI/HSC GigEther
GigEther-01	PCI/PCI-X GigEther
Iether-00	PCI Ethernet
RAID-00	PCI RAID

   The following software selections will be **turned off**: Although it is convenient to install the additional software during the OS installation, we will download and install the required software later to ensure that we have the latest releases installed on the server. This step will ensure that all the newest features and security fixes are included.

B8111AA	Java 2 RTE for HP-UX (700/800), PA1.1 & PA2.0 Add-on
B9098AA	Java 2 Plugin for HP-UX (700/800)

B9789AA	Java2 1.3 RTE for HP-UX
MOZILLA	Mozilla 1.2 for HP-UX
MOZILLAsrc	Mozilla 1.2 Source Distribution
T1455AA	Java2 1.3 Netscape Plugin
hpuxwsApache	HP-UX Apache-based Web Server
hpuxwsTomcat	HP-UX Tomcat-based Servlet Engine
hpuxwsWebmin	HP-UX Webmin-based Admin
hpuxwsXml	HP-UX XML Web Server Tools
perl	Perl Programming Language

All other software selections will be left at the default of OFF.

2. **HPUX Additions**

The following software selections will be **left on**:

BUNDLE11i	Required Patch Bundle for HP-UX 11i, June 2003
FEATURE11-11	Feature Enablement Patches for HP-UX 11i, Sept 2002
General Patches	Mark to load all patches, unmark for just critical and HW patches
HPUXBaseAux	HP-UX Base OS Auxiliary
HWEnable11i	Hardware Enablemnet Patches for HP-UX 11i, June 2003
OnlineDiag	HPUX 11.11 Support Tools Bundle, June 2003

The following software selections will be **turned off**:

B6848BA	Ximian GNOME 1.4 GTK+ Libraries for HP-UX 11.00 and 11i

All other software selections will be left at the default of OFF.

3. **Uncategorized**

There are no selections to be made in the uncategorized category.

System Tab

This tab allows the assignment of additional parameters for the server. These parameters include the setting of the time, networking configuration, and root account password assignment.

1. **Final System Parameters** – Leave this setting at the default of "Set parameters now" because we will set these configuration settings now.

2. **Hostname** – Enter the host name. This is the name of the server.

3. **IP Address** – Enter the IP address that will be used for this system.

4. **Subnet Mask** – Enter the subnet mask that is used on the segment of the network the server will be plugged into.

5. **Time, Day, Month, Year** – Set the current date and time.

6. **Set Time Zone** – This option will open another window that will allow the selection of the time zone where the system is located.

7. **Network Services** – This option will open another window that will contain the following tabs:

    a. **Static Routes** – For this installation we will not assign any static routes.

    b. **DNS** – This tab allows the assignment of the DNS settings. We will enter in two DNS servers in case the primary server is not running. As a general rule, always have at least two servers that are operational acting as DNS servers.

        1) **Domain Name** – Enter your domain name on the network here.

        2) **DNS Server IP Address** – Enter the IP address for the DNS Server and then select the "Add" to set the value.

    c. **NIS** – This option will be left empty in this installation. Due to a lot of security concerns, it is better to not elect to run this service on any HPUX server.

    d. **XNTP** – If your network has a NTP server, enter the IP address here. In this installation we will leave this field blank.

8. **Set Root Password** – When selected, a new window will appear that will allow the selection of the password for the root account. Be sure you remember this password, it is very important. As a security note, this password should be a mixture of capital and lower case alphabetic characters, numeric characters, and special characters (as a general rule,

stay away from the @ and the # because these characters have special meaning at the logon prompt).

File System Tab

This tab allows distribution of disk space across the mount points we will create. The installation process already creates the main mount points that we will use but there are additional mount points that we will create here. Using this tab to setup additional mount points and increase the size of the default mount points will allow us to save time and reboots when doing it later.

It is important to note that after you modify a mount point you must select Modify to make the changes take affect. When adding a mount point ensure that you select Add.

Another important note to take is that you are not required to use all the disk space available. You could use the rest later when you notice that some of the file systems may be too small.

1. **Default File Systems** – The following are already setup by the installation process. All items that are not identified are left at the default setting.

    a. **/stand** – No changes are required for this file system.

    b. **/** - Increase the root file system to 300 MB. This file system should not contain anywhere close to 300 MB if the server is setup properly but someone may dump a core file there one day that could be very large or someone could copy a very large file to the system under this mount point. We increase the file system to ensure this will not cause a problem later.

        - **Change Size to 300 MB**

    c. **/tmp** – Increase the temporary files file system to 300 MB. This file system is usually used by users or applications for files that are not critical or for a working area. We set this file system large to ensure it will not impact the root file system.

        - **Change Size to 300 MB**

    d. **/home** – Change the default of Range MB to Fixed MB because we will assign a specific amount of disk space of 300 MB to it. This file system is used to store account configuration files and is also used by users to store their working files. We set this file system large

for the same reason of ensuring it will not impact the root file system in the event all the space is utilized.

- **Change Range MB to Fixed MB**
- **Change Size to 300 MB**

e. **/usr** – Change the default of Free % to Fixed MB because we will assign the specific amount of disk space of 1500 MB to it. This file system is used for applications and other software installed on the server.

- **Change Free % to Fixed MB**
- **Change Size to 1500 MB**

f. **/opt** – Increase this file system to 1000 MB. This file system is also used for applications and other software that is installed on the server.

- **Change Size to 1000 MB**

g. **/var** – Change the default of Range MB to Fixed MB because we will assign the specific amount of disk space of 1500 MB to it. This file system is used for the logging activities that occur on the server. Take note that the system crash files, which could be extremely large, are also located here.

- **Change Range MB to Fixed MB**
- **Change Size to 1500 MB**

2. **Additional File Systems** – The following file systems are new and will be setup specifically for this server.

a. **/usr/seos** – This file system will be used for the security application, eTrust Access Control. We will set this file system as Fixed MB with 252 MB of space.

- **LV Name** à **seos**
- **Size** à **Fixed MB – 252 MB**
- **Mount Dir** à **/usr/seos**

b. **/opt/hpws/apache/htdocs** – This file system will be used for the web site that will be hosted on the server. We will set this file system as Fixed MB with 4000 MB of space.

- **LV Name** à **htdocs**

- **Size** à **Fixed MB – 4000 MB**
- **Mount Dir** à **/opt/hpws/apache/htdocs**

c. **/audit** – This file system will be used to store any log data that we decided to archive. Keeping this data on a server will allow the quick research through the data instead of restoring from an archive on another form of media such as tape, CD, or DVD. We will set this file system as Fixed MB with 4000 MB of disk space.

- **LV Name** à **audit**
- **Size** à **Fixed MB – 4000 MB**
- **Mount Dir** à **/audit**

Advanced Tab

This tab will be left at the default of nothing.

Installation Time

1. All the configuration has now been completed so select the GO! option.

2. If there is currently data on the hard drive the system will prompt to inform you that all data on the disk will be destroyed. Select the GO! option again to continue.

3. The installation process will prompt for you to replace the CD in the drive. All three discs will be used for this installation.

4. The installation process will run through reboots and further configuration. The process will display OK in banner format and then will reboot the final time.

5. If you are using a USB keyboard the system will prompt again for the language. Use the same option as before.

6. If using the Common Desktop Environment (CDE) at the console you may have a problem with the interface starting. The problem is most likely that the server cannot resolve its own host name. This is because, by default, the server tries resolving to the primary DNS server. To fix this problem, create a file name /etc/nsswitch.conf with the following line in it:

   hosts: files[NOTFOUND=continue TRYAGAIN=continue] dns

Section VI – Steps After Installation

This section details the tasks that are to be completed right after the OS is installed.

Patch Installation

These instructions still require you to remain at the console.

Installing the newest patches is always a good practice but when some patches may be critical that you install. To install the newest releases of the patches we can either download them or we can utilize the Support Plus CD that HP publishes. For this installation we will use the CD labeled Support Plus, HP-UX 11i version 1, September 2003. The following are the steps we use to install the patches off the disc.

1. Read the instructions included with the disc and heed all advice and warnings in the booklet.

2. Place the CD mentioned above in the CD drive.

3. Mount the CD to a file system on the server.

    a. Create a directory to mount the cd

        /usr/bin/mkdir /cdrom

    b. If you do not know the device file for the CD drive, use the ioscan utility to find out.

        /usr/sbin/ioscan –fnC disk

    c. Mount the CD to the directory created. Use the device file found in the output of the above command.

        /usr/sbin/mount –r /dev/dsk/c0t0d0 /cdrom

4. We will install the Base Quality Pack and the Applications Quality Pack. The Base Quality Pack provides all the stable patches available for the OS, graphics, and network drivers. The Applications Quality Pack provides all the stable patches for the applications installed along with the OS. Fortunately HP has combined both of these bundles into

one called GOLDQPK11I. Use the swinstall utility to install this combined bundle.

/usr/sbin/swinstall –s /cdrom/GOLDQPK11I

5. The above step will bring up the interactive session so we can easily identify what patches will not be installed. Mark both the GOLDBASE11i and the GOLDAPPS11i bundles for installation. If interested you can look inside the bundle to see all the patches that will be installed.

6. Install the marked bundles by selecting Actions on the menu bar and the selecting Install.

7. Review the log files if there were any errors during the installation. Select Done when you are completed with the review.

8. When the system prompts, reboot the system and then all the patches will be installed.

Software Removal

We are now done installing from the discs provided by HP. Now we will go in and determine what was installed that should not have been. This is easier to accomplish through the sw interface. Execute the **/usr/sbin/swremove** command and review the software that is currently installed. In the event that you find software that does not belong mark it for removal. Along with the main bundles found, traverse down into the bundles to see what software was included in the bundle. You can always experiment with marking applications for removal. The swremove utility will identify the other software applications that may be affected by the removal. Even if you want to remove a specific application, like the Network File System (NFS), it may not be feasible if there are other applications that may not work properly. In this case we will have to settle with disabling the application. The following is the software removed from this installation:

HPUXBase64

Asian-Core	Asian Core
Asian-PRINTER	Asian Printer
Asian-TERM	Asian TERM
Asian-UTILITY	Asian Utility
UUCP	Unix to Unix Copy

When complete, select Actions on the menu bar and then select Remove. Depending on the software you are removing, you may need to reboot the server again. As always, review the log file and fix any errors that may have been encountered.

Additional Software to Install

During the installation of the OS we elected not to install additional software that we will be using. We did this so we can acquire the newest releases of the software. This ensures we have all the latest security fixes and the newest version of the software.

To acquire the additional software that we will be using, open a internet browser window to http://software.hp.com. This we page is the software depot home page that is hosted by Hewlett Packard. You can either use the search function or traverse through their pages to find the appropriate software packages. Download the following software from a workstation since we removed the browser applications on this server. We can then utilize the File Transfer Protocol (ftp) service to transfer the files to the server.

Hewlett Packard provides the following software that we will be downloading at no cost.

The following are the software applications that we will download along with a brief explanation of what they will be used for. We will transfer the depot files to the /tmp/depots directory for the below examples.

1. **Perl** – This programming language package provided by HP and Active State was optimized to run on the HPUX platform. We will download version 5.6.1. The reason why we download the older version of the programming language is because we are going to use mod_perl with Apache web server application. This application has dependency in the mod_perl functionality in the web server that requires the library files for the older version.

   We will install the Perl programming language so we can support the current processes that were coded in the language and so we can create new processes. Perl has many uses and is a lot quicker than the standard shells provided by the OS.

   **/usr/sbin/swinstall -s **
   /tmp/depots/ perl_B.5.6.1.E_HP-UX_B.11.11_32.depot perl

2. **Strong Number Generator** – HP provides an application that creates the /dev/random and /dev/urandom files similar to the ones that currently exist on the Linux OS. This application provides true random numbers that cannot be reproduced. The version number we will use is B.11.11.07 and the product number is KRNG11I.

   Before HP released this number generator, we were forced to write a program internally to acquire a random number from various statistics of the server. This method is not completely secure so we utilize HP's package. We will use the generator for the encryption in Secure Shell and for the Secure Socket Layer on the web server.

   This application requires a reboot after the installation so we use the –x option and set autoreboot to true. This is necessary because the command line interface does not, by default, allow the installation of software that requires a reboot.

   **/usr/sbin/swinstall -x autoreboot=true -s **
   **/tmp/depots/KRNG11i_B.11.11.07_HP-UX_B.11.11_32+64.depot **
   KRNG11i

3. **Apache Web Server** – The Apache web server provided by HP is the same packages as you can download and compile from the source code. However HP provides the compiled packages and ensures that it will work with its OS. HP also modifies the packages a little to provide better performance and reliability with its platform. For this example we will download version 1.0.10.01 which was released on November 18, 2003. The product number is HPUXWSATW101001. HP has included the Apache Server, Webmin, Tomcat, and XML Tools with their suite.

   We will use this web server to provide a secure repository for reports that will be generated by the various security processes in the environment. This repository will also have an archive system made into it so we can easily review the current reports along with the old reports.

   Although the web server comes complete with Apache, Tomcat, and Webmin, we will only install what we need. This installation will only be for the Apache application. You should always install the minimum amount of software for a lot of reasons, the most important being that less security exposures will affect the server if most of the applications have been removed.

   **/usr/sbin/swinstall -s **
   /tmp/depots/HPUXWSATW-A101001-1100PA.depot hpuxwsApache

4. **Secure Shell** – HP provides the Secure Shell package that is based on the OpenSSH product that is freely available. Again we install the HP package for reliability and performance. The version number we will use is A.03.61.002 which is based on OpenSSH version 3.6p2. The product number is T1471AA.

   We install the Secure Shell package so we can connect to the server through an encrypted tunnel. This means that we will no longer have to transfer in clear text across the network. This measure reduces the risk of a "sniffer" attack where an attacker monitors network traffic to retrieve passwords, account names, and any other sensitive information.

   **/usr/sbin/swinstall -s **
   **/tmp/depots/T1471AA_A.03.61.002_HP-UX_B.11.11_32+64.depot **
   T1471AA

To save file system space, remove the depot files that were used to install the above applications.

It is always a good idea to review the log files after installing software to ensure all went as planned.

/usr/bin/more /var/adm/sw/swinstall.log
/usr/bin/more /var/adm/sw/swagent.log
/usr/bin/more /var/adm/sw/swconfig.log

Shadow Password Files

A shadow password file is a mechanism in UNIX that allows the storage of the password in a file other than /etc/passwd. In the past, the passwords on all accounts were stored in encrypted format in the /etc/passwd file. This is also the default setup after installation. This is a security exposure because the /etc/passwd file is required to be read by every account on the server. Even though the password is encrypted it is possible to run a password cracking program it to retrieve the password. The HP Trusted Computing Base (TCB) system provides the shadow password file functionality on an HPUX server. Right now we want to setup the TCB system. Later in this document we will configure the settings. Execute the following command to implement the TCB system.

/usr/lbin/tsconvert

Current Accounts

The installation of the OS leaves accounts on the system that should not be there and does not do a good job of securing the ones it does place out there.

We start with the root account. This account is the superuser account on the UNIX OS so you want to be careful when editing the configuration. The home directory is currently set to /. The / directory is accessible by all the accounts on the server. To limit access to the root account's home directory we create a directory /root and limit the permission settings and ownership.

Next move all the configuration files (the files with a dot (.) at the beginning with the exception of . and ..) to the /root directory. Since you will be using the root account for these steps it is necessary to edit the password file directly. To do this we will use the /usr/sbin/vipw utility which will check our work to make sure the file is not corrupted. Change the home directory (the sixth field) to /root instead of /. Also place a name on the account (the fifth field). We will make the name "UNIX Root Account".

All the other accounts on the system should not be logged into so we will go through the steps to disable them.

The daemon and sys accounts also used the / directory as their home directory. Since these accounts are not logged into and we do not want them to we will create a .profile file in the / directory that will be owned by the root account and the sys group with the permission settings set to read only by the owner and the group. This file will have one line and it will state exit. We will also change the assigned shell program to /bin/false on both accounts. We will use the modprpw utility to disable the account with the administrative lock.

/usr/lbin/modprpw –m alock=YES daemon

There are other accounts that need to be disabled in the same manner. A good measure to determine if an account is needed on the server is to check if the account owns files on the server, runs any processes, or runs any subsystems that are in use on the server. If none of these apply to the account, it can be safe to say that the account should not exist on the server.

The accounts that will be disabled are the adm, lp, sshd, webadmin, and www accounts.

The accounts that should be removed from this server are the hpdb, ids, nuucp, and uucp accounts. These accounts are used for the Allbase, IDS/9000, and UUCP applications respectively. Since we will not be utilizing these applications on this server, we will remove the accounts.

New Accounts

To make the system usable there must be user accounts. This step includes all the personnel responsible for administering the server. Add the appropriate accounts and groups to the server. Because of the purpose of this server we will only add the bare minimum number of accounts. These include the system administrator and security administrator accounts. Any other support personnel should not have access to the system.

We will use the group "datasec" to be the primary group for the UNIX security personnel.

We will also create a new group and account named delownr. This account's purpose is to be an interim file owner when an account has been removed from the server. To ensure this account is not misused, we will disable it through the administrative lock in the TCB system, set a very complex password, place the command "exit" in the account's profile configuration file, and change the shell program to /bin/false.

Network Services

The Internet Services Daemon (inetd) is the daemon that listens for incoming network connections and then spawns the appropriate daemon such as telnetd for a connection to the server using telnet. This saves system resources because you only have one daemon listening instead of all the services it supports having it own listening daemon.

The configuration file for this daemon is the /etc/inetd.conf file. This is the file where we can remove many of the network services on the server.

When you first edit the file you will notice there are already lines that have been commented (# as the first character). This action disables the service being invoked on that line. The following lines were already commented on this installation and will remain that way. We will not be running any of these services on this server so the disabling is done to remove the risk associated with running the service. For example, if a new vulnerability in a network service is discovered and the server is running that service, the server is vulnerable. However, if the service was disabled already, the server would not be vulnerable and therefore the patching is not as critical. A brief explanation of each service is given below.

1. **tftp** – The Trivial File Transfer Protocol service allows the remote transfer of files without authentication. The restrictions are usually set by which

files are “exported”. This action is accomplished in the inetd.conf file so ensure those lines (if any are present) are commented as well.

```
#tftp        dgram  udp wait   root /usr/lbin/tftpd    tftpd
```

2. **bootps** – The Internet Boot Protocol Server allows the serving of data to other systems. The data includes network configuration, startup files, etc. The network configuration is given if the server is being used as a Dynamic Host Configuration Protocol (DHCP) server. The other files would be given if this server is being used as an Internet Boot Protocol (BOOTP) server.

```
#bootps      dgram  udp wait   root /usr/lbin/bootpd   bootpd
```

3. **finger** – The Finger service is a network based version of the local finger utility. This service provides a detailed report of what accounts are currently logged into the server. Most of the information given could be deemed sensitive.

```
#finger      stream tcp nowait bin  /usr/lbin/fingerd  fingerd
```

4. **uucp** – The UNIX to UNIX Copy Protocol (UUCP) provides the framework for many utilities that are used to transfer files between UNIX systems.

```
#uucp        stream tcp nowait root /usr/sbin/uucpd    uucpd
```

5. **time** - The time service is internal to inetd and provides the local time of the server for anyone connecting to the service. The output is in machine-readable time, i.e. the number of seconds past January 1, 1900.

```
#time        dgram  udp nowait root internal
```

6. **rpc.rexd, rpc.statd, rpc.rusersd, rpc.rwalld, rpc.rquotad, and rpc.sprayd** – Instead of identifying all these services we will combine them under the title Remote Procedure Call Services. These specific services run with the portmapper process (or rpcbind) which basically dynamically assigns port numbers for network services.

```
#rpc  stream tcp  nowait  root  /usr/sbin/rpc.rexd     100017  1    rpc.rexd
#rpc  dgram  udp  wait    root  /usr/lib/netsvc/rstat/rpc.rstatd  100001  2-4 \
    rpc.rstatd
#rpc  dgram  udp  wait    root  /usr/lib/netsvc/rusers/rpc.rusersd  100002 \
    1-2  rpc.rusersd
#rpc  dgram  udp  wait    root  /usr/lib/netsvc/rwall/rpc.rwalld  100008  1   \
    rpc.rwalld
```

```
#rpc dgram udp wait   root /usr/sbin/rpc.rquotad 100011 1 \
    rpc.rquotad
#rpc dgram udp wait   root /usr/lib/netsvc/spray/rpc.sprayd 100012 1 \
    rpc.sprayd
```

7. **ncpm-pm and ncpm-hip** – These services are components of the Network Connection Policy Manager (NCPM) which provides load balancing features to a DNS setup.

```
#ncpm-pm        dgram udp wait root /opt/ncpm/bin/ncpmd ncpmd
#ncpm-hip       dgram udp wait root /opt/ncpm/bin/hipd  hipd
```

There are also several more services that we will be disabling. The following are these services and the reason why we are disabling the service.

1. **ftp** – This service allows the connection to and transfer of files to the server. This is done with the network traffic not being encrypted. This service is replaced by the Secure Shell (SSH) sftp service, which does encrypt the network traffic.

```
ftp         stream tcp nowait root /usr/lbin/ftpd      ftpd -l
```

2. **telnet** – This service allows the interactive logon and a shell given to the session. This service is also not encrypted and is replaced by SSH as well.

```
telnet      stream tcp nowait root /usr/lbin/telnetd  telnetd
```

3. **login and shell** – These services provide connection to the server while allowing the "password-less logon" with a trust relationship being setup. This trust relationship is setup in /etc/hosts.equiv and ~/.rhosts files. In the event that these services are needed, use the –l option to disable the authentication based on account's .rhosts file. This step disables all the .rhosts file with the exception of the root account's. This service is also not encrypted and any of the functionality is replaced with SSH as well.

```
login       stream tcp nowait root /usr/lbin/rlogind  rlogind
shell       stream tcp nowait root /usr/lbin/remshd   remshd
```

4. **exec** – This service provides the ability to execute commands on the local server remotely. The authentication performed is based on the account name and password on the local server. The problem with this service is that the network traffic is not encrypted. The functionality needed is replaced with SSH as well.

exec stream tcp nowait root /usr/lbin/rexecd rexecd

5. **ntalk** – This service provides functionality for the talk utility so users can communicate with each other. Due to the functionality of this server we can clearly state that this service is not needed. If we were to run the service it will place unneeded risk to the server. If the service was to have a vulnerability and we do not need to have it running, the server would get the same vulnerability for no reason. It is always best to disable services that will not be used.

   ntalk dgram udp wait root /usr/lbin/ntalkd ntalkd

6. **ident** – The Identification Protocol service allows a remote service to query who is connecting to its server. The information given can be stripped somewhat but may still be considered sensitive. Due to the fact that most administrators disable this service, not too many applications require it to be active. We disable this service because it should not be needed and therefore gives unneeded risk.

   ident stream tcp wait bin /usr/lbin/identd identd

7. **printer** – The Remote Spooling Line Printer Daemon (rlpdaemon) is used to handle remote requests to print. There have been several vulnerabilities already found in this service and as always, future ones could also be found. Because we will not be needing this daemon we are disabling the service.

   printer stream tcp nowait root /usr/sbin/rlpdaemon rlpdaemon -i

8. **daytime, time, echo, discard, and chargen** – All of these services are internal to inetd and will not provide any needed functionality to this server. We will disable all these to rid ourselves of the risk associated with running them.

   The daytime and time services provide the local time of the server for anyone connecting to the specific ports. The daytime provides the time in human readable format and the time provides the time in machine-readable time (number of seconds past January 1, 1900).

   The echo service allows a connection to be established on the local server that sends the input to it as the output to the connection. This is just like the echo utility on the local server but this one is a network service.

   The discard service allows a connection to be established on the local server but all input is discarded or thrown away.

The chargen service generates strings of characters for incoming connections regardless of the connection's input.

The User Datagram Protocol (UDP) version of the time service is already disabled so we will leave it that way.

```
daytime      stream tcp nowait root internal
daytime      dgram  udp nowait root internal
time        stream tcp nowait root internal
#time        dgram  udp nowait root internal
echo        stream tcp nowait root internal
echo        dgram  udp nowait root internal
discard     stream tcp nowait root internal
discard     dgram  udp nowait root internal
chargen      stream tcp nowait root internal
chargen      dgram  udp nowait root internal
```

9. **kshell and klogin** – These services provide the same functionality as the remsh and rlogin services do. The difference is that these services interact with the Kerberos authentication application which will not be used on this server. Therefore we will disable these services to removed the risk.

   ```
   kshell stream tcp nowait root /usr/lbin/remshd remshd -K
   klogin stream tcp nowait root /usr/lbin/rlogind rlogind -K
   ```

10. **dtspc** – This service is the Subprocess Control Service for CDE. It provides the necessary responses when a CDE session is being requested. The users on this system will be connecting to the server with SSH so CDE is not required. Therefore the risk is not warranted and the service will be disabled.

    ```
    dtspc stream tcp nowait root /usr/dt/bin/dtspcd /usr/dt/bin/dtspcd
    ```

11. **rpc.ttdbserver** – This service is part of the ToolTalk database server which will not be used on this server. We will disable this service as well to remove the risk associated with running it.

    ```
    rpc xti tcp swait root /usr/dt/bin/rpc.ttdbserver 100083 1 \
    /usr/dt/bin/rpc.ttdbserver
    ```

12. **registrar** – This service is used by the Event Monitoring Service (EMS) which will not be used on this server. To eliminate the risk of running this unneeded service we will disable it.

registrar stream tcp nowait root /etc/opt/resmon/lbin/registrar \
/etc/opt/resmon/lbin/registrar

13. **recserv** – This service is the HP SharedX Receiver Service that allows the sharing of windows without the use of xhost utility. This service will not be used on this server so we will disable it as well.

    recserv stream tcp nowait root /usr/lbin/recserv recserv -display :0

14. **rpc.cmsd** – This service is the calendar manager service daemon that provides an appointment calendar for users. The client that uses this is included in CDE. Users will not be using this service so we will disable it to remove the risk associated with running it.

    rpc dgram udp wait root /usr/dt/bin/rpc.cmsd 100068 2-5 rpc.cmsd

Now that we are finishing editing the /etc/inetd.conf file it is necessary for inetd to be configured with the new settings. To accomplish this we run the daemon with the "c" option which forces the current inetd to reread the configuration file.

/usr/sbin/inetd –c

When we review our changes that we just made to the /etc/inetd.conf file we realize that we commented every service that was configured. From this point we could disable the daemon itself, instead we will keep it running in the event we need it in the future.

We also selected to disable the services by commenting the line. This was only temporary to ensure we do not encounter any problems. After we are ensured that there are no problems, we can remove the entire line from the file. Most experts say that removing the line is the most secure because there are widely available hacking toolkits that just remove the # from the line to enable the service. However, if the hacking toolkit is modifying the configuration file, it is already running with root account privilege so it would be a trivial matter to just add the line back in. Therefore it is not necessary to remove the lines from the configuration file.

Time Zone Settings

The time settings on a server are very important to both applications and security monitoring so you can be sure that a specific event happened at a specific time. During the installation we set the time zone to the appropriate setting so now we want to go back and ensure that the settings were set properly.

The system startup process looks at the /etc/TIMEZONE file by default for this information. Review this file to ensure it is set correctly. This server has the following data in the file:

```
TZ=CST6CDT
export TZ
```

System Startup

The system is started with the init process. This process reads its configuration file, /etc/inittab, to know what actions to perform. As a word of caution, do not edit this file unless you know what you are doing. For this installation though we will be commenting the lines that pertain to EMS since we will not be using it. Everything else we will leave the way it is.

An important step in the /etc/inittab process is the execution of the /sbin/rc file. This file is a shell script that reads the /sbin/rc*.d directories to start the appropriate software on the system. For example if you are going to run level 3 at boot all the applications in the /sbin/rc1.d, /sbin/rc2.d, and /sbin/rc3.d directories. By default, the files in these directories are symbolic links to the shell scripts located in /sbin/init.d.

When disabling the application from starting up you can either remove the link entirely or you could change the first character, S or K, to a lower case character. Changing the first character is good practice so it will be easier to recover in the event that you do require the application. You could then come back and remove the link later.

When we disable the applications we will first disable the startup scripts. After they are disabled we will reboot the server. When the system comes back up the shutdown scripts are no longer needed so we can then disabled them. These steps are done to ensure that the applications shutdown properly.

The following application startup and shutdown files will be disabled for this installation.

/sbin/rc0.d

This directory is used for final system shutdown scripts. We will not modify anything here.

/sbin/rc1.d

This directory contains a lot of the network applications' startup and shutdown scripts. The following scripts will be removed since we will not be using the applications on this server.

- K100ems -> /sbin/init.d/ems
- K230audio -> /sbin/init.d/audio
- K570nfs.client -> /sbin/init.d/nfs.client
- K580nis.client -> /sbin/init.d/nis.client
- K590nis.server -> /sbin/init.d/nis.server
- K592nisplus.client -> /sbin/init.d/nisplus.client
- K594nisplus.server -> /sbin/init.d/nisplus.server
- K600nfs.core -> /sbin/init.d/nfs.core
- K630named -> /sbin/init.d/named

/sbin/rc2.d

This directory contains more of the startup scripts that the run level 1. The following scripts will be removed since we will not be using the applications on this server.

- K900nfs.server -> /sbin/init.d/nfs.server
- S202clean_uucp -> /sbin/init.d/clean_uucp
- S370named -> /sbin/init.d/named
- S400nfs.core -> /sbin/init.d/nfs.core
- S406nisplus.server -> /sbin/init.d/nisplus.server
- S408nisplus.client -> /sbin/init.d/nisplus.client
- S410nis.server -> /sbin/init.d/nis.server
- S420nis.client -> /sbin/init.d/nis.client
- S430nfs.client -> /sbin/init.d/nfs.client
- S770audio -> /sbin/init.d/audio
- S900ems -> /sbin/init.d/ems
- S900emsa -> /sbin/init.d/emsa

/sbin/rc3.d

This directory contains more of user defined application startup processes. There should not be a lot of files in this directory because we have not installed any third party applications yet. The following scripts will be removed since we will not be using the applications on this server.

- S100nfs.server -> /sbin/init.d/nfs.server

/sbin/rc4.d

At installation time there are no files located in this directory.

System Log Files

To ensure there are no problems with the server configuration review the /etc/rc.log and /var/adm/syslog/syslog.log files. When reviewing these files, take extra time reviewing the startup of the services. You want to make sure that the services that should be disabled are and you also want to make sure that the services we will be using actually did start with no problems. If there are any problems they will be identified in these log files.

Host Resolution Setup

The host resolution on the HPUX server is setup by default to be accomplished by consulting the DNS servers setup in the /etc/resolv.conf file. We will change this default behavior by using the /etc/nsswitch.conf. This file is not on the system when it is first installed so we will create it. The permission settings are read only for the owner, group, and the world. The file will be owned by the root account and the root group. The following will be the contents of this file.

```
hosts: files[NOTFOUND=continue TRYAGAIN=continue] dns
```

These settings will cause the server to check the /etc/hosts file first before consulting the DNS servers. This will speed up the resolution because it won't have to go off server. It also can be used to prevent DNS attacks, such as DNS spoofing, if we place the IP addresses most used by this server in the hosts file. An important server to place in this file is the primary UNIX security server. We also check the hosts file for the correct permission settings. It will be setup with the bin account and the bin group owning the file with read only set for the owner, group, and the world.

We will also need to review this file to ensure that all the host names to IP addresses are set correctly.

SUID and SGID Files

When installed the operating system contains over two hundred SUID and SGID files. A important note is that just because the vendor supplies a binary with the SUID bit set, it does not necessarily mean that it should be set. We need to investigate the uses of all the files that have these bits set and decide if the program truly needs the SUID or SGID bit set on it. Sometimes the vendor will supply the programs this way so all users on the system can execute it, when in this specific environment, the program may be considered a systems administration function so all users do not need that access.

Since the list is too long to be documented here, we will go through a few of the files set with the SUID or SGID bit set.

We first go through the list and eliminate the permission settings on the utilities we will not be using on this server. The following is a sample of these utilities we will either remove from the server or just remove the permission settings.

/usr/bin/nispasswd
/usr/bin/yppasswd
/usr/bin/newgrp
/usr/bin/nfsstat
/usr/bin/kermit
/usr/bin/uucp
/usr/bin/uuls
/usr/bin/uuname
/usr/bin/uusnap
/usr/bin/uustat
/usr/bin/uux
/usr/lbin/uucp/uucico
/usr/lbin/uucp/uuclean
/usr/lbin/uucp/uusched
/usr/lbin/uucp/uusub
/usr/lbin/uucp/uuxqt

Next step we go through the files that remain in our list and determine which utilities should only be ran by the root account or another system level account. There is no point in allowing any user to execute it when they should not be executing it anyway. The following is a sample of the programs we disabled the SUID or SGID permission setting.

/usr/bin/lpalt
/usr/sbin/swinstall
/usr/sbin/swpackage
/usr/sbin/swacl

/usr/sbin/swconfig
/usr/sbin/swcopy
/usr/sbin/swlist
/usr/sbin/swremove
/usr/sbin/swverify
/usr/sbin/swreg
/usr/sbin/swmodify
/usr/sbin/acct/accton
/usr/sbin/lvchange
/usr/sbin/lvcreate
/usr/sbin/lvdisplay
/usr/sbin/lvextend
/usr/sbin/lvlnboot
/usr/sbin/lvreduce
/usr/sbin/lvremove
/usr/sbin/lvrmboot
/usr/sbin/pvchange
/usr/sbin/pvck
/usr/sbin/pvcreate
/usr/sbin/pvdisplay
/usr/sbin/pvmove
/usr/sbin/pvremove
/usr/sbin/vgcfgbackup
/usr/sbin/vgcfgrestore
/usr/sbin/vgchange
/usr/sbin/vgchgid
/usr/sbin/vgcreate
/usr/sbin/vgdisplay
/usr/sbin/vgexport
/usr/sbin/vgextend
/usr/sbin/vgimport
/usr/sbin/vgreduce
/usr/sbin/vgremove
/usr/sbin/vgscan
/usr/sbin/lpadmin
/usr/sbin/lpfence
/usr/sbin/lpmove
/usr/sbin/lpsched
/usr/sbin/lpshut

System Profile

The system profile, /etc/profile, is used by all accounts that logon to the server. Since all the accounts on this server we will be utilizing this configuration file, we should take a minute to review its contents.

The first items we need to check are the PATH and MANPATH settings. We check all the directories set in the path settings to ensure they are appropriate for this server.

Next we see that there is not a umask setting on the server. This setting is used to determine the permission settings that are set on new files created on the server. Since this check is part of the automated security review, detailed information on the umask value can be found in Section 9. We place the following line in the /etc/profile file.

umask 027

Since we will not be utilizing the following functionality on the server, we will comment the following lines.

```
# Message of the day

    if [ -r /etc/motd ]
    then
            cat /etc/motd
    fi

# Notify if there is mail

    if [ -f /usr/bin/mail ]
    then
            if mail -e
            then    echo "You have mail."
            fi
    fi

# Notify if there is news

    if [ -f /usr/bin/news ]
    then news -n
    fi

# Change the backup tape
```

```
if [ -r /tmp/changetape ]
then    echo "\007\nYou are the first to log in since backup:"
        echo "Please change the backup tape.\n"
        rm  -f /tmp/changetape
fi
```

We also notice that the configuration file outputs the /etc/copyright file to the screen. We need to place an approved warning banner for our company in this file. We will put the following in our copyright file after the entries that are already in there.

THIS IS A PRIVATE COMPUTER SYSTEM ---
USAGE MAY BE MONITORED AND UNAUTHORIZED ACCESS
OR USE MAY RESULT IN CRIMINAL OR CIVIL PROSECUTION

Section VII – eTrust Access Control

This section will be used to discuss the third party security application we will be using and what it will be used for.

Introduction

The security application, eTrust Access Control, is sold by Computer Associates and is one of those products that penetrates the operating system which then has the ability to deny even the root account. Like all software there are problems but the uses of this product outweigh the problems.

This section only outlines what we will need to accomplish on this server. We already have a system designed to be the master server for the administration of this application.

Installation

To install eTrust Access Control you will need root account access (surrogate access will work). Since this server is being added to the current network, there are already procedures used to install the security application. That is to say we have already developed the automated installation process that we will use to install the security application on the server. However, we will go through what the installation process does and what needs to be done manually on the server.

The first step is to create the mount point for /usr/seos. We created the file system and mount point earlier during the installation of the OS.

We need to transfer the installation files from the central security server to this system. The installation files are in tar format so we need to extract the files to the /usr/seos directory.

We then execute the installation script. This script will perform the following steps.

- Install the security application to the /usr/seos directory with the files being owned by the root account and the datasec group.
- Since this server is a UNIX security server, it will be setup with the server and client packages.
- Configure the log routing daemon to send all log messages to the central security server.

- Set the encryption for all network communications to utilize Triple DES.

- Change the default seos.ini configuration file with the standard configuration file used on all HPUX servers on the network. This standard file includes all the standard settings used on all the servers. These settings include the which daemons to start up, daemon configuration, encryption settings, maximum size of log files, and other settings needed to ensure the security application runs the same on all the servers.

- Change the other configuration files in the security application so they match the standard used on all the servers on the network.

- Create an account that exists only in the security application. This account will be the owner of the rules in Access Control. This will prevent users from owning rules in the security database and therefore all users will require the admin attribute to modify the rules.

- Automatically configure the system and security administrators accounts on the server to have the proper attributes within the security application. The system administrators receive the operator attribute while the security administrators will receive the admin and auditor attributes.

- In the event that a system administrator is installing the security application, that account would receive the admin attribute within eTrust Access Control automatically. This installation removes the attribute if this is the case.

- Create and configure the startup process in the system startup directory for run level 2.

- Implement the default rule base for the HPUX server. This rule base contains a compilation of rules that will be used to protect system utilities and configuration files, access lists for terminal access, surrogate protection, and to protect other sensitive files and security processes. Due to the size of this rule set it will not be included in this document.

- Modify the terminal class to ensure only the appropriate personnel have access to modify the database and from which servers the database can be modified from. Take note, that these accounts still require the admin attribute to modify rules in the security database.

- Install the in-house developed exit scripts that are used by eTrust Access Control when creating, deleting, or modifying accounts and groups. Consult further down in this section for additional details on exit scripts. We also remove the default exit scripts installed with the security application.

- Install in-house developed programs on the server which call the security application's utilities. The programs are used for members of the group datasec to rebuild the index files in the database and to start the security application. They are used when the security application is not running and are SUID programs to the root account. The reason we install this is because most users that get root account access do not know the root account's password so therefore can not use the su utility to surrogate to the root account.

- Ensure the /usr/seos/bin/sesu file is set with the appropriate permission setting with read and execute for all fields, SUID set, and the root account and the datasec group owning the file. Problems have been encountered many times if this step is not accomplished.

- Send the log file created of all the above steps and send the data to the appropriate personnel for review to ensure the installation worked properly.

- Rebuild the kernel and reboot the server.

Access Rules

One access rule that we will go into further detail is in the surrogate class and is the rule for the USER.root. This rule is used to make all root account access to be reviewed and checked for authorization before it occurs. This means that even though a file is marked SUID to the root account, this SUID action will be denied unless the rule for the access is provided in the USER.root surrogate rule. We allow the server and security administrators to have access while the other accounts will have access via specific programs like passwd and login.

This rule is very useful in limiting the impact that SUID programs have because it limits the accounts that can execute them. This greatly improves the security of the UNIX platform and makes it easier because you can control the security database remotely from the central security server.

Another concept to handle is over all the rules we create in the database. We can use this security application to prevent system and security administrators or even the root account from accessing specific files on the server. With this

application you can determine that security administrators should not be modifying the settings for the disks and file systems. So we write the rules to allow only the system administrators to have access to the utilities used. Now, even though the security administrator has access at the UNIX level because they have root account access, eTrust Access Control will check its access control lists and deny the access. This works because the security application tracks who you were when you first logged in, not who you are now. We use this type of rules to deny the root access to certain files because our policy will state that the root account should not natively logon to our server.

By default the security application is used for all authentication. However this does not work that well with TCB files in the HPUX OS. It also creates a vulnerability in the event that the security application is not running. If this is the case there will be no password checks or password aging done. Since we can't run both TCB and eTrust Access Control password controls, we will only run with the TCB files enabled. This will provide coverage even though the security application is not running and will benefit us if we decide to remove the third party application from the server. For this installation we will disable the password class in the security application.

Exit Scripts

When any action is made to an account or group from the security application, it hands off control before and after the change is made to a set of scripts called exit scripts. If configured, there will be a script that executes before the action takes place and another after the action has been performed. These scripts are quite useful when utilized for account additions, account deletions, and password resets on accounts. The following are the exit scripts we will utilize along with an overview of the functions of each.

- **Account Additions** – After adding an account on the server, the exit script is utilized for making the appropriate changes to the account's configuration settings and files along with ensuring specific authority is setup at the UNIX level for the account. The exit script knows what steps to take because of the location setting on the account. Since we will not be using this setting on this network, we use it for a variable setting to pass data to the exit script. The exit script also rebuilds the user component of the look aside database. Further information on the look aside database is found a little further down in this section.

- **Account Deletions** – Before removing an account from the server, the file system is searched for any files owned by the account. The exit script also takes an archive of the account's home directory and places it in the /home/deleted secure repository in the event the files were

actually required. After the account has been removed from the server, the exit script changes the file ownership of the files owned by the account to the delownr account (explained earlier in this document). The list of files changed are then sent to the UNIX security analyst for review. The files will either be removed or the ownership will be changed to a valid user.

- **Account Modifications** – Since the TCB files are being utilized, it is necessary to modify the settings in the file when changing an account's password. The only time we use the exit script for modifications is when the password is being changed. Before the account has the password changed we use the exit script to check the current lockout settings. If the administrative lock is set on the account, the password is not changed. This step allows a method for not allowing an account to be reset. All other reasons for a lockout will be cleared. After the security application changes the password on the account, the lockout field is checked again to ensure there are no problems. The exit script then looks to determine if the location variable was changed to a specific value. If this value is set, the exit script will be complete. If the value is not set, the password is set to expire on the next logon. This is helpful for the administration of the servers on the network because it makes the monthly password changes on two hundred servers much quicker.

Related Processes

To ensure the security application is running correctly we will implement processes to run routinely to perform the necessary maintenance.

The Look Aside Database (LADB) for the security application is required for HPUX 11.x servers. The LADB is used to speed the resolution of accounts, groups, hosts, and services for the security application. Instead of the security application waiting on the OS to perform the resolution, the security application consults its own resolution database. Unfortunately the problem with the LADB is there will be no clear error messages if there is a conflict in the resolution. An example of this problem would be if a remote computer has changed its IP address since the last update of the LADB. This means that a process is needed to update the LADB on a routine basis. Fortunately for us a utility named /usr/seos/bin/sebuildla has been included with the security application. We will place this command with the –a option in the root account's crontab file. Since this network is fairly large we will schedule this process to run twice a day.

The security database that holds all the access rules is internally protected by the security application. This prevents the backup of the files located in the seosdb

directory. To ensure we have backups of the database we implement a process that will utilize the dbmgr utility to extract the rules from the running database. This database dump is then transferred to the central security server. We schedule this process to run once a day.

The security application does not help the security of the server if it is not running. Therefore we implement a process that checks all the daemons that should be running on this server. If the process finds any of the daemons not running it will create an error report and send it via e-mail to the appropriate personnel. We will schedule this process to run every four hours.

Reporting

As with all security applications, if you do not review the log files produced, it will not help the overall security of the server. The eTrust Access Control product includes the functionality to send specific log messages anywhere you want but since there are a lot of servers on the network we will compile all the log messages on the central security server.

The central security server analyzes all the log files that are sent to it and produces several reports that are sent to the security analysts for review. One of these reports is compiled every fifteen minutes. This report is a list of all the denied access attempts in the last fifteen minutes. The next set of reports is accomplished daily on the logs for the day prior. The following are the reports generated:

1. Deny Messages – This report compiles all the denied access attempts. The UNIX security analyst reviews the report to ensure all denied attempts should have been denied. If they were, the analyst must follow the proper procedure in reporting this violation.

2. Startup and Shutdown Messages – This report compiles all the log messages that identify when the security application was turned on or off. The UNIX security analyst reviews this report to ensure all events were authorized.

3. Success Messages – This report compiles all the successful changes made to the security database. The UNIX security analyst checks to ensure only authorized users made changes and that only authorized changes were made.

4. Untrusted Messages – This report compiles all the untrusted programs and secfiles. These messages are from the built in file integrity checking

with the security application. Since we do not use this functionality on this server there should be none of these messages.

5. Warning Messages – This report compiles all the messages marked as warning. This happens when the rule is set in warning mode. These attempts would have been denied. This mode is used when a new server is installed and the analyst is not completely sure what the rules will do in the environment or what access the specific applications on the server will require. The UNIX security analyst reviews this report in the same manner as the deny messages report.

Additional reports are made on all the access made by a specific user that is being monitored. Since there are none of these user accounts on this server, we will not get into detail on this type of report.

Section VIII – UNIX Security Automated Processes

This section will be used to discuss the automated processes we will setup on this server and what they are used for.

Introduction

There are quite a few processes that will be installed on the server that we will use to ensure the security of the server. If all the items we wanted to check on each server had to be done manually we could never analyze everything we needed unless of course we were going to hire a small army for the securing of the UNIX servers on the network. This is why we automate processes and set them up to routinely check the servers on the network and get back to us with any errors or misconfigurations that are found. This way we take the personnel we do have and get a lot more security work out of them.

We will provide a list of the automated processes we will implement including a description of what the process does, how often the process will run, and what is done with the reports made by the process.

Installation

The installation of all the automated processes are accomplished by using the Software Distributor (SD-UX) provided with HPUX. The software depots are stored on our central security server so we will use the swinstall utility to install. The following command will be used (we substitute *host* for the central security server's name):

/usr/sbin/swinstall –s *host*:/var/spool/infosec INFOSEC_PROCESSES

This command will automatically install and configure all the processes for the security checking. The configuration includes the setting of all the necessary processes in the cron facility on the server.

The remainder of this section will be used to identify the processes installed.

Automated Security Review

This process is a compilation of various scripts that is used to check the server for a variety of items. This process is detailed in its own section later in this document. This process is scheduled to run during the off production hours of the server. The main process handles the scheduling of the checks based on the

day of the week, with every process running at least once a week with some running every day of the week.

Automated File Transfer

This process is not set to run but is worth mentioning because it will be the framework used for most of the processes in this section. In the HPUX OS you can't just select another server and then transfer files to it without some form of authentication. There are several options available for us to transfer files across servers in the environment and all of them have their own issues and vulnerabilities. For example, we could setup a trust relationship with the /etc/hosts.equiv file so we can transfer files but the remote server is then authenticating this server based on the IP address. This example would open the remote server to the possibility of an IP spoofing attack. Another example would be the coding of a password directly in the process which allow any account with read access to the process to obtain the remote server's account and password used for the file transfer. Add to that, if we utilize ftp for the transfer the process is open to a packet sniffing attack when the data is going across the network.

Instead of using services already there we will create our own process and protect all the executable files and configuration files with the security application, eTrust Access Control. This securing will even be prohibiting the root account from modifying and reading the files associated with the process. This process will utilize the OpenSSH file transfer mechanism and digital certificates to authenticate the server. We will create a public key for the remote server and a private key for this server. The keys will be locked down for this server to server connection. The public key is stored in the incoming account's home directory and is protected by the security application. The private key is stored in a secured repository and will be used at the time of transfer and then will be removed again back to the repository. In this configuration a manual step is required to establish the remote server's authenticity by accepting the RSA key fingerprint for the server. The account on the other system will also be locked down to ensure the only logon will occur using sftp or scp from this server.

All requests to transfer files automatically from this server will have to be accomplished using this process.

Cron Settings Check

This process compiles the root and datasec account's crontab files and then transfers the data to the primary central server. The datasec account is the locked down account used for processing of security processes that do not

require root account access. We schedule this process to run three times a week.

The central server analyzes these files to ensure the jobs that should be scheduled are and the jobs that are scheduled should be. All the files on the central server are also checked to ensure they have been updated. This process is important for making sure that all the security processing is being completed on all the remote servers. It is also important because it checks the jobs scheduled on the remote servers to ensure the facility is not being misused. Only authorized jobs should be scheduled and all jobs scheduled should be reviewed to ensure they are secure.

Group and Password File Backup

This process takes the /etc/group and /etc/passwd files on the server and copies them to a backup copy. The backup copies are also transferred to the central security server. Obviously the backup is a good idea in case one of these files are corrupted but additional processing on these files are done on the central server. We schedule this process to run daily.

The group and password files for all the servers are checked on the central security server to ensure that the system level and other limited access groups only contain the authorized accounts. This process is scheduled to run twice a week from the central server.

TCB and Password File Collector

This process compiles the /etc/passwd file and all the files under /tcb and then transfers the files to the central server for additional processing. The processing on these files are done off the server to centralize the processing and to offload the utilization of system resources off of the production servers in the environment. This compilation process is scheduled to run every six hours.

The following are the processes on the central server that utilize this data.

The main process to utilize this data is the reconciliation process. This process takes all the accounts from the password files and then checks the time of the last logon for each account. The logon information is stored in the tcb file for the account in epoch format, number of seconds since January 1, 1970. This epoch time is then converted to Gregorian format (i,e, 01/01/1970) for ease of use in the reporting. This compilation of all the accounts is then used to reconcile against the master list of accounts and servers they should have access to. Any differences in the lists is compiled in a report that is reviewed by the security

analyst. The main compilation process of this list is compiled every four hours and the reconciling process is accomplished once a day. The following are additional processes that utilize this compiled list.

- The list is also checked daily for the accounts that have not logged in for over a year. This report is sent to the UNIX security analyst for review daily. The analyst will then disable the inactive accounts and then schedule them to be removed from the remote servers in one week. The reason we perform this check is because an inactive account is a good account to break into because it is unlikely that the owner of the account will notice that the account was broken into. If the account is removed, there will obviously not be a way to break into it.

- The list is also used by the password reset and user add processes. The password reset process checks the compiled list to ensure the account exists on the remote server before connecting to the remote server. The user add process checks the compiled list to ensure the account does not already exist on the remote server before attempting to add it.

- The compiled list is reviewed every time it is created for what accounts have been added or removed since the last run. The security analyst then receives a report of all the account changes made to the servers on the network. The analyst reviews this report to ensure all account additions or deletions were authorized. In the event an attacker was to add an account on a server, this process would report it.

Another process to utilize the compiled files from the remote servers is another compilation process. This process is similar to the one that produces the reconciliation report with the exception that it also checks the time of the last password change. This field is also in epoch format and is also converted to Gregorian format. This compiled report is used by another process that is used to check for administrative level accounts that have not had the password changed in more than thirty days. These accounts have privileged access and the passwords should change often in case one is cracked.

An additional process to utilize the compiled files is a password cracking process. This process compiles the password files and tcb files and creates a “shadow” file similar to the /etc/shadow files on other UNIX operating systems. This is basically the password file with the asterisk removed and the encrypted password in its place. This file is then ran against a open source program named John the Ripper which attempts to crack the password. Cracking a password consists of taking a password, encrypting it, and then matching the output to the encrypted password found in the file. This is the same way the system utilities that check passwords, such as login and passwd, work. These steps are needed

because it is not possible to decrypt the passwords. Without getting into too much detail on this, the password is encrypted with a one-way hash. This makes it computationally impossible to decrypt the password. The report of passwords cracked are sent to the UNIX security analyst for review. The analyst will notify the user of the weak password and also change it for them. This process is ran weekly from the central server. As a note, realize that just because the password was not able to be cracked with the dictionaries in use does not mean that the password is 'secure' or unable to be cracked by different dictionary attacks or a more personal password guessing process. Even so, this step is a must because attackers have access to the same tools so it is a matter of who 'cracks' the password first, the security administrator or the attacker.

Another process that uses the collected files is the TCB settings check process. This process analyzes the TCB files for system settings and account settings to ensure they abide by policy. Any setting that does not fit the policy is placed into a report that is reviewed by the security analyst. This report is generated daily. The following tables show the settings that are to be used on this server.

The command line field column is to be used with the following HPUX utilities, /usr/lbin/getprdef, /usr/lbin/getprpw, /usr/lbin/modprdef, and /usr/lbin/modprpw. These utilities were unsupported until the 11.11 release of the operating system.

Table 8.1. TCB System Settings

TCB System Settings			
TCB Field	Command Line Field	Description	Required Value
d_boot_authenticate	bootpw	Whether or not boot authentication is required to boot the machine.	NO
u_minchg	mintm	Minimum password change time	0
u_maxlen	maxpwln	Maximum length of password used when system generates password.	13
u_exp	exptm	Password expiration time.	30
u_life	lftm	Password lifetime.	60
u_llogin	llog	Maximum time allowed between logons.	60
u_pw_expire_warning	expwarn	Amount of time before password expires to warn user.	5
u_pickpw	usrpick	User allowed to pick password.	YES
u_genpwd	syspnpw	System generates	NO

		pronounceable password.	
u_restrict	rstrpw	Password triviality checks are performed. * See NOTE(1) below.	YES
u_nullpw	nullpw	Null password allowed.	NO
u_genchars	syschpw	System generates character only password.	NO
u_genletters	sysltpw	System generates letter only password.	NO
u_maxtries	umaxlntr	Maximum number of consecutive unsuccessful login attempts allowed per account.	3
t_logdelay	dlylntr	Time delay between unsuccessful login attempts.	3
t_maxtries	tmaxlntr	Maximum number of consecutive unsuccessful login attempts allowed per terminal.	5
t_login_timeout	lntmout	Login session timeout time.	0

NOTE(1): The password triviality checks are performed on password selections made by users. The checks performed (taken from HP documentation) include verifying that the password does not represent a login or group name, a palindrome, or a word recognized by the spell utility. The spell utility uses a spelling list that contains more words than an ordinary dictionary (contains more proper names and popular technical words) but does not cover specialized words, such as biology, medical, and chemistry vocabularies. This check also ensures the password selected is at least six characters long.

Individual accounts on the server also have TCB files. The settings found in the account TCB files override the system settings and should be set with care. The following table lists the values found in account TCB settings and the required settings for the field. The default settings found below default to the system settings while a dash means that the settings are account specific and would not be found in the system-wide settings.

Table 8.2. TCB Account Settings

TCB Account Settings			
TCB Field	Command Line Field	Description	Required Value
u_name	-	Account name that matches the name of the file and the	-

		account name found in the password file.	
u_id	uid	The UID for the account that corresponds with the UID found in the password file.	-
u_pwd	-	The encrypted password for the account.	-
u_boot_auth	bootpw	Whether or not boot authentication is required to boot the machine.	NO
u_auditid	audid	The audit ID for the account.	-
u_auditflag	audflg	The audit flag for the account.	-
u_minchg	mintm	Minimum password change time	DFT
u_maxlen	maxpwln	Maximum length of password used when system generates password.	DFT
u_exp	exptm	Password expiration time.	DFT
u_life	lftm	Password lifetime.	DFT
u_succhg	spwchg	Time of last successful password change.	-
u_unsucchg	upwchg	Time of last unsuccessful password change.	-
u_llogin	llog	Maximum time allowed between logons.	DFT
u_pw_expire_warning	expwarn	Amount of time before password expires to warn user.	DFT
u_pickpw	usrpick	User allowed to pick password.	DFT
u_genpwd	syspnpw	System generates pronounceable password.	DFT
u_restrict	rstrpw	Password triviality checks are performed. * See NOTE(1) above.	DFT
u_nullpw	nullpw	Null password allowed.	DFT
u_pwchanger	-	Account name that last changed the account's password if it was not the account's owner.	-
u_pw_admin_num	admnum	Random number generated for user logon.	-
u_genchars	syschpw	System generates character only password.	DFT

u_genletters	sysltpw	System generates letter only password.	DFT
u_tod	timeod	Account restrictions on use during a specific time frame.	Not Set
u_suclog	slogint	Time stamp of last successful login time.	-
u_suctty	sloginy	Name of the terminal or remote host used for the last successful login.	-
u_unsuclog	ulogint	Time stamp of last unsuccessful login time.	-
u_unsuctty	uloginy	Name of the terminal or remote host used for the last unsuccessful login.	-
u_numunsuclog	culogin	Number of unsuccessful login attempts to the account. This number is reset after a successful login to the account.	-
u_maxtries	umaxlntr	Maximum number of consecutive unsuccessful login attempts allowed per account.	DFT
u_lock	alock	Administrative lock set on the account. This prevents the account from being logged into. * See NOTE(1) below.	-
-	lockout	Bit string that shows if the account is locked or not. * See NOTE(2) below.	-

NOTE(1): The accounts on the server are reset using the password management process. This process utilizes eTrust Access Control to reset the TCB lockout field and then changes the password on the account. Exit scripts executed on the remote server handle this reset of the TCB file. However, this exit script will not reset the account if the administrative lock is set on the account. This prevents the helpdesk and other password changing personnel from resetting accounts that an administrator has placed a lock on.

NOTE(2): The lockout field is a string of seven characters. Normally all the characters are zero which means that the account is not disabled. When there is a one in the string, it means that the account is disabled. Depending on the

location of the one, is how the account was disabled. The following list explains what each character stands for with the first character starting on the left.

- One Password lifetime exceeded
- Two Time between logins exceeded
- Three Account absolute lifetime has been exceeded
- Four Unsuccessful login attempts exceeded
- Five Null password set but not allowed
- Six Administrative lock set on the account
- Seven Initial password was not set on the account

Software Configuration Reporting

This process utilizes the /usr/sbin/swinstall utility provided as a component of the Software Distributor application (SD-UX) included with HPUX. The basic function of this process is to identify all the applications and patches installed with SD-UX. This compilation is then transferred to the central security server where all the servers on the network places its data. This compilation can then be reviewed to ensure specific patches or applications are installed on the remote servers. We will schedule this process to run daily.

Valid Logon Activity Archival

The /var/adm/wtmp file contains information on all the logins, logouts, and system shutdowns that occurred on the server. This file will be backed up for historic and trend analyzing purposes. The backups of this file should not be stored on the local server and the optimum storage medium would be a tape in a locked room. This file also will grow too large and will fill the file system, so the file should be cropped at the same time that it is backed up. However, an accounting process can be utilized in the HPUX environment to accomplish this. This process is included with the OS and is performed with the adm account. We will not delve into how this process is setup because there is plenty of documentation available. The script used is /usr/lib/acct/runacct with the backup copy being /var/adm/acct/nite/owtmp. This process does have its problems and is flaky at best. Therefore our process will manually crop the file on a routine basis and copy the archive off to the central security server. Take note that if the wtmp file is removed from the file system, the auditing functionality that it provides will cease to exist because the utilities that place the data into the file will not create the file if it does not exist. We will set this process to run daily.

Invalid Logon Activity Reporting and Archival

The /var/adm/btmp file contains information on the bad login attempts made to the server. This file must be reviewed daily for trends in bad logins and maybe a password cracking attack. This file could be the only trace of a break in attempt to the server. Backup copies of this file will be made for historic purposes and trend analyzing. These backup copies should not be stored on the local server because if the server's security is compromised so will the backup files. This file will continue growing and eventually will fill the file system so it is necessary to crop the file on a periodic basis. We will crop the file when the backups of the file are accomplished. When removing the contents of the file, we will null the file (i.e. cat /dev/null >), so it will not be removed. If this file is removed the auditing functionality will not work because the utilities that place the data into the file will not create the file if it is missing. Along with the backup of the file, we will also create a report of all the invalid logon activity.

Since the OpenSSH product does not output the invalid logon activity to the btmp file we also analyze the system log file, /var/adm/syslog/syslog.log for this data. This information is also compiled for backup purposes and is placed into the report along with the data collected from the btmp file.

We will schedule this process to run daily. The UNIX security analyst will analyze the report for any indication of an attack.

Surrogate Activity Reporting and Archival

The /var/adm/sulog file is used to store all the information pertaining to the surrogate activity performed on the server. We will review this file on a daily basis and any abnormal events will be investigated thoroughly. This file should also be backed up just like the btmp and wtmp files described above so we will also archive the file at the same time. The report and archive is then sent to the central security server for compilation with all the other servers' data. This process will be scheduled to run daily.

Trust Relationship Reporting

Although we disabled the network services that utilize trust relationships for authentication, we will monitor all the files for any setup that may be on the server. This is important because the service may be enabled at one time so we would rather have it locked down in this case. Another reason is that an attacker may add a relationship to the server as part of a "toolkit" being used. In this case it could be the only indicator you will find on this attack. The reports generated will be from the /etc/hosts.equiv file and all the .rhosts files on the server. In the

event an .rhosts file is found it will correlate with the /etc/passwd file to identify the account that the .rhosts file is for. A separate report of changes made since the last run of the process will be created as well. This process will be setup to transfer the reports to the central security server and will be scheduled to run three times a week. On this server there should be no trust relationships setup.

FTP Access Reconciliation

For the same reason as the trust relationship reporting process we will also analyze the accounts with access to utilize the ftp network service to connect to the server, even though it is disabled. This process analyzes the /etc/passwd, /etc/ftpusers, and the /etc/shells files to determine which accounts on the server would have access to use ftp in the event the service was enabled. The final report is then transferred to the central security server to be matched against the master authorization list to ensure the only accounts that have the access are authorized. This process will be scheduled to run twice a week.

Root Account Access Reporting

Access to the root account is very sensitive because the account would have access to the entire UNIX OS. Although this process will not run on this server, the central security server will connect to this server as part of its process. The process is used to routinely monitor which accounts have access to the root account on all the HPUX servers. On these servers, root account access is given to authorized users by the security application, eTrust Access Control. This process analyzed the security database on each server and identifies all the accounts that have this access and if appropriate, through which program this access is given (i.e. the login program requires root access in order to access the shadow password files, so the security database would allow all identified users on the system to have root account access via the login program). The data collected is then used to generate a report of the current access. The data is also analyzed for any changes that were made since the last run of the process and a separate report is generated. This process is scheduled to run twice a week.

Section IX – Automated Security Review

This section identifies all the checks we will incorporate into the automated security review.

Introduction

Since going through a checklist of items to check on a couple hundred servers would take a very long time, we will automate the checks and then provide the reports of the items found that do not follow our policy.

This section identifies what checks will be made and why they are made. All the checks in this section have been programmed in either Korn Shell or Perl. The programs are then scheduled to be executed on the remote servers. All the checks below will be scheduled to run at least once a week with some being scheduled daily.

This section will only identify the checks we will run on this server with the automated security review. Consult the other sections in this document for other processes.

Section Conventions

The following documentation is setup to be easier to read by identifying why we are performing a step and what the actual step to perform is.

- Each step that is performed is identified with this type of bullet. This is the actual check we will have programmed.

Automated Account Administration Checks

The following checks are the checks we will make in regards to the accounts on the UNIX server.

UNIX Password File

The standard password file for HPUX servers is the /etc/passwd file. This file is in ASCII format and contains the following fields:

- Login name
- Encrypted password

- Numerical user ID (UID)
- Numerical group ID (GID)
- Reserved field used for account identification (commonly known as gecos)
- Initial working directory (commonly known as the home directory)
- Program to use as shell

To prevent any unauthorized modifications to this file, it must have restricted permissions assigned to it. Only accounts with an effective UID (EUID) of 0 (i.e. root account access) should be able to make any modifications to this file.

- ❑ The password file is to have ownership by the root account and the root group and have the permissions of 444 (read for all fields).

The UIDs of 17 and 18 are reserved for the Pascal Language operating system and the BASIC Language operating system respectively. Problems may be encountered if they are used for any other accounts.

- ❑ Check to ensure the UIDs of 17 and 18 are not being used on the server.

The login shell for the root account must be /sbin/sh. Other shells (i.e. Korn, C, etc.) are located under the /usr directory which may not be mounted at earlier run levels during the boot process. Changing this shell may prevent the system from coming up. The root account should also be the first account listed in the password file.

- ❑ Check to ensure the root account's login shell is set to /sbin/sh.

- ❑ Check to ensure that the root account is the first account listed in the password file.

The pwck (password file checker) utility in HPUX scans the password file and reports any inconsistencies. The following are the checks performed on the password file (information provided by HP).

- There are seven fields in the file and a colon delimits each one. All accounts listed in the password file are on different lines.
- There are no space characters in the file with the exception of the gecos field. A space character in any other field will either corrupt the password file on the system or prevent that user from logging in.
- The login (account name) field can be up to eight characters.
- The UID field must consist of an integer ranging from -2 to 2,147,483,646.
- The GID field must consist of an integer ranging from -2 to 2,147,483,646 and must also be a valid GID listed in the system group file.
- The initial working directory (home directory) field can be no longer than 63 characters.

- The program (user shell) field can be no longer than 44 characters.

- Use the pwck utility to check for any inconsistencies in the password file. Perform the appropriate steps to fix any problems found.

All accounts in the password file should have a unique UID. Processes on the UNIX server are performed using the UID. The password file exists to match the UID with an account name. If two accounts share one UID there will be no accountability for either account.

- Scan the password file to ensure all users have a unique UID.

All accounts listed in the password file should have a name associated with it in the gecos field. This name provides the accountability for the use of the account. In the event that the system password file is compromised, it should not contain data that will allow an attacker to identify which site the password file is from (i.e. do not place company specific data in the file). The reconciliation process that we will perform on all accounts on the network provides the verification that this name is valid. This process should be executed on all servers. This process takes all the names and accounts and matches it with the names and accounts in the central repository of accounts on the network.

- Check to ensure that all accounts listed in the system password file have a name associated with it in the gecos field.

- Check to ensure that the gecos field on all accounts does not contain company specific data that will allow an attacker to identify where the system password file was taken from.

- Check to ensure that the tcb collector process that is the front end for the reconciliation process is running on the server.

UNIX Group File

The standard group file for HPUX servers is /etc/group. This file is in ASCII format and contains the following fields:

- Group name
- Encrypted password
- Numerical GID
- List of users in the group (delimited by a comma)

To prevent unauthorized modifications to this file, it must have restricted permissions assigned to it. Only accounts with an EUID of 0 (i.e. root account access) should be able to make any modifications to this file.

- The group file is to have ownership by the root account and the root group and have the permissions of 444 (read for all fields).

The GID of 9 is reserved for the Pascal Language operating system and the BASIC Language operating system. Problems may be encountered if it is used for any other reason.

- Check to ensure the GID of 9 is not being used on the server.

The grpck (group file checker) utility in HPUX scans the group file and reports any inconsistencies. The following checks are taken from HP documentation to identify what their program does.

  - There are four fields in the file and a colon delimits each one. All groups listed in the group file are on different lines.
  - There are no space characters in the file. A space character will either corrupt the group file on the system or prevent the appropriate group memberships. (Tests with the grpck failed picking up space characters in the group name field)
  - The group names in the file must be unique.
  - The GID field must consist of an integer ranging from -2 to 2,147,483,646.
  - The GID for each group must be a unique integer.
  - Groups with no members are flagged.
  - All account names that are members of the group must also have an entry in system password file.
  - The number of characters in one line does not exceed the system maximum characters for one line. When more space is needed for the group another line entry is added with the same group name and GID.

- Use the grpck utility to check for any inconsistencies in the group file.

- Check the group file to ensure there are no space characters in the group name field.

The group file has a field to store an encrypted password for the group. Since we not allow users to log into a group, we will make this field null.

- Check to ensure the encrypted password field in the group file is either null or has an asterisk in it.

The group file, /etc/group, exists to allow users to change their group manually using the newgrp utility. However, there is a way to allow users to have multiple group rights which is accomplished by creating the /etc/logingroup file. This file has the same format as the /etc/group file. However, like all administrators we do not like to double the work load by maintaining two separate group files, so we will create the /etc/logingroup file as a symbolic link to the /etc/group file. The permissions on a symbolic link are ignored.

- Check to ensure the /etc/logingroup file exists and is a symbolic link to the /etc/group file.

Shadow Password Files

The HP Trusted Computing Base (TCB) system provides the shadowed password file functionality on an HPUX server. This system provides stringent password and authentication settings, auditing functionality, access control based on terminals and time settings. All HPUX systems on the network must utilize the shadow password file.

- Check to ensure the system is using the shadow password file. The files are located in /tcb and the encrypted password field in the system password file will contain an asterisk instead of an encrypted password.

To prevent unauthorized modifications to the TCB files, it must have restricted permissions assigned to it. Only accounts with an EUID of 0 should be able to make any modifications to the files. Due to the sensitivity of the contents of the TCB files, only an EUID of 0 should be able to read the files as well.

- Change all directories in the /tcb directory tree to have permissions to read and execute by the owner and execute only for the group. The owner should be set to root and the group should be set to root.

NOTE: The permissions on the TCB files for the account change to read and write for owner and group and read for other after a user logs into the server. This is done automatically by the operating system. This will not cause an exposure because only the root account and the root group are able to access the /tcb directory tree so therefore the files within the directory are only able to be opened by these accounts.

- Change all files in the /tcb directory tree to be owned by the root account and the root group. The file permissions should set to the maximum of read and write for the owner and group and read only for the world.

The authck (authentication database checker) utility in HPUX checks the structure and the fields in the TCB database on the system. All items not correct are reported. The checks include the user files and the terminal files. These checks also check the references of the account in the system password file, check for duplicate entries in the TCB database itself, determine if any account does not have a password set, etc.

- Use the authck utility to check for any errors in the TCB database.

Take note that the checks on the actual settings themselves are not performed with this automated process. Instead another process handles this process for all the servers at one time. Consult Section 8 for details.

Account Home Directories

Each account has a home directory that serves as the starting point on the file system for that account. The account's configuration files are also stored in this directory. On HPUX servers the home directories for accounts are usually stored in /home or /users with a few exceptions such as the root account. The account's home directory can be found by looking at the sixth field for the account in the system password file. Accounts on the server should not share home directories. These home directories store the account's configuration files and if the accounts were to share them they can modify each other's startup process.

- Check to ensure that home directories are not shared among accounts.

To prevent unauthorized modifications to the /home file system the permission settings should be read, write, and execute for the owner and read and execute for the group and the world. The owner should be set to root and the group should be set to datasec. The group datasec is used because the only accounts that belong to the group are the UNIX Security Analysts and is controlled. These settings prevent home directories from being created, removed, or altered.

- Check to ensure the settings on the /home directory are set to 755 and is owned by the root account and the datasec group.

On some servers, the home directory used for accounts could also be set to /users. For consistency on our UNIX servers on the network, the /users directory should be symbolic link to the /home directory if it exists on the server. This will give only one directory tree to secure and administer.

- Check to ensure that if the /users directory exists on the server, that it is a symbolic link to the /home directory.

To prevent unauthorized modifications to the account configuration files and the files stored by the account, the home directory for all accounts should not be writable by any account other than the owner and the owner should be set to the account that uses the home directory. This will prevent users from adding and deleting files from the account's home directory.

- Ensure all home directories on the server are only writable by the owner and the owner is set to the account that uses the home directory.

The configuration files (i.e. 'dot files') in the account's home directory should not be writable by any account other than the owner. The owner of the file should be the same as the account using the file and the group owner should be set to that account's primary group. To further restrict access to these files, the only permissions that should be allowed are for the owner of the file. When an account is added to the system, the permission should be set to read only for the owner (we will address the adding of accounts in Section 12). The default permission settings can be changed but should not be accessible by any other account other than the account that owns the file.

- Ensure all configuration files in the account home directories are only accessible by the owner and the owner is set to the account that uses the configuration files.

Account Configuration Settings

Each account has the ability to set environment variables within the UNIX environment. One environment variable that causes a security exposure is the PATH variable. This variable states the order and location where programs or binaries can be found and executed from. This variable is a matter of convenience so users either do not have to know or type in the full path where the programs are located. The exposure with this is that a malicious program (Trojan horse), may be executed in the place of the legitimate program. For example an account may have its path set to execute from /tmp and /usr/bin, in that order. When the user types in 'ls', the path is searched from /tmp first. If there is a file named 'ls' in that directory it will be executed instead of the 'ls' utility found in /usr/bin. For this reason root level accounts should have the minimum path settings or none at all. If there are settings in the path, the directories should not be writable by any account other than the root account. Other accounts on the server should follow the same constraints, but since these accounts would not have root access, it is not as major of an exposure.

- Ensure all directories in any root level account's PATH environment variable are owned by the account and are not writable by any account other than the owner.

- If possible, ensure all directories in all account's PATH environment variable are not writable by any account other than the owner.

Another shortcut used in the PATH environment variable is using a '.' as a directory. Placing a '.' in the path allows the search to be in the current working directory as well as the remainder of the path setting. This could cause an account on the server to inadvertently execute a file from the current directory instead of the directory that stores system utilities.

- Ensure all root level users do not have a '.' in the PATH environment variable.

- If possible, ensure all other accounts on the server do not have '.' in the PATH environment variable.

Another environment variable set on accounts is the umask setting. This setting sets the initial permission settings on a file that is created with the account. It masks the settings that are not to be set. For example when a umask is set to 027, a file created under it is automatically set to have the permission equal to 750. To prevent unauthorized modifications to files created by an account, the minimum umask setting for root level accounts should be set to 027. This setting disables all permission settings for the world and write permission for the group owner. If more relaxed permission settings are needed on a file, the user will have to make a conscience decision to change the permission settings on the file and therefore will know the exposures made to the file. The accounts that need a more relaxed umask setting should find 022 to be suitable. This setting prevents the write permission from being set on the group and the world.

- Check to ensure that the umask setting on all root level accounts is set to a minimum of 027.

- Check to ensure that the umask setting on all other accounts on the server is set to a minimum of 022. Document situations where this setting can not be met.

Account Security

On the UNIX server there are accounts for individual users, functional (group) accounts, and system accounts that are part of the Operating System. All accounts are vulnerable to attack and should be protected. The password security and password aging aspects of account security are covered elsewhere,

but there are other points of interest that should be implemented to ensure accounts are not going to be misused.

Accounts that log into the server and do not log off usually create inactive sessions. Depending on where these sessions originate, there may be a serious exposure. For example, if an administrator was to logon remotely to the server, obtain root account access, and then get up and go to lunch without logging out or securing the terminal the session originated, anyone could walk up to that terminal and receive instant root account access to our UNIX server. Since we do not want to rely on user terminals to provide the security needed for out server we constitute a time-out mechanism that will log off accounts that have been inactive for a specific time period. We will not get into all the details of this mechanism (called assassin) in this section because it is not a component of the automated security review. Instead refer to Section 12 for more details. We will get into the details of checking the settings and configuration of this application though because we do want them checked on a routine basis. The configuration files used in this inactivity checker must also be protected to ensure that unauthorized modifications are not made to the file. This configuration file should have the ownership set to the root account and the datasec group. The permission settings set on the file should be read, write, and execute for the owner, read only for the group, and no permissions set for the world. The executable files for this inactivity checker should also be protected to ensure that unauthorized changes are not made to them. The root account and the datasec group should own these files. The permission settings for these files should be read, write, and execute for the owner, read and execute for the group, and no permissions set for the world.

- Use the assassin utility to ensure accounts logged into the server are not inactive for longer than fifteen minutes. Document all special circumstances that do not abide by this policy.

- Regularly check exceptions placed into the configuration files of the inactivity checker program.

- Check to ensure that the configuration files for the inactivity checker are owned by the root account and the datasec group and the permission settings are set to read, write, and execute for the owner, read only for the group, and no permissions set for the world.

- Check to ensure that the executable files for the inactivity checker are owned by the root account and the datasec group and the permission settings are set to read, write, and execute for the owner, read and execute for the group, and no permissions set for the world.

The HPUX Operating System provides system accounts that either own specific binary files, run subsystems in the environment, or run automated processes on the server. Some of these accounts are not needed and should therefore be removed or disabled (this step is handled in Section 6). All system accounts (i.e. daemon, hpdb, lp, sys, etc.) that must remain on the server must be disabled through the TCB system, the initial password must be changed, the shell should be changed to a non-functional shell to prohibit login. Delete any account that exists with the name default, guest, or public. These account names are commonly used for visitors and are generally easier to break into.

- All system accounts that are to remain on the server should be in a disabled state. In the TCB settings on the account, ensure an administrative lock is set on the account.

- Change the shell program used by the system accounts that are to remain on the server. In the system password file, check to ensure that the shell program associated with the account is set to to either /bin/false, /dev/null, or another similar non-functional shell.

- Check to ensure the default, guest, or public accounts do not exist on the server.

All accounts that exist on the server should either have a valid shell or, when the account is disabled, a /dev/null or /bin/false (non-functional) shell. A valid shell is a binary file that should not be writable and is owned by root or a disabled system account. The shell must never be a SUID or SGID program because that will give permissions of the account that owns the file to the account executing the shell.

- Check to ensure that all accounts have valid shells.

- Check to ensure that all shell programs are not writable and are owned by the root account or a disabled system account.

- Check to ensure all shells being used do not have the SUID or SGID bit set.

In some instances a shell script or a locally developed program is used as the shell program for an account. In these cases, the program/script must be tested thoroughly for security exposures and must not be writable and be owned by either the root account or a disabled system account. These should also not have the SUID or SGID bit set on them.

- Check to ensure all accounts using shell scripts or locally developed programs being used as shell programs are not writable, are owned by the

root account or a disabled system account, and do not have the SUID or SGID bit set.

Login Auditing

The HPUX servers have auditing files that track all the login information for the server. They are the btmp, utmp, and wtmp files and are described in detail below. Take note that these files are dynamically updated by the login programs on the server and they are not in text format so special utilities are available in the operating system to view the data stored in them.

These audit logs should have restricted permissions to ensure unauthorized personnel can not modify these files. The root account and the root group should own the utmp file. The permission settings on the file should be set to read and write for the owner and read only for the group and the world. The btmp file should be owned by the root account and the group ownership should be assigned to the adm group. The permission settings should be set to read and write for the owner and group and no permissions set for the world. The adm account and the adm group should own the wtmp file while the permission settings are set to read and write for the owner and the group and read only for the world. Take note that the above permission settings are changed with the accounting software that is run on the server. The permission setting that should never be set on the server is the write permission for the world.

- Check to ensure the root account and the adm group own the btmp file with the permission settings set to read and write for the owner and group and no permissions for the world.

- Check to ensure the root account and the root group own the utmp file with the permission settings set to read and write for the owner and read only for the group and the world.

- Check to ensure the adm account and the adm group own the wtmp file with the permission settings set to read and write for the owner and group and read only for the world.

Additional information on these files can be found in Section 8.

Surrogate Auditing

The need arises sometimes to have an EUID for a time longer than the execution of one command. This need is fulfilled by using the 'su' utility that allows one account to assume the identity of another account (this is called surrogate). Due

to the sensitivity of this feature, there needs to be a way to audit when it is done. In the HPUX environment there is the /var/adm/sulog file, which is in plain text and displays all the users that have surrogated to another user. The root account and root group should own the sulog file. The permission settings should be set to read and write for the owner and no permissions set for the group and the world.

- Check to ensure the root account and the root group own the sulog file with the permission settings set to read and write for the owner and no permissions set for the group and the world.

Additional information and processing on this file can be found in Section 8.

Security Configuration File

The security configuration file is /etc/default/security. This file is used for many items but we will set the password history and the minimum password length setting. The password history will be set to six while the minimum password length will be set to eight. These settings will ensure that the same passwords are not being reused and that the passwords are a little stronger. The following is the format for these commands.

```
MIN_PASSWORD_LENGTH=8
PASSWORD_HISTORY_DEPTH=6
```

This file should be set to be owned by the root account and the sys group with the permissions set to read only by the owner.

- Check to ensure that the security file exists and has the entries for minimum password length and password history depth set.

- Check to ensure the root account and the sys group own the security file with the permission settings set to read only for the owner and no permissions set for the group and the world.

Automated File System Security Checks

The following checks are the checks we will make in regards to the integrity of the file systems on the UNIX server.

System Backups

When backups are done to tape it is very important not to leave the device for the tape able to be read by the world. If the device is left with open read permissions for the world, anyone on the system will be able to read the contents of the backup, regardless if the account that is being used would normally have access to the file or not. The device files are located in /dev/rmt and include all the files except for stape_config.

- Check to ensure that the device files used for backups does not have the read access for the world set on it.

SUID and SGID Programs

If a program or shell script has the SUID bit set, it means that when the program/script is executed, the account executing the file temporarily receives an EUID equal to the owner of the file. If the owner of the file is the root account and the file is 'broken out of', the account that executed the file still has an EUID of 0 (i.e. root). This is an issue with both system binaries provided by the vendor and locally developed programs and shell scripts.

If a program or shell script has the SGID bit set, it means that when the program/script is executed, the account executing the file temporarily receives an EGID equal to the group owner of the file. This is an exposure just like the SUID bit being set on the file.

If a directory has the SGID bit set on it, it means that when a file is created within the directory it automatically assumes the same group ownership as the directory.

Due to the intense testing and vulnerability checking needed it is generally not a good idea to have a shell script have either the SUID or SGID bit set on it.

- Check to ensure that there are no SUID or SGID shell scripts in use on the file system. If these types of files are needed, thorough testing and documentation must be provided.

In the event that SUID and SGID files are required (which they will be), it is important to ensure that these files do not change. The permission settings should not be set for the ability for the world to modify the file's contents. On SUID files the permission settings should only have the owner with the ability to modify the file.

- Check to ensure that SUID and SGID files on the server are not able to be written to by the world.

- Check to ensure that SUID files on the server are not able to be written to by any other account other than the owner.

A complete solution for monitoring SUID and SGID files along with other sensitive files can be found in Section 11.

Files Without Ownership

Accounts on the server may own various files all over the file system. The way the HPUX operating system assigns ownership to a file is by associating the UID of an account with the file. When using the 'ls' or any other file manipulation utility, the system cross-references the UID with the system password file to retrieve and then display the account name instead of the UID. This is done for convenience because an account name is easier to remember and associate with a user than the UID is. When an account is removed from the server (or the system password file), this association does not exist for any files that the account may have had ownership of.

When looking at these files, the ownership will be assigned to a UID instead of the account name because there will be no association available in the system password file. The security exposure for this exists when a new account is added to the server and is assigned the same UID as the account that was previously deleted. The files that the deleted account owned would now belong to the new account. To fix this problem the system must be scanned on a periodic basis to ensure all files on the file systems have an account that is assigned proper ownership. Although this step must be done, a proactive step must be done as well. This step would be accomplished when removing an account off the server. Before deleting the account, the file system should be searched for any files that the account may own. The files on the file system must either be removed or assigned a different account to own it.

In the event that a file is found that does not have ownership, it should be investigated why the file has no ownership. This could either be a routine maintenance problem or a trace of a break-in to the system.

- Periodically scan the file system for any files that have no account ownership. Either assign these files a new owner or delete the file off the file system. Investigate all events to ensure it is not a trace left from a break-in.

The above issue with the files without an account ownership assigned also holds true for accounts without a group assigned. Although this is an issue it is not as

probable and easier to fix than the account ownership. When performing the periodic scans of the file system and the scans done when removing a group the file should either be removed or assigned a different group ownership. If a file has an account that owns the file and no group assigned ownership, the group should probably be the account's primary group.

As well as the account ownership issue, any files found without a group ownership assigned should be investigated to ensure it is not a trace left behind from a break-in to the system.

- Periodically scan the file system for any files that have no group ownership. Either assign these files a new group or delete the file off the file system. Investigate all events to ensure it is not a trace left from a break-in.

Binary and Library Directories

These directories are used to either store executable files or the referenced libraries required for these files to be executed. Modifications made to the referenced libraries could seriously impact the security and availability of the server because the command could run something completely unexpected. To ensure that the binary (bin) and library (lib) directories are not modified, the permission settings should not have the write permissions set for the world or groups that are not administrative groups. Document any situations that can not abide by this rule.

- Check to ensure all bin directories do not have the write permission set for the world or a group that is not an administrative group.

- Check to ensure all lib directories do not have the write permission set for the world or a group that is not an administrative group.

Text Editors

The vi and ex utilities are used to edit text files in the UNIX environment. Both utilities utilize editor initialization files that can be placed into the EXINIT environment variables or in a file. This file is named .exrc and can be executed from the current working directory or the account's home directory. Because this file is searched for in the current working directory there is the possibility that an account will execute a .exrc file that was not intended to be executed. This file could be malicious which could have devastating consequences on the server if executed. The file system should be scanned regularly to ensure that the only .exrc files are located in accounts' home directories.

- Regularly scan the file system to ensure that the only .exrc files are located in accounts' home directories.

The emacs editor allows the execution of a start-up configuration file for the application. This file is named .emacs and allows the execution of commands in the emacs LISP language. These commands can be hidden in valid entries and may never be found. To eliminate this risk, the .emacs file should not be used on the file system.

- Regularly scan the file system to ensure there are no .emacs files.

System Log Daemon

The system log daemon is known as syslogd. This daemon reads log messages and places them in specific files, e-mails, other servers, etc. that is defined in the /etc/syslog.conf file. This configuration file should be protected to ensure that it is not modified without authorization. The bin account and the bin group should own the syslog.conf file with the permissions set to read only for the owner, group, and the world. The main log files that store auditing information are mail.log and syslog.log. Both of these files are stored in the /var/adm/syslog directory. This directory should have the ownership set to the bin account and the bin group and the permissions set to read and execute for the owner, group, and the world. The root account and the root group should own the log files with the permissions set to read only for the owner, group, and the world.

- Check to ensure that the bin account and the bin group own the syslog.conf file with the permissions set to read only for the owner, group, and the world.

- Check to ensure that the bin account and the bin group own the /var/adm/syslog directory with the permissions set to read and execute for the owner, group, and the world.

- Check to ensure that the log files stored in the /var/adm/syslog directory are owned by the root account and the root group with the permissions set to read only for the owner, group, and the world.

Disk Usage

When the file systems on the server utilize all the space allocated to them, the server will either stop functioning (users may not be able to log in) or programs running will not be able to write any more data. Either way, the file systems should never fill to capacity. Reviewing of the space available must be done on a periodic basis and when a file system is close to capacity it should either be

given more disk space or files that are not being used any longer should be deleted. The same applies to the utilization of inodes on a file system.

- Periodically check the file systems on the server to ensure they are not close to capacity on disk space or free inodes.

World Writable Directories

World writable directories pose a security threat because any account on the server would be able to modify the contents or delete the files stored in the directory. The file systems should be searched for all world writable directories. This list of directories should be analyzed to ensure that these directories need to have this permission set on it. If the directory does need the write permission set for the world the sticky bit should be set on the directory. The sticky bit enforces the policy that an account can only delete files that belong to it within the directory.

- Periodically check the file system for world writable directories. If the directory does not need to have this permission set on it, remove it.

- When a directory requires the world writable permission set on it, set the sticky bit setting on the directory.

Automated Network Security Checks

The following checks are the checks we will make in regards to the security of the network services on the UNIX server. We will not go into every network service because we disabled most of them earlier. Instead we will look at problematic services that have an uncanny way of coming back on. We will also delve into other network security issues we should be concerned with.

Banners

All network services that provide an interactive logon are to have the following legal banner present. This banner is to be displayed before the user authenticates. In today's legal world this banner is necessary but we also want to perform this step so we can replace the default banner that displays sensitive information about the server.

THIS IS A PRIVATE COMPUTER SYSTEM ---
USAGE MAY BE MONITORED AND UNAUTHORIZED ACCESS

OR USE MAY RESULT IN CRIMINAL OR CIVIL PROSECUTION

- Check all interactive network services to ensure that the approved legal banner is displayed before the user authenticates.

Modems

When setting up a security system in the UNIX environment it is important to protect all points of entrance. The UNIX systems all have network connections to the local network. However, some systems also have a modem installed. A modem uses the telephone lines to connect to another computer. The UNIX system allows a user to either dial-out from or dial-in to the modem. This introduces a whole new point of entry to the system, which circumvents the network security changes made, and must be secured from misuse.

One basic statement in regards to a modem is, if the communication device is not needed, either remove or disable it.

- Check to ensure that the modem configuration, if it exists, disables the device.

UNIX to UNIX Copy

The UNIX to UNIX Copy (UUCP) system is an older networking scheme that was used to connect two UNIX servers. UUCP allows the connection of two UNIX servers over regular phone lines so no new hardware is required. However, this system is old an inefficient with today's servers and lines so it is rarely used.

We disabled this service on the server earlier when we were deciding which services to disable. However, along with the disabling of the service, the entire UUCP system may be removed from the system. If this can not be done, the permission settings and/or the SUID settings on the programs should be removed. The uucp account and the nuucp user account should also be removed from the server.

- Check to ensure the UUCP is disabled and either all the subsystem's files were removed or the permission settings and/or the SUID settings on the subsystem's program files were removed. Check to ensure that the uucp and nuucp accounts were removed from the server.

Trusted Hosts

In the UNIX environment there is a way for UNIX servers to 'trust' each other. That is to say that accounts on one server will be able to logon, execute commands, and copy files to the other server without authenticating to the server. However, there are too many security implications to have all UNIX servers trust one another. The trusted host process uses IP addresses for authentication purposes so it is vulnerable to IP spoofing attacks. An IP spoofing attack consists of an attacker can send out IP packets that appear to come from a different server. If an attacker was able to compromise one system, any systems that have a trust relationship setup for that server could also be compromised without the attacker having to authenticate. For these reasons, the trust relationships in the network should be kept to a minimum.

In this situation we have decides to disable all the network services that utilize these trust relationships. However it is necessary to secure the files in the event the service is started.

The trust relationships in the HPUX environment are setup using the /etc/hosts.equiv file and the .rhosts files located in the accounts' home directories. These files should be controlled to ensure that only authorized changes are made to them. The hosts.equiv file contains a list of all the trusted hosts. The accounts on the remote server, except for the root account, can perform commands, copy files, or log in as the same account name on the local server. The .rhosts files work the same way with the exception that this trust relationship would be setup for individual accounts. Take note that a .rhosts file in the root account's home directory is the only way a trust relationship can be setup for the root account. These files should not have a '+' character in it. With this character, the trust is setup for all servers that can access the server. The .rhosts file can also be used by user's to allow access to their account from a different account. If needed, the service can be initiated with a –l option which will disable the .rhosts files except for the root account's file.

We will check the server to ensure that the root account's .rhosts file and the /etc/hosts.equiv file exists and is appropriately secured. We will also check to ensure that there are no entries in the files. We leave an empty file just in case a malicious user was to create one of the files. Since it is already on the system, the user must have that access that the file was setup as.

- Check to ensure that there are no .rhosts files in directories other that /root.
- Check to ensure that the /root/.rhosts and /etc/hosts.equiv files are null.

- Check to ensure that the root account and the root group own the /etc/hosts.equiv and /root/.rhosts files with the permissions set to read only by the owner and no permissions set for the group and the world.

Services File

The /etc/services file is used for client and server networking services to get a port number that the service will use for network transfer. This file lists the network services, which port it should use, and any aliases for the service.

The format of this file is an important item to check to ensure that it will work properly. The file is organized with each service having its own line. Each line must begin with a character that is not a space or tab. Each item with the line is separated by any combination of space and tab characters. A '#' character is considered a comment and will not be searched by any utility that checks this file for port numbers. Space and tab characters are allowed at the end of the line. This file should also be checked regularly for changes and to ensure that correct syntax is used but this step is not in the scope of the automated security review.

- Check the services file to ensure that correct syntax was used.

The port numbers 0 through 1023 are considered privileged ports and accounts will need an EUID of 0 to start up any service running these port numbers. This design is used to ensure accounts will not be able to eavesdrop or startup any of the privileged services. Take note that this design is purely a UNIX standard and accounts on an NT server would be able to start a trojan horse privileged service.

The entries in this file do not mean the service is started up (some of the services listed can not even be started in the HPUX environment), instead it is used as a name resolution service for network services. If the service is definitely not going to be used as a client or server on the system it can be deleted from the file. However, the file is also used to ensure that a new network service that is defined in the file does not share the same port number as any network services that may conflict at some point.

Due to the sensitivity of this file, it should be protected from any unauthorized modifications. The bin account and the bin group should own the services file with the permissions set to read only for the user, group, and the world.

- Check to ensure that the bin account and the bin group own the services file with the permissions set to read only for the user, group, and the world.

Network Services Daemon Configuration File

The /etc/inetd.conf file is used as the configuration file for the network services daemon (inetd). If modified an attacker can start specific network services that have been turned off, place various arguments on the startup of the network daemons so they will function differently, or even run a command or program. The attacker would not need root access because the inetd daemon is ran with the root account and therefore any modifications to the file will also be ran with the root account. The bin account and the bin group should own the inetd.conf file and the permissions should be set to read only for the owner, group, and the world. Along with the securing of the inetd.conf file is the securing of the directory that the file is stored in, /etc. If an attacker is able to write in the directory the inetd.conf file can be replaced with a different file. The bin account and the bin group should own the etc directory with the permissions set to read and execute for the owner, group, and the world.

- Check to ensure that the bin account and the bin group own the inetd.conf file with permissions set to read only for the owner, group, and the world.

- Check to ensure that the bin account and the bin group own the etc directory with permissions set to read and execute for the owner, group, and the world.

File Transfer Protocol Service

The File Transfer Protocol (ftp) service allows the transfer of files between two servers. The 'ftp' utility is the client utility and the ftpd daemon is the server program. When connecting to the ftpd daemon the port used is port 21, which is also used to send commands. Sometimes the data is transferred using port 20, however, most of the time the server and client mutually negotiate on using a port above 1024 (i.e. a non-privileged port). Take note that this service transfers all data, including the password, in plain text across the network.

Although we decided to disable this service, it is still necessary to secure it because an administrator may one day get fed up with the alternative and enable it "only for one transmission" and then of course forget to disable it again. So we will look at this service and secure it as though it is running. Take note that this is true for all services but this one is especially problematic. Our first check would be to ensure that the service is not running.

- Check to ensure that the daemon, ftpd, is either commented or non-existent in the inetd.conf file.

Like all network services there should be some type of logging performed when users utilize the service. When users logged into the server using the ftp service,

the login is recorded in the wtmp file. However there is also a logging mechanism that can be used. To utilize this, the ftpd daemon should be started with the '-l' option. This option will send the logon information to the syslog file on the server. The initialization for this daemon is located in the inetd.conf file.

- Check to ensure that the ftpd daemon is initiated with the '-l' option in the inetd.conf file.

When an account logs into the server using the ftp service, the password file and the TCB files are checked to ensure the account has authenticated correctly. After that the /etc/shells and the /etc/ftpusers files are checked. The shells file is checked to ensure that the shell program assigned to the account in the password file is a valid shell. The shells file should have only valid shells listed and should be owned by the root account and the root group with the permissions set to read only for the owner, group, and the world. The ftpusers file is checked to ensure that the account is not listed in it. If an account is listed in the ftpusers file it means that the account will not be able to use the ftp service to connect to the server. The ftpusers file should contain all system accounts and all user accounts that do not require this access. The reason this access is limited is because the ftpd may have another vulnerability found and the server will have already limited the access to the minimal number of accounts. The ftpusers file should be owned by the root account and the root group with the permissions set to read only for the owner and the group and no permissions set for the world.

- Check to ensure that the /etc/shells file only contains valid shells.

- Check to ensure that the /etc/shells file is owned by the root account and the root group with permissions set to read only for the owner, group, and the world.

- Check to ensure that the /etc/ftpusers file contains all system accounts and all user accounts that do not require the access.

- Check to ensure that the /etc/ftpusers file is owned by the root account and the root group and the permissions are set to read only for the owner and the group and no permissions set for the world.

In the HPUX environment there is the ability to setup the ftp service as anonymous server. This means that personnel would not have to authenticate to the server to access the data that is stored on the server. The anonymous ftp service is not used on this server for obvious security reasons. The password file should not contain the ftp account and tests to ensure the service is not running should be completed.

- Check to ensure that the ftp account is not on the server.

- Test the anonymous ftp service to ensure it is not functioning on the server. This is accomplished by opening an ftp session to the server with the account anonymous and a valid formatted e-mail address.

Telnet Service

The telnet service gives the user a 'virtual terminal' session on the remote server. This session is a logon to the server that goes over port 23. However, all transmissions over the network are done in plain text. Because anyone on the network can use a packet sniffer and view traffic, an account that logs on to the server utilizing the telnet service is vulnerable to a password stealing attack.

This service is also problematic because it may also be enabled temporarily and then left on. So we want to check to make sure this service is not enabled.

- Check to ensure that the daemon, telnetd, is either commented or non-existent in the inetd.conf file.

Trivial File Transfer Protocol Service

The trivial file transfer protocol (tftp) service is a file transfer protocol that utilizes UDP on port 69. This service provides very little security and should not be used. Check to ensure that this service is not started on the server.

- Check the inetd.conf file to ensure that the tftpd daemon is either commented out or removed.

This service is identified here as a check because the Ignite-UX backup application opens it back up. Other then ensuring it is disabled we must check to ensure that the access is limited to the ignite directories.

- Check the inetd.conf file to ensure that the tftpd daemon is limited to the appropriate directories.

Automated Root Account Security Checks

The following checks are the checks we will make in regards to the security of the root account. The following are not all the checks we make but instead is in addition to the other checks found in the automated security review.

Root UID

This component of the automated security review delineates the security that should be in place to secure the root account. However, all tasks in the HPUX environment are identified with the UID. This means that accounts should not share the same UID because it provides no accountability for either account. This is especially true for the root account, there should be no other account that has the same UID as the root account (i.e. 0).

- Scan the password file to ensure the root account is the only account that has a UID of 0.

Root Account Access

Due to the sensitivity of the root account and the limitless access it has in the HPUX environment, it is best to limit the access available to this account. There are several ways to limit the access to this account and they will be discussed below.

Personnel will not log into this server directly with the root account. Instead, personnel will log into the server using their individual account and use the 'su' (HPUX surrogate) or 'sesu' (eTrust Access Control surrogate) utilities to surrogate to the root account. This action adds to the accountability of actions performed with the root account because the actions would be traceable to a user or small group of users. To ensure that users do not directly log on to the server as the root account we will utilize the /etc/securetty file. This file, if it exists, only allows the root account to log in from the terminal(s) listed. The only entry that should exist in this file is the console. This will prevent the root account from being logged into from any terminal with the exception of logging in from the console (no network logins). The securetty file should have the permission settings set to read only for the owner and group and nothing for the world with the owner being the root account and the group owner being the root group.

- Check to ensure the securetty file exists and is read only by the owner and group and nothing for the world with the owner being the root account and the group being the root group. The only item in this file should be 'console'.

The problem with the securetty file is that CDE sessions will ignore it. To stop the logon through CDE we need to modify the /etc/dt/config/Xstartup file to include and exit command if the variable USER is root.

- Check to ensure the Xstartup file exists and is read only by the owner, group, and world with the owner being the root account and the group being the root group.

Login Auditing

There is also a need to log all login activity to the root account. To gain some accountability to the account there must be a process in place that will track the person who logs in with the account and what the account was logged into for. For this we will check the server to ensure that it has the proper root account tracking process in place and that it is properly configured.

This root account tracking process is basically a script that is called from the root account's profile when the account is directly logged into (i.e. not a surrogate action). This script prompts the user who they are and what they are doing. The script then compiles various data on the session and then notifies the appropriate personnel of the action. The /etc/syslog.conf file requires modifying because the script uses the syslog logger to record the activity. As well as using the logging facility we will also use sendmail to alert the appropriate personnel via e-mail. Although this method is not foolproof, when backed by management and policy it is effective because it holds personnel accountable.

- Check to ensure the root account tracking process is properly configured.

Job Scheduling

Due to the amount of access the root account has, actions performed with the account have the potential to crash the server, damage the file systems, or make any other unauthorized changes to the system. The job scheduling facility on the server allows the execution of commands by accounts on the server at any time of the day. There is an exposure when one of these programs either has a design flaw or is modified. In this event, the program has the potential to change or remove anything on the server. To limit this exposure the root account should only be used for programs that require that access. Routine activities that do not require this level of access should never be setup to use the root account. Also, jobs that are setup to run using the root account should be examined for security flaws and the ability for accounts, other than the root account, to make changes to the program. The jobs should also be analyzed for alternatives to using the root account (i.e. sudo, etc.). Consult the next component, Job Scheduling, for further details on the job scheduling facility on the system.

- Check all programs set to run in the job scheduling facility using the root account are only writable by the root account or a limited access group (i.e. datasec).

Automated Job Scheduling Security Checks

The following checks are the checks we will make in regards to the security of the job scheduling facilities on the UNIX server.

There are three utilities that are used in the UNIX environment for job scheduling. They are at, batch, and cron. Each one of these utilities are described below.

at

The 'at' utility is used to schedule jobs to execute at a specified time. This is a one-time execution that will only occur at the specified time. When the job is placed into the queue it is stored in the /var/spool/cron/atjobs directory. Stored in this directory are the commands entered and the environment variables that are to be used during the execution of the job. Accounts are permitted to schedule jobs using the 'at' facility if their account name exists in the /var/adm/cron/at.allow file. However, if the at.allow file does not exist then the /var/adm/cron/at.deny is checked for accounts that do not have access to the 'at' facility. If neither of these files exist on the server then only the root account has access to schedule a job using the 'at' facility.

We will need to check the server to ensure that the at.allow file exists. The list of accounts in the at.allow file should be kept to a minimum because regular user accounts should not have this access. The bin account and the bin group should own the at.allow file and the at.deny file should not exist. The permission settings for the at.allow file should be set to read only for the owner, group, and the world.

- Check to ensure that the at.allow file exists and contains only the allowed accounts.

- Check to ensure that the bin account and the bin group own the at.allow file with the permissions set to read only for the owner, group, and the world.

- Check to ensure that the at.deny file does not exist on the server.

The file /var/adm/cron/.proto is added to the program submitted to the 'at' facility. Adding this file makes the environment used to execute the program the same as the account's environment. Due to the sensitivity of this file, it should be

protected from unauthorized modification. The bin account and the bin group should own the .proto file with the permission settings set to read only for the owner, group, and the world. This file should also not be modified from its original content. If the file is required to be modified than the special circumstances are to be documented.

- Check to ensure that the bin account and bin group own the .proto file with the permissions set to read only by the owner, group, and the world.

- The contents of the .proto file should consist of the regular account. If the file needs to be modified than document the situation.

The /var/spool/cron/atjobs directory is the repository where future jobs are stored. This directory must also be protected from unauthorized modification. Set the owner for the atjobs directory to the bin account and the bin group. The permission settings for this directory should be set to read and execute for the owner, group, and the world.

- Check to ensure the atjobs directory is owned by the bin account and bin group with the permissions set to read and execute for the owner, group, and the world.

The programs queued for execution by the 'at' facility are executed by the cron daemon that is running on the server. There are other configuration files that are used by the cron daemon that will be described below.

batch

The 'batch' utility is used to schedule a job to execute immediately or when the system resources are available. All items that were described above for 'at' facility apply to the 'batch' utility as well. All precautions described above must also be set for this utility.

cron

The 'cron' facility on the server allows the execution of commands/programs at specific times and dates. Unlike the 'at' and 'batch' utilities discussed earlier, this facility allows the execution to continue whenever it is scheduled, not just one time. Like the other two utilities, these programs are automated and should be secured to ensure the system will not be misused.

Just like the at.allow file that was described above, there is a similar situation used for the cron facility. The /var/adm/cron/cron.allow file should exist on the

server. The list of accounts in the cron.allow file should be kept to a minimum because regular user accounts should not have this access. The bin account and the bin group should own the cron.allow file and the /var/adm/cron/cron.deny file should not exist. The permission settings for the cron.allow file should be set to read only for the owner, group, and the world.

- Check to ensure that the cron.allow file exists and contains the minimal number of accounts.

- Check to ensure that the bin account and the bin group own the cron.allow file with the permissions set to read only for the owner, group, and the world.

- Check to ensure that the cron.deny file does not exist on the server.

The crontab (individual account execution schedules) files should only be able to be changed by either the root account or the account that the file is named. These files are stored in the /var/spool/cron/crontabs directory. There is one file for each account and they are named the same as the account name. When executing the program from the crontab, the program assumes the same permissions and access levels as the account that the file is named. Just like the atjobs directory above, this directory must be restricted to prevent the unauthorized modification of the files. The bin account and the bin group should own the crontabs directory with the permission settings set to read and execute for the owner, group, and the world. The cron directory (parent of the crontabs directory) should also have the same permissions and ownership set as the crontabs directory. The files stored within the crontabs directory should be owned by the root account with the permissions set to read only by the owner and no permissions set for the group and the world. The 'crontab' utility is a SUID program and owned by the root account. When executed, this program has the ability to modify the contents of the crontab files.

- Check to ensure that the /var/spool/cron and /var/spool/cron/crontabs directories are owned by the bin account and the bin group with the permissions set to read and execute for the owner, group, and the world.

- Check to ensure that the files stored in the crontabs directory are owned by the root account and the permissions are set to read only by the owner and no permissions set for the group and the world.

The crontab files for all accounts should be checked to assure that the programs scheduled for execution by the account can not be written by any unauthorized group. As a general rule, the programs must never be able to be written by the world. The programs should also not be able to be modified by any unauthorized groups. The contents of the crontab files should also be checked for correct syntax.

- Check the entries in the individual crontab files to assure that the programs being executed are not able to be modified by the world and any unauthorized groups.

- Check the syntax of the crontab to assure that it is correct.

The /var/adm/cron/queuedefs file is used to configure the 'at', 'batch', and 'cron' queues. The options available are the maximum number of jobs that can run at the same time, the nice value given to the programs when they are executed, and the number of seconds to wait when too many jobs were executing at the same time. This file should not be modified by anyone other than a system administrator so the file permissions should be restricted. The bin account and the bin group should own the queuedefs file with the permission settings set to read only by the owner, group, and the world.

- Check to ensure that the bin account and the bin group own the queuedefs file with the permissions set to read only by the owner, group, and the world.

The /var/adm/cron/log file stores all actions taken by cron. This log is important to determine which programs were executed at a specific time using the 'cron' facility. This log should be protected to assure that it can not be modified without authorization. The root account and the root group should own the log file with the permission settings set to read and write for the owner and read only for the group and the world.

- Check to ensure that the root account and the root group own the log file for the cron facility and the permissions are set to read and write for the owner and read only for the group and the world.

The directory /var/adm/cron contains the configuration information that the cron daemon uses. This directory should be protected to assure that it and its contents are not changed by an unauthorized account. The bin account and the bin group should own the cron directory with the permissions set to read and execute by the owner, group, and the world.

- Check to ensure that the /var/adm/cron directory is owned by the bin account and the bin group and the permissions are set to read and execute by the owner, group, and the world.

Section X – Apache Web Server Configuration

This section is used to document the configuration of the Apache Web Server that will be used on this server.

Introduction

The Apache Web Server is the open source web server that is freely available. It is also a very widely used web server application. Hewlett Packard has ported the open source application to the HPUX platform. This is important because it saves us time by not having to compile it for our platform. Since features were added to make full use of the OS, the application will work better and hopefully more secure.

We will be using this web server for reporting purposes and general documentation requirements for the UNIX security group. We will use Secure Socket Layer (SSL) on the whole web site to keep all traffic encrypted. Instead of using the standard .htaccess files we will utilize the SSL realm security. We will also have the user at a browser session have the ability to connect to the server name, which is by default unencrypted, and the we server will redirect automatically to the encrypted web site.

The following documentation is used to implement the Apache Web Server on this server and to ensure adequate security measures have been taken.

Configuration

The Apache product was installed earlier in this document so now we will go through the steps necessary to configure it properly. All of the configuration files are stored in the /opt/hpws/apache/conf directory. By default, the files are configured very well but the following will go into changes we need to make to make the web server work for us.

httpd.conf

The httpd.conf file is the main configuration file for the Apache web server. The following are the changes we will be making to this file. Since the server is setup by default the following are only the changes. There are other settings that are required but have already been set correctly. We will not be delving into detail on the items we will be leaving alone.

- **mod_perl** – The mod_perl module is used to speed the execution of programs in the Perl programming language. Since our Common

Gateway Interface (CGI) programs will be written in this programming language we will remove the comment from the following line.

```
LoadModule perl_module        modules/mod_perl.so
```

- **DirectoryIndex** – This setting tells the web server what constitutes an index page in the directory. We comment the default entry and place the following line in its place. By default we do not want PHP and CGI programs being used as index pages.

```
DirectoryIndex index.htm index.html
```

- **Apache Manual** – By default the manual for the Apache web server is included. This documentation is not included in the document root tree. We disable this documentation because it is not required for the scope of this web site and may give too much information to potential attackers. We comment the following lines.

```
AliasMatch ^/manual(?:/(?:de|en|fr|ja|ko|ru))?(/.*)?$ "/opt/hpws/apache/manual$1"
<Directory "/opt/hpws/apache/manual">
    Options Indexes
    AllowOverride None
    Order allow,deny
    Allow from all
    <Files *.html>
        SetHandler type-map
    </Files>
    SetEnvIf Request_URI ^/manual/de/ prefer-language=de
    SetEnvIf Request_URI ^/manual/en/ prefer-language=en
    SetEnvIf Request_URI ^/manual/fr/ prefer-language=fr
    SetEnvIf Request_URI ^/manual/ja/ prefer-language=ja
    SetEnvIf Request_URI ^/manual/ko/ prefer-language=ko
    SetEnvIf Request_URI ^/manual/ru/ prefer-language=ru
    RedirectMatch 301 ^/manual(?:/(de|en|fr|ja|ko|ru)){2,}(/.*)?$ /manual/$1$2
</Directory>
```

- **PHP File Association** – Since we will not be using the PHP programming language on this web site we will comment the following lines that set the PHP filter.

```
<Files *.php>
    SetOutputFilter PHP
    SetInputFilter PHP
</Files>
```

- **Virtual Host** – This web site will have two web servers listening. The normal http (port 80) will be setup as a virtual host to the directory /opt/hpws/apache/dummy_web. This directory will have an index page and that is it. The index page can be anything because all access transferred here will be redirected by the web server to the encrypted web pages under /opt/hpws/apache/htdocs. The virtual host for the encrypted web pages will be setup in the ssl.conf file. We add the following lines to the configuration file. The *<server name>* is the fully qualified name not just the server name.

```
NameVirtualHost <server IP address>

<VirtualHost <server name>:80>
   ServerAdmin www@<server name>
   DocumentRoot /opt/hpws/apache/dummy_web
   ServerName <server name>
   ErrorLog logs/<server name>-error_log
   CustomLog logs/<server name>-access_log common
   Redirect / https:// <server name>/index.htm
</VirtualHost>
```

- **HPUX Apache Web Server Documentation** – Hewlett Packard also included their own documentation for the web server. We will disable this as well by commenting the following lines.

```
<IfModule mod_alias.c>
   # Allows access to hp_docs from a browser
   # use http://yourserver.com/hp_docs
   # To prevent access, comment the following lines out.
   Alias /hp_docs "/opt/hpws/hp_docs"
   <Directory "/opt/hpws/hp_docs">
      AddHandler cgi-script .cgi
      Options ExecCGI FollowSymLinks MultiViews Indexes
      Order allow,deny
      Allow from all
   </Directory>
</IfModule>
```

- **HPUX Integration** – Hewlett Packard also integrated files from the OS onto the web server. Since we do not want to give more information than we need to, we will disable this as well by commenting the lines below.

```
Alias /web "/usr/share/web"
Alias /doc "/usr/share/doc"
Alias /hp-ux "/usr/share/doc"
```

```
<Directory "/usr/share/doc">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
    AddHandler server-parsed .shtml
    AddType text/html .shtml
    Options +Includes
</Directory>
```

ssl.conf

This configuration file provides the necessary information for the Apache web server to serve web pages over an SSL encrypted tunnel. Like the above configuration file, we will only document the changes required to be made from the default file included with the installation.

- **SSLRandomSeed** – When using encryption it is necessary to have a random number generator. This is why we installed the Strong Number Generator application earlier. This application created the /dev/random and /dev/urandom files. For this implementation we will utilize the /dev/urandom file. First we need to comment the following lines.

  ```
  SSLRandomSeed startup builtin
  SSLRandomSeed connect builtin
  ```

  Next we will remove the comment from the following lines to make them active.

  ```
  SSLRandomSeed startup file:/dev/urandom 512
  SSLRandomSeed connect file:/dev/urandom 512
  ```

- **Virtual Host** – We create a virtual host for the pages that will be served through the SSL tunnel here. These files will be everything under the /opt/hpws/apache/htdocs directory. Earlier we setup the virtual host in the /opt/hpws/apache/dummy_web directory that automatically redirects traffic to this virtual host. We will first comment the following line.

  ```
  <VirtualHost _default_:443>
  ```

  Next we add this line to replace that entry. The <*server name*> is the fully qualified server name.

  ```
  <VirtualHost <server name>:443>
  ```

- **Certificate Authority** – We will not be paying for an external Certificate Authority (CA) to issue the certificate files. Instead we will be creating our own. The instructions for creating our own certificate files are included below. First, remove the comment from the following line.

  SSLCACertificatePath /opt/hpws/apache/conf/ssl.crt

  Next add the following line.

  SSLCACertificateFile /opt/hpws/apache/conf/ssl.crt/ca.crt

- **SSLOptions** – These options are used to configure the SSL engine. The main configuration line for this is initially commented. We will leave the following line commented.

  #SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars +StrictRequire

  Then we will add the following line to ensure that the proper SSL tunnel is used for the directories with SSLRequireSSL set.

  SSLOptions +StrictRequire

- **Authentication Setup** – While still within the VirtualHost definition (i.e. before the </VirtualHost>) we setup the directories that will require authentication to access. The following is an example of this setup.

```
<Directory "/opt/hpws/apache/htdocs/procedures">
  SSLRequireSSL
  AuthUserFile /opt/hpapache2/ssl/.passwd
  AuthGroupFile /opt/hpapache2/ssl/.group
  AuthType Basic
  AuthName "UNIX Security Access"
  require group infosec
  ErrorDocument 401 /access_denied.htm
</Directory>
```

  The example provided above secures the web pages under /procedures on the web site. The password and group files that are to be used for this authentication is set to the files listed. The password file has the format:

  <account name>:*<encrypted password>*

  The encrypted password above is the same encryption as the HPUX OS performs for the TCB files. The group file has the following format:

<group name>: *<space delimited list of accounts>*

The accounts that are listed in the groups are looked up in the password file for the authentication information. This allows us to secure based on groups instead of listing all the accounts that should have access.

The AuthName is what the pop-up authentication window will prompt. This is the name of the Security Realm. The ErrorDocument is set to the error page if the authentication failed. You should create a web page for this function so the users will know why they were taken to that page.

SSL Certificates

If there is already a CA on the network, the following steps do not have to be accomplished. Since this web site will be for the UNIX security group we will create our own certificates. To perform this we use the SSL Certificate Generation utility (mkcert.sh) provided with the application.

/opt/hpws/apache/util/mkcert.sh --custom

The custom option above will generate a certificate for a CA and the server. This process is very self-explanatory and will walk you through the creation. The following are the prompts during the creation.

1. Enter the Apache Location [/opt/hpws/apache] :

   This is the main directory for the Apache application. We keep this at the default.

2. Enter the Certificate(s) Location [/opt/hpws/apache/conf] :

   This is where the certificates will be stored. This is also kept at the default.

3. Signature Algorithm ((R)SA or (D)SA) [R]:

   This selection is for the signature algorithm used. We will keep the default of RSA. This step will generate the RSA private key and output it to the file, ca.key.

4. The next set of options will prompt for a lot of information, such as location and various names. This information is used to create the certificate signing request for the CA we are creating. This information will be stored

in the ca.csr file. This step will also create the certificate for the CA and will store the output in the ca.crt file.

5. Encrypt the private key now? [Y/n]:

   The process will ask if you want to encrypt the RSA key that was stored in ca.key. We answer yes to this and then are prompted to enter a pass phrase. This is a phrase, not a password, so we make it long. The RSA key is now encrypted with the Triple DES algorithm and a pass phrase.

   The CA certificates are now complete. The process will now move onto the server configuration.

6. The process now continues to process by creating another RSA private key. This time the key will be used for the server configuration. This key is stored in the server.key file.

7. We now go on to create the certificate signing request for the server. The process will prompt for information regarding the server this time. The output will be stored in the server.csr file.

8. The certificate for this server will be stored in the server.crt file. This certificate will be signed by the CA that we created earlier. Since we added a pass phrase earlier, the system will prompt for the pass phrase so it can decrypt the file to retrieve the RSA private key for our CA.

9. Encrypt the private key now? [Y/n]:

   This time the server is prompting to encrypt the server's RSA private key stored in the server.key file. We answer yes here and then will be prompted for another pass phrase. This file is encrypted using the Triple DES algorithm and the pass phrase entered. Remember this is a pass phrase, not a password, so make it long.

10. The following report is then displayed on the screen. We have successfully created the necessary files.

    RESULT: CA and Server Certification Files

    o /opt/hpws/apache/conf/ssl.key/ca.key
      The PEM-encoded RSA private key file of the CA which you can use to sign other servers or clients. KEEP THIS FILE PRIVATE!

    o /opt/hpws/apache/conf/ssl.crt/ca.crt
      The PEM-encoded X.509 certificate file of the CA which you use

to sign other servers or clients. When you sign clients with it (for SSL client authentication) you can configure this file with the 'SSLCACertificateFile' directive.

- o /opt/hpws/apache/conf/ssl.key/server.key
  The PEM-encoded RSA private key file which you configure with the 'SSLCertificateKeyFile' directive (automatically done when you install via APACI). KEEP THIS FILE PRIVATE!

- o /opt/hpws/apache/conf/ssl.crt/server.crt
  The PEM-encoded X.509 certificate file which you configure with the 'SSLCertificateFile' directive (automatically done when you install via APACI).

- o /opt/hpws/apache/conf/ssl.crt/server.csr
  The PEM-encoded X.509 certificate signing request of the server file which you can send to an official Certificate Authority (CA) in order to request a real server certificate (signed by this CA instead of our own CA) which later can replace the /opt/hpws/apache/conf/ssl.crt/server.crt file.

Congratulations!! You can now establish a SSL enabled server.

Now the certificates have been created but we need the ca-bundle.crt file to be created. To create this file you just follow the Apache documentation step by step. Since there are no steps outside of that documentation it will not be placed in this document. This documentation is included with the application. Consult the file /opt/hpws/apache/hpws_docs/ ssl.admin.guide for additional details. Take note that we removed the browser so the certificate database was needed to be built on another server.

Apache Web Server Startup

All the configuration has now been complete. We use the apachectl utility to start the application.

/opt/hpws/apache/bin/apachectl startssl

We now see the following:

```
HP-UX_Apache-based_Web_Server/2.0.48 mod_ssl/2.0.48 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass phrases.
Server <server name>:443 (RSA)
Enter pass phrase:
```

At the prompt we enter the pass phrase that we created for the server's RSA private key.

Mod_Perl Problems

The Apache server application we installed has a dependency on an older version of the Perl programming language. Since we require the another version of Perl we need to go back to If you installed version 5.8.0 on this server instead of the version 5.6.1 you will have to remove the newer version and install the older version. This dependency is built in to the mod_perl functionality in the web server. To remove the newer version of the Perl programming language, execute the following command.

/usr/sbin/swremove perl

We install the depot file to the server with the following command.

**/usr/sbin/swinstall -s **
/tmp/depots/perl_B.5.6.1.E_HP-UX_B.11.11_32.depot perl

Password Synchronization

Although you can use utilities to assign and reset accounts for the web server, they will not have the same password aging and constraints as found on the OS. Since we already have processes that compile the TCB files from the servers, we can retrieve the TCB files from our primary security server. This server would have all the personnel that would be requiring access to the web site at any point in the future.

So we develop a process that will take the /opt/hpapache2/ssl/.passwd file and get the account names. These account names are then used to retrieve the encrypted password in their TCB file. The password file is then recreated with the current passwords. We set this up in cron job scheduling facility to update the password every four hours.

Log Monitoring

Like all applications it is very important to monitor the log files on a routine basis, preferably daily. We develop a process that produces a report on the day's activities and from that report we can modify the process to ignore the items we are not interested in. Over time you will have a report that is just errors. The log

files for the Apache web server are located in /opt/hpws/apache/logs. You would be most interested in the error_log file but the access_log file can be useful as well. We set this process up to run daily and a security analyst reviews the files for abnormal activity.

Reports on the Web Site

The main purpose of this web site is to host the reports that the UNIX security group is required to review and archive. To accomplish this we need to perform the following steps.

HTML Formatted Reports

The most important step is to figure out what the most pleasing layout for a particular report will be. For example we may want to add a table or some color to make particular items stand out better.

After the planning has been completed, we need to modify the process that creates the report so it will create it in HTML format. Since the teaching of HTML coding is out of the scope of this document, we will only touch on the major points of the web site creation.

- Browser Cache Problems – The browser at the workstation automatically caches the pages that are viewed so if that page was to be updated the browser will not show it. This is a problem because the CGI program we use changes the time stamp on reports after modifying them. To get around this problem we implement the following code in the <HEAD> section of the document.

  <meta Http-Equiv="Pragma" Content="no-cache">

- CGI Interface – We will utilize a CGI program that will update the current report with the comments we input. The program will also mark reports as finished, send the comments to a web page that contains all the comments the user has made, send the comments to a web page that contains all the comments that were made on that type of report, and create the confirmation page to confirm that the appropriate changes had been made.

  There is no way to pass the authentication data back to the CGI program so we protect the CGI program directory behind authentication. This allows us to report on who updated the report with comments and therefore will be able to hold people accountable.

- Comments – The ability to add comments helps it audit situations and helps hold personnel accountable for the actions performed. The comment field will be recorded with the report even when it goes into the archive. The complete example of the CGI process is included after the next bullet.

- Finished? – We add the ability to mark a report as finished so we can differentiate in the archive files, which reports have not been completed yet. Another process is developed that e-mails the appropriate personnel the reports that have not been completed. This process runs on a daily basis.

  The complete CGI process is as follows:

```
<form action="/cgi-bin/post.cgi" method="POST">
 <h3>Comments:</h3>
 <textarea rows="10" cols="120" name="comment" value="No Actions Needed">No actions required.</textarea>
 <p></p>
 <input type="submit" value="Log Comments">
 <input type="reset" value="Default Comments">
 <p></p>
 <input type="radio" name="finish_select" value="NO" checked>Not Finished
 <input type="radio" name="finish_select" value="YES">Finished
</form>
```

Implementation on the Group Web Site

To place the reports on the group web site, the process that created the report will transfer the file to this server to a previously defined directory. A process on this server will then take the file and place it in the appropriate directory on the web site. At the same time, the process will take the old current report and place it in the archive and then will recreate the archive main page. This main page will show the archived reports that have not been marked as finished in a different color so it will stick out better.

Section XI – File System Integrity Checks

This section is used to document the in-house developed file system integrity checking system. This system is used to ensure that the identified sensitive files on the servers on the network have not been modified.

Introduction

If an account on the server or an account that was broken into modifies a sensitive or critical file, undesired consequences could result. The results may consist of system crashes, password collector, other spoof programs (i.e. trojan horses), etc. When securing a system, the administrator needs to be assured that critical system files and programs have not changed. For this reason it is very important to check the contents of these files and programs on a routine basis to assure they have not been changed without authorization. The changes made include modification times, permission changes, file size, and changes to the file itself. It is very important to check these files and programs. An easier and more reliable method of checking the contents of files would be to check the cryptographic hash code of the files and programs. If the hash code is changed and no administrative action was performed then it is safe to assume that the file was changed by an unauthorized account. When this happens, it is imperative that the file is resorted to the last known unchanged version, which can be extracted from a backup. This is another important aspect of performing routine backups on the server, especially backups of critical system files. To perform this analysis of the file system an automated tool should be used. The baseline that will be used to perform the analysis should not be stored on the local system because it could get modified as well as the file or program on the system. The baseline should also be checked before and after a patch is installed on the system. If there are no deviations from the baseline before the patch is installed, any differences after the patch is installed were obviously made by the patch installation and should be trusted and a new baseline would be made.

Purpose of Application

This application is used to identify any sensitive files that have been changed and then alert the appropriate personnel. The application is also setup to allow the identification of who changed the file(s) and what changed within the file(s). The sensitive files that are identified are completely configurable on a per-system basis with a default file database installed on each system.

Why In-House Development?

There are quite of few applications on the market today that can be used to ensure sensitive files have not been modified. One of these is even the security application, eTrust Access Control. Unfortunately all the products have their problems or inadequacies. Some of the products are so widely used that attackers have automated tools to bypass them. Since the software currently available on the market cannot do everything that we want we go about and develop our own version. Writing this application in house will also enable the modification of the program with a quick turn around instead of waiting for a company to get around to it and release a new version of their application. The application will also be more customizable for the environment, which will make the application easier to use and provide reports that are also customized for the environment.

To be able to quickly identify what files have changed on a server will aid in intrusion detection and therefore will enhance the security of the systems. The added ability of identifying who changed the file, when the file was changed, and what changed in the file will add a dimension to the application that most third party products do not provide.

Another added benefit of coding the application yourself is that you can easily add additional features to the application. This helps because additional algorithms may come out that you want to incorporate into the design of the file integrity checker. With the layout of this application it is very easy to implement additional checks into it.

Data Repository

There is no data stored on the servers on the network. This is done for two reasons. The first reason is that if an attacker was to compromise a system, there would be no trace of a file integrity application existing on the server. The second reason is that since we will be keeping all the data on a central UNIX security server it will be easier to manage, control, and secure. The central server will store a copy of the application, the baseline files, and the configuration files.

The eTrust Access Control security application will be used to monitor access to the sensitive files that are marked for monitoring. Due to the system resource usage of this method it will not be recommended to monitor every file. When the log files for this security application are required they will usually be found on the remote system. In the event that the remote system's log files were rolled over due to size constraints, the additional log files can be found in the archived log files on the central security server.

Application Usage

There is a great importance of ensuring the integrity of the files being used on the system. However, there are times when specific sensitive files need to be changed. At these times a new baseline for the system will be created. This application should be used before a patch is installed to ensure integrity and then after the patch has been installed a new baseline should be created. This usage will also alert the administrator to what a new patch or application is actually doing. The application will also be executed on a routine basis to ensure sensitive files have not changed.

Methods of Ensuring Integrity

The following attributes of an identified sensitive file will be stored in the data repository and will be checked against when ensuring the integrity of the file:

- **Time Stamp** – There are three different time stamps taken from the file. These are the last modification time, the last access time, and the last inode modification time. The setting most used will be the last modification time, which is time of the change made to a file. This attribute can be easily changed on the UNIX operating system and should not be taken to be absolutely accurate. The other two time stamps are off by default but can be used in the event they are needed.

- **File Size** – The file size is the amount of space that the file uses on the disk. This attribute also can not be completely relied upon because it would be easy to add "junk" to a file to ensure the file sizes match.

- **File Account Owner** – The file account owner is the actual account that has ownership permission to the file. This account would have complete access to the file and should therefore be a trusted user. When a sensitive file is identified to the application it will be owned by this trusted user and will be checked to ensure that it remains that way.

- **File Group Owner** – Similar to the file account owner this attribute is the group that owns the file. Accounts on the UNIX operating system have specific memberships in groups that give them access to specific files that the group owns.

- **Permission Settings** – The permission settings is a mini access control list that allows access to the file on the levels of read, write, and execute to the account owner, the group owner, and everyone on the system. The permission settings are very important to sensitive files to ensure that only authorized personnel can change the file.

- **File Inode** – The inode for the file is a pointer to where the file exists on the physical disk. In the event that a file is removed and replaced with a new one, the location on the disk would change.

- **Message Digest** – A message digest or fingerprint is the result of taking a file or message and using an algorithm to output a hash. The algorithm to be used for this application is MD5, which was developed in 1991 by MIT Laboratory of Computer Science and RSA Data Security, Inc. This algorithm is an industry standard and is considered to be secure.

File Monitoring

The identification of who modified a file is a little more resource intensive than checking a file to ensure it has not changed. To do this a daemon will need to be setup to monitor the file and who accesses it and changes it. Fortunately on the HPUX servers on the network there is already an application in place that can be used to perform this function. Using the eTrust Access Control application will provide a secure system to monitor the file. This application also will be able to tell which account changed the file even after the account acquired root account access.

Due to the possible resource usage the option will not be required of each file and will be setup in the database on a per file basis. Since this application and eTrust Access Control are separate there is a routine check process to ensure that the files identified as being monitored are truly being monitored.

What Changed?

The ability to tell what changed in a file is another important feature incorporated into this application. This feature can be used on sensitive non-binary files identified to the application. The application will store a clean version of the file. In the event that the file was found to be tampered with and the appropriate flag was checked, the application will report on the differences between the clean file and the tampered file. So now, this application will be able to tell who changed the file and what changed in it.

Take note that this option can only be used on non-binary files. For example, a good use for this check would be the /etc/passwd file on the server. In this example you would be able to tell who modified the file and what that person did to it. This would greatly assist in the time the server is down after a break in because an administrator can easily tell what the attacker did.

Remote Connectivity

For this application to work there must be a secure method of transferring the application to the remote server and then executing it with the appropriate settings. If done wrong we could open a potential vulnerability on our servers so we need to take caution. We handle these requirements by using a mix of Secure Shell and eTrust Access Control.

Application Account

The first step we need to make is an application account that we utilize for this process. The name of the account should be something that will not clue an attacker to the fact they are being monitored. We will also add a controlled group for this application as well. We add both the account and group with the same UID and GID on all the servers on the network.

Now that we just opened another vulnerability on the server, we must address it. We will have to disable this account so first we modify the account's .profile configuration file. We remove the old file and add a new one with three lines in it.

```
PATH=/usr/bin:/usr/sbin:/usr/lbin:/sbin:/usr/seos/bin
umask 027
exit
```

In the event that account is able to interactively logon it will be denied access because of the exit command.

Next we will modify the account's password aging settings to never expire. We use the modprpw utility to modify the account's settings. The fields changed will be the password lifetime, password expiration time, and the time between logons.

/usr/lbin/modprpw –m exptm=0,lftm=0,llog=0 <*account name*>

The ability to use ftp to connect to the server must also be disabled. We accomplish this by placing the account name in the /etc/ftpusers file. Although the ftp network service was disabled, we perform this step in the event that the service is enabled at one point.

/usr/bin/echo "<*account name*>" >> /etc/ftpusers

Now we will set the password stored in the TCB file for the account to an invalid encrypted password. This is an encrypted password that the system would never generate. We will use the encrypted password "LOCKED" for this. The

field you will be modifying is the u_pwd field. If the account was created with a password (which all accounts should be) you will remove the encrypted password. If the account was not created with a password the field will have a asterisk in it which means the initial password was not set. Either way we change the password field to "LOCKED". Use caution when editing the TCB files directly.

:u_pwd=LOCKED:

Take note that this application account still requires a valid shell in order to work properly. This is not a big exposure because we mitigated the risk with the steps above.

Secure Shell

The application account created above requires remote connectivity to the servers on the network. This is accomplished by providing an encrypted network tunnel to work through with Secure Shell. However, we must lock this down as well to ensure it is not misused.

Since we do not have a valid password assigned to the account, we must use digital certificates for authentication. We first need to set the application account up with a .ssh directory in its home directory. We accomplish this by surrogating to the account and then running the /opt/ssh/bin/ssh utility to connect to the central security server. This step will prompt us because the utility cannot successfully authenticate that the server is the server. It will show the RSA key fingerprint of the central security server and when answered yes, will place the fingerprint in the account's known_hosts file. Now we need to get a copy of the authorized_keys file. This file is the public key of our central security server and is locked down to just that server able to come in to this server. This file will be located in the home directory for the application account under the .ssh directory. We will change the permission settings to read and write for the owner only. The file will be owned by the application account and the application group.

eTrust Access Control

The eTrust Access Control security application will be used to authorize the root level access and will also be used to secure the files the application will use.

First we need to setup the sudo access within eTrust Access Control. This access is the mechanism that will allow the root level access on the remote server. The following commands are performed within the security application's command line interface.

**nr sudo <*name*> defacc(none) owner(filownr) targuid(root) \
comment('<*home dir*>/*command*_file')**

auth sudo <*name*> uid(<*account name*>) acc(x)

We will also need to authorize the appropriate access for the application account in eTrust Access Control. This is accomplished on all the rules in the file class within the security application so the checks will not be denied access. This step is performed at the shell prompt on the remote server.

**/usr/bin/echo "list file" | /usr/seos/bin/selang -s | /usr/bin/grep -v \
^"(localhost)" | /usr/bin/awk '{print "auth file " $1 " \
uid(<*account name*>) acc(r)"}' | selang**

Since the home directory for the application account is very sensitive, we will use the security application to secure it. The access allowed will be for the application account and the datasec group (UNIX security personnel) to have all access to the directory. The root account will also require read access through the Secure Shell daemon so the proper key files can be read. The following steps are performed within the security application's command line interface.

nf <*home dir*>/* owner(filownr) defacc(none)

**auth file <*home dir*>/* uid(<*application account*>) gid(datasec) \
acc(all)**

auth file <*home dir*>/* uid(root) acc(r) via(pgm(/opt/ssh/sbin/sshd))

The above steps were used to limit the access to the files used to run this application on the remote servers. When these rules are setup within the security application it will secure the files against all accounts, including the root account.

Administration

Now that the configuration has been completed there must be an easy way to administer the application. Making the interface easy to use will reduce training time for the application, reduce mistakes made, and will allow personnel working with it to be more efficient with their time. There will be several methods used for administering the application and all will be explained below.

Manual Modifications

The most complicated method of updating the configuration and baseline files would be to manually edit them with a text editor, such as vi. This method has the highest probability of mistakes being made which would end up in corruption of the baseline file. This step should only be taken by the personnel who know the process best. This method can be accomplished by any account that has UNIX access to the files (i.e. the application account and the root account).

Command Line Menu System

This menu system is the tool used when creating a baseline on a remote server or adding to the current baseline of a remote server. The menu system walks the user through baseline and integrity check operations. The baseline options provided are to create a new baseline, add to the current baseline, remove from the current baseline, list the current baseline, and make configuration changes to the current baseline. The integrity check options provided are to run the check either on a specific file or all files that are currently in the baseline.

When creating a new baseline, the menu system will walk the user through the various types of files to input into the baseline. The options include adding a specific file or directory, all the SUID and SGID files, the HPUX configuration files on the server, all the files that are components of UNIX security processes, and content changing files such as the TCB files and log files. After the selection of a type of file to add to the baseline the menu system will prompt for the selection of the checks to implement on the file. This allows the simple toggle between on and off for each of the checks that can be made. After completed with the selection process, the menu system will transfer the appropriate files to the remote server and run the appropriate commands on that server. After completed, the process and the configuration files will be copied to the central security server and removed from the remote server.

The change of the configuration settings was given because there will be obvious files where a specific check does not make sense. An example of this would be the last modification time and the system log file, /var/adm/syslog/syslog.log. If the setting for last modification time was turned on, the file would always be marked as changed because the syslog daemon is constantly updating the file when system events occur. It would make more sense to monitor the owner and permission settings to ensure that they do not change. By default the menu system turns off the inappropriate settings for the files selected, while the user still has full control over the process.

All of the integrity checking options are also included in the menu system. The user will have the option of checking specific files or a complete check of the

server. The menu system will then display all the files that have been changed and allow the selection to view the reports on specific files or the full report on the check.

This menu system must be executed by the application account because of the Secure Shell connectivity.

Automated Check Process

This process checks all the remote systems based on a specific schedule for each server. The process performs an integrity check on all the files in the current baseline and uses the configuration file to determine which settings are to be checked. After the process is completed with the integrity check operation, the final report is created along with a report showing the current options used (i.e. which settings are on and off for each file). Both of these reports are generated in html format and are placed on the UNIX security group web site.

The scheduling for this process is handled through a scheduling table. The remote server is listed along with the day of the week and the hour of the day to perform the check.

This process is executed by the application account on the central security server from within the cron job scheduling facility.

Web Site Reporting

The web page for the integrity check includes all the files found to be changed and what attributes were modified. Each file has a selection box for retrusting when the analyst has determined that the change was authorized. The analyst will also document everything about the actions taken in a text area at the bottom of the report. This data is then compiled along with the files that require retrusting and sent to a custom made CGI program that handles the input. The CGI program takes all the files marked for retrusting and places them in the repository for another automated process to check and process. The report information is then placed back on the report page, a full report page for the process, a full report page for the user making the change, and the confirmation page for the user. The report is also marked as finished when all the findings have been addressed. The report web page also gives the option for the user to baseline all files again.

The main page that identifies all the server reports marks the reports that have not been finished differently so they stick out. From this main page the archives

can be retrieved along with the current reports. The archives show what changes were made in the past and by whom.

The access to portion of the group web site is handled through the SSL realms and is limited to the UNIX security analysts. The CGI files are also protected through the same means.

Important Notes

When creating a baseline of a file, you need to be sure that the file should be trusted as-is. For this reason, the optimum time to create the baseline is right after the system has been installed. Remember to check the files before putting them in the baseline file to ensure you are not checking to make sure a malicious program stays malicious. This can be accomplished by comparing system utilities against a clean image of the utilities.

You want to especially check all the configuration files on the server before trusting them. Since we do not have an automated check for all the processes, this is our check. Once we are assured that the configuration file is correct, we place it in the baseline. Any deviations are then reported and an analyst will review the file to ensure the changes were authorized and do not affect the security of the system.

Another important consideration is that you need to be sure the reports are being reviewed on a regular basis, preferably weekly. If you find a file that is modified and the last time the report was reviewed was a couple months ago, it will be difficult to investigate the reason behind the change. Remember that a security tool is worthless if it is not used on a routine basis.

Section XII – Other Processes

This section is used to summarize other processes that will be used to help secure the server.

Security Patch Checks

Most of the attacks encountered today is because system administrators have not installed the most recent security patches available from their OS vendor. So this process is one of the most important processes we will be implementing. We will be creating this process on this server but all the HPUX servers on the network will be checked.

For this process we will utilize another tool provided by Hewlett Packard. This tool is named Security Patch Check Tool and is used to analyze a system for the state of the patches to ensure all of them are current. Since this tool is meant to be used on one system, we will incorporate it into our process that will analyze hundreds of servers. This tool must be downloaded from http://software.hp.com and installed on this server.

We start with the dumps we made of the software installed on the HPUX servers. This data was sent to the primary security server for compilation. The first step of our process will be to transfer those files to this server.

After we receive the files we use the products main interaction process, /opt/sec_mgmt/spc/bin/security_patch_check.pl to download the latest security catalog files from HP. This step requires internet access. Opening a hole through your firewall for a half hour from only this server would be adequate. The hole would have to use ftp to connect to ftp.itrc.hp.com.

For each server we take the updated security catalog and the software dump file and get the output. We then modify this output to HTML format so we can place it on the group web site. After that report is complete and the web site has been updated and the old current report is now in the archive, we go on to the next server.

After all the servers have been reviewed, we create the main page. This main page will identify the servers that are missing security patches differently than the servers that are up to date with patches so they will stand out more. The main page will also include the date the server was last up to date on the patches. If this date is more than a month ago there is a problem.

This portion of the group web site will be opened to the system administrators. However, it still should be behind authentication so we need to set their accounts up as well for that SSL Realm.

This method of reporting is very effective because it saves the administrators time and it displays in plain view, on a daily basis, which patches are missing. This report would also be suitable to send up to management and auditors.

Account Additions

With hundreds of servers to support, it is not time efficient to be adding accounts all day long. Instead we create a process from the central security server that utilizes a menu system to add accounts. This process would use the encrypted tunnel provided with the security application, eTrust Access Control to remotely manage user accounts. With the help of in-house developed exit scripts on the remote servers it is possible to add accounts and configure the accounts properly without even logging in to the servers. Automating the configuration process also eliminates mistakes made and possibly misconfigurations that give more access than needed.

Further information on the exit scripts used in the security application can be found in Section 7 of this document.

Account Deletions

Similar to the account additions are the deletions of the server. To remove an account correctly is very time consuming so we automate it. This process also utilizes the remote connectivity of the security application and the exit scripts to execute the appropriate commands on the remote servers. Again, not only does this save time on the UNIX security analysts part, but it also eliminates mistakes made.

Inactivity Checker

In Section 9 of this document, we discussed the importance of checking current sessions on the system to see if they are inactive. If they are inactive we will terminate the session. Instead of rehashing why we perform these checks, refer to Section 9 for more details.

The process we create looks at all the terminal sessions that are currently active on the server. It then looks at the inactive time associated with the terminal. This time setting is shell inactivity so this account could be executing something.

Since we do not want to terminate a process running, we check the account's terminal and determine if the account is executing something. If the process finds that the account is inactive at a shell prompt and the fifteen minute time period is past then the process writes a warning to the account's terminal. After reviewing all the open sessions, the process sleeps for a few minutes and then starts over with the check. If that same account that was already warned is found to be inactive still, the process terminates the session.

Quicker Way to Find Files

Since a lot of security processing requires the execution of the find command on the entire file system you may notice that on some larger servers the task takes forever or never completes. A quicker way to execute the find command is to break the find up by mount points.

For example, lets say you need to find all the .rhosts files on the file system. You would execute the following command.

```
/usr/bin/find –name .rhosts –exec /usr/bin/ls –ald {} \;
```

Now lets take that same find and break it up by mount point. We use the xdev option in the find command so we will not cross over to another mount point.

```
for mount_point in `grep -e " vxfs " -e " hfs " /etc/fstab | /usr/bin/awk '{print $2}'`
do

 /usr/bin/find $mount_point –xdev –name .rhosts –exec /usr/bin/ls –ald {} \;

done
```

Although this is not a security check, it was included in this document because it will be useful when developing processes to implement all the checks identified in this document.

Section XIII – Checking Configuration

This section is used to identify some checks we can make to ensure that the security settings are in effect.

Root Account Access

We determined that the root account was not to have remote access to the server. To test this we will utilize a the telnet, ftp, and ssh services to connect to the system.

At another terminal use all three network services to attempt to logon to the server with the root account. The telnet and ftp sessions should not even connect and the ssh logon should deny the access.

File Integrity Checking

We developed an in-house process to track all the sensitive files on the server. To test this we need to change one of the files and then perform the integrity check.

First perform the integrity check of the server to ensure no files have been modified. After that report is clean, pick a sensitive file that is being monitored, such as /etc/hosts. Modify this file by adding another hostname to IP address combination or a comment. Now go back and perform the integrity check on the system again. You will see that the /etc/hosts file was changed and if the text monitoring was on you will see the changes you just made to the file.

Password History

We set the password history depth to have the setting of six. To test this we will need to logon to the server and attempt to change our password.

First logon to the server. When the command prompt comes up, execute /usr/bin/passwd. First change your password to a new one. Now execute the passwd command again and try to change the password to the initial password you logged in with. If you receive the error, “You may not re-use a previously used password.”, the password history is set at least to one. Change the password to a new one and then continue the process until you can change the password back to the initial password.

Disabling of the Telnet Service

We made the decision that we were not going to have the telnet service running on our server. To test this we attempt to connect to the server using telnet and we also look at the ports that are currently listening.

First we attempt to logon from another system using telnet. The connection should be refused. Second we will analyze the data from the netstat utility. Execute the following command.

/usr/bin/netstat –an | grep –e LISTEN –e ESTABLISHED | grep "\.23"

This command will check all the services that are listening or established and then will check if any of them are using port 23 (used by telnet).

Files Without Ownership

Our automated security review checks the system on a routine basis to ensure that there are no files that do not have an account or group owner. Since we have an automated deletion process that is supposed to change the ownership of the files after the removal of an account, this is important to check. To test this we check the file system.

After logging on to the server we will search the file system on a per-mount point basis. The check will be for no user and no group. We should not see any output.

```
for mount_point in `grep -e " vxfs " -e " hfs " /etc/fstab | /usr/bin/awk '{print $2}'`
do

 /usr/bin/find $mount_point –xdev –nouser -o –nogroup

done
```

Attachment 1 – References

The following sources were used in the development of this document. In the event that more detail is needed than what was provided in this document, refer to the following sources.

Allen, Julia, Klaus-Peter Kossakowski, Gary Ford, Suresh Konda, and Derek Simmel. April 2000. *Securing Network Servers*. CMU/SEI-SIM-010. Software Engineering Institute, Carnegie Mellon University.

Australian Computer Emergency Response Team. October 8, 2001. *UNIX Computer Security Checklist*. Version 2.0. Prentice Centre, The University of Queensland.

Computer Emergency Response Team Coordination Center. November 5, 1993. *Generic Security Information*. Software Engineering Institute, Carnegie Mellon University.

Computer Emergency Response Team Coordination Center. February 12, 1999. *UNIX Configuration Guidelines*. Software Engineering Institute, Carnegie Mellon University.

Curry, David A. April 1990. *Improving the Security of Your UNIX System*. Menlo Park, CA: SRI International.

Garfinkel, Simson and Gene Spafford. 1996. *Practical UNIX and Internet Security*, 2nd ed. Sebastopol, CA: O'Reilly & Associates, Inc.

Hewlett Packard. 2000. *HP-UX Version 11.11 Reference*. Palo Alto, CA: Hewlett Packard.

Steves, Kevin. March 27, 2001. *Building a Bastion Host Using HP-UX 11*. Version 1.23. Hewlett Packard.

System Administration, Networking, and Security Institute. January 18, 2001. *How to Eliminate the Ten Most Critical Internet Security Threats: The Expert's Consensus*. Version 1.32. System Administration, Networking, and Security Institute.