Global Information Assurance Certification Paper

Building a Cost Effective Enterprise-Wide Monitoring Solution Using Big Brother

GIAC Certifiied Unix Security Administrator (GCUX)
Practical Assignment 2.0
Option 1 - Securing Unix Step by Step
CDI East December 2003 – Washington DC
James B. Horwath
Submitted April 28, 2004

Contents

4.5.g Big Brother Bliss

Abstract

The modern business environment is cost and customer driven leveraging technology for mission critical decisions and data. When mission critical data or services are unavailable, business operations suffer, revenues are lost, and the business could suffer a PR nightmare. Foobar Industries is in the insurance business dealing with sensitive personal information tightly regulated by government agencies. Due to the sensitive nature of the information with which Foobar deals, it is time sensitive and confidential. Recent regulations place a huge financial burden on Foobar if sensitive data such as Social Security numbers were compromised. Disruptions in the business flow tarnish the relationship between technical staff and customers and could result in long hours, which may have otherwise been avoided with proper monitoring. In the past there was a failed rollout of a large commercial monitoring product that consumed large amounts of capital and resulted in zero business benefit. Management is now reluctant to spend on a project that doesn't improve the bottom line. This has resulted in problems going undetected and spiraling into revenue sapping outages. There are no budgetary dollars for a new monitoring initiative; however the technical staff recognizes the need for effective resource monitoring. Management and staff have created a secret project; code name Gazinta. This project must be implemented with a budget of 2500 U.S. dollars, and result in an Enterprise Monitoring Solution. Management has provided IT with an opportunity to excel. The paper will detail the steps required for the deployment of a cost effective Web based monitoring system utilizing the software Big Brother and Apache.

Section I: System Description

1.1 Server Role

This server will be responsible for the monitoring of Foobar's computing infrastructure, alerting staff to issues requiring intervention or escalation. Currently staff relies on homegrown scripts or user complainants for monitoring purposes. The current method of enterprise monitoring is not reliable and is a poor business model. The ad-hoc model, or worse the customer complaint model should be changed for the following reasons:

- Monitoring, alerting and notification will be modeled using the company's processes and procedures.
- Centralized monitoring will eliminate monitoring redundancy; this allows precious computing cycles to be used for business functions and not for redundant monitoring.
- Inter-departmental communication and cooperation improves by staff seeing the whole picture of how each piece integrates and affects the business structure.
- Monitoring functions execute with minimal privileges to get the job done.
- Scripts and or processes are reviewed by Security Staff to check company policy compliance, and potential security flaws. The interaction between Security

personnel, management and technical staff will cultivate an educational environment regarding security and increased business efficiency. Security methods and best practices will be viewed as a friend and not an impediment to job performance.

- Network traffic types and flows will have a baseline, allowing security and network staff to spot any abnormalities, determining what needs investigation and what is normal business activity.
- Development, application and administration staff will have an increased awareness of leveraged computing resources, normal resource states, and an increased appreciation of the relationship of business processes and computing services.
- The monitoring system will run with a minimal number of daemons and services to meet the needs of the business. As an added benefit, a secure system runs more efficiently and is easier to maintain.

The Big Brother monitoring server will function as an appliance box having two functions: gather monitoring data and display Web pages for monitored data. Although Big Brother is capable of monitoring almost anything, Foobar's server will monitor only the Unix infrastructure. Security policy dictates all servers operate with a very narrow business scope; general purpose computing machines are not encouraged by the Security Office.

1.2 Hardware Requirements

One of the strengths of Linux is the ability to run well on legacy hardware where other vendors would cringe at the thought. Secure Linux implementations do not suffer from code bloat; the systems are lean, efficient, and fast with a small footprint. The Unix environment we will be monitoring consists of IBM AIX and Sun Solaris leased machines. Our hardware must run the selected monitoring software Big Brother and any adjunct applications; meeting the current business initiatives. Our decision is based on the most inexpensive model which meets our business needs of enterprise monitoring. The company Foobar is currently working on a server consolidation initiative, management and staff are actively removing leased assets from the enterprise. All current AIX and Sun servers are leased assets with vendor costs associated with them. Therefore using one of the Legacy AIX or Solaris servers is out of the question; vendor support costs will increase, license costs do not decrease, and a leased asset remains in the enterprise. The buyout cost for a leased asset is cost prohibitive for the rate of return. The leased servers are not current generation technology. Although Linux will function well on the beaten windows server lying in the corner; there is no business justification for trusting a mission critical application to an old beaten down piece of hardware used for nothing more than Internet Explorer or Solitaire. Luckily, hardware is cheap and the astute buyer can find powerful bargains. During a recent Dell sale three machines were procured for under $1500 U.S. dollars. Although one machine would have been acceptable, three machines are preferable. One machine will be used in the production environment, one in the test environment, and finally a machine in the Disaster Recovery Center. Below is a summary of the machines we bought.

Dell Dimension 2440 Pentium 4 at 2.40 GHZ 256 MB of RAM 40 GB Hard Drive 48X Max CD-RW Drive 48X Max variable CD-ROM Drive No floppy drive 17 inch monitor Keyboard Mouse 1 Year Warranty

Although the system is inexpensive, it will meet our business needs perfectly. This may appear to contradict the consolidation initiative, but it actually fits in very well. We have added a low cost machine which no contact or vendor costs. In return we are adding a needed business function of system monitoring which was previously missing.

1.3 Operating System Type and Version

The Foobar infrastructure to be monitored is a mix of leased Solaris and AIX machines. Routers, switches and wintel machines will be monitored with a different tool at this point. The company is actively involved with a server consolidation initiative, although at first glance using an existing machine may seem like a valid option, the machines are leased assets and management is trying to decrease leased assets and support costs where ever possible. The BTF (Better than Free) version of Big Brother software runs on most any Unix variant. Although there are many free or low cost Unix distributions, the Engineering Team still needs to approve the Operating System chosen. Free BSD is very secure out of the box, but there is little name recognition to non-technical or non-Unix people, so this will be difficult sell to management. The Red Hat brand has name recognition and is a highly popular, making it an easier sell to management as our choice. Our cause will be championed by the fact large, reputable companies such as IBM have thrown their support behind Red Hat. Red Hat 9.0 can be downloaded for free from the Red Hat site, making the end cost for the Operating System nearly zero dollars. Dealing with strict security regulations and a small budget, Red Hat can be transformed into a secure low-cost monitoring solution without much effort. One concern is Red Hat being at End-of-Life as of April 30, 2004. At that point Enterprise Edition will be evaluated to see if it fits into our future business plan. Some Big Brother installations have experienced memory leaks using Red Hat 9.0 requiring reboots every few weeks, this is a risk we need to take.

1.4 Third Party Software Required

a) Big Brother

The Big Brother application was selected to monitor Foobar's infrastructure. The package was selected due to its flexibility, ability to customize, low cost and security

features. We will be implementing version 1.9c of Big Brother. There is a newer release of the application; however this happened while this paper was being drafted. Later this year Foobar will be upgrading its version of Big Brother to the latest release. Big Brother has two products available: the BTF (Better than Free) version; and the professional version offered through Quest Software. Having to work within the confines of a very tight budget, the BTF version is the natural choice. As per the licensing agreement shipped with bb19c, a one-server license must be purchased if Big Brother (BB) is being used to monitor revenue-generating systems. Foobar does not want to violate any license agreements; therefore we have decided to buy a site license. The cost of $695.00 does not sap our budget and will allow us to remain within the license agreement regardless of where Big Brother is deployed within the enterprise. There is also a very strong rumor the license fees will increase with future releases. Luckily, legacy licenses will be grandfathered, current licensed users won't face increased license fees since Big Brother is a one-time license. Throughout this document Big Brother will be referred to as BB. The software can be downloaded from http://bb4.com/download.html.

b) Apache

Big Brother is a Web based system and network monitoring tool which works with most popular http servers. Our goal is the deployment of a very secure Web based monitoring system. To achieve this goal, securing the Web services is an integral part of the system design work. Apache has a good security reputation; only a few exploits have been reported against Apache since January of 1997. Obviously another consideration is the cost of ownership, Apache is free. Although Apache has a strong record against exploits, caution will be exercised during the configuration of Apache. Even the most secure application is susceptible to exploitation with a less than secure configuration. The latest stable release of Apache is 2.0.48, available for download at: http://httpd.apache.org/download.cgi.

c) mod_security

Mod_security is an open source intrusion detection and prevention module for Apache protecting Web based applications from attacks. Mod_security is written and maintained by Ivan Ristic, the code was modeled after the behavior of SNORT. I was introduced to mod_security at the SANS CDI conference in December of 2003. I was amazed at the security, power, flexibility and ease of use. Apache teamed with mod_security can create a formable defense for any Web server. The code can be downloaded at: http://www.modsecurity.org/download/index.html. I had originally tried to use version 1.7.4 which was the latest stable release at the time. However, I found a bug in the code trying to chroot() the Apache process. Working with Ivan I ended up using version 1.56 2003/12/17 23:25:08. I had received this version directly from Ivan to correct the chroot() issue I was having. Although this version says 1.56, I believe it is from the 1.7X source tree.

A set of predefined mod_security rules can be downloaded from:
http://www.modsecurity.org/documentation/snortmodsec-rules.txt.

Although the Web server is shielded from the Internet by firewalls, we will be using mod_security to add an extra layer of defense to our server.

d) TSM
Tivoli **S**torage **M**anager (TSM) is the backup solution implemented by Foobar. We will be using the stock version of TSM without adding any bells and whistles. Although it is possible to encrypt the backup data stream via stunnel, in a large production environment this is really not feasible due to the overhead of encrypting and decrypting the data across the tunnel. We will be installing the code from the Foobar tool server, the original version was supplied by IBM on the distribution cd-rom.

e) Syslog-ng
The default logging facility shipped with Red Hat 9.0 is the old standard syslog, although it works, syslog will not fulfill our needs from a security point of view. Balazs Scheidler is the developer behind syslog-ng, he developed and still maintains the syslog-ng source tree. Standard syslog transports data to the remote host via UDP, there is the possibility of losing data during transport since UDP does not guarantee delivery. Syslog-ng was designed to use either UDP or TCP for data transport, using TCP guarantees the data will arrive in tact. Additionally, syslog-ng supports advanced filtering and forwarding without any additional development. To bolster an already strong case, syslog-ng easily integrates with stunnel for an encrypted, reliable, data transfer. All system log files will be stored locally for staff convenience, as well as, on a centralized, secure logging server for longer-term storage. Syslog-ng is freeware downloaded from http://www.balabit.com/downloads/syslog-ng/1.5/src/.

f) Tripwire
Tripwire is a Unix integrity checker written by Gene Kim and Dr. Eugene Spafford at Purdue University. Tripwire makes a baseline of your system and then checks the system against the original baseline for discrepancies. Tripwire has two versions available, the freeware version and the commercial version. The free version is available only for noncommercial Unix systems, luckily Red Hat 9.0 falls into that category. Foobar has made a commitment to implmenent the commercial version of tripwire throughout the enterprise. Until there is an enterprise deployment of Tripwire, we will be using the free version. Tripware can be downloaded at: http://www.tripwire.org/downloads/index.php

g) CIS Linux Benchmark
The Center for Internet Security has a tool available to verify and check the security rating of a Linux system. This tool will be used to verify the security rating of the system. There is no additional software to download in order to get CIS to run since the application is written in Perl. The application is easy to use, even for automated system verification. The code can be downloaded at: http://www.cisecurity.org/sub_form.htm.

1.5 Business Critical Non-System Processes
This server will be responsible for monitoring the Foobar Unix server infrastructure. As a result, the server will have a narrow business scope, and will be modeled after an appliance server. There should be very few non-system processes running on this machine. Besides core operating system processes only Big Brother and Apache

should be running. Below is a listing of the essential core operating system daemons. Although stunnel and syslog-ng are third party applications, they are included in the core listing since they are actually replacements of the standard operating system utilities and considered core/essential.

Process Table

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	29-Feb	?	0:00:04	init
root	715	1	0	29-Feb	?	0:00:00	/usr/sbin/sshd
ntp	731	1	0	29-Feb	?	0:00:00	ntpd -U ntp -g
root	756	1	0	29-Feb	?	0:00:00	crond
root	891	1	0	29-Feb	?	0:00:00	login -- root
root	892	1	0	29-Feb	tty2	0:00:00	/sbin/mingetty tty2
root	893	1	0	29-Feb	tty3	0:00:00	/sbin/mingetty tty3
root	894	1	0	29-Feb	tty4	0:00:00	/sbin/mingetty tty4
root	24593	1	0	2-Mar	?	0:00:00	/usr/sbin/stunnel
root	24596	1	0	2-Mar	?	0:00:03	/usr/local/sbin/syslog-ng
root	24601	1	0	2-Mar	?	0:00:00	klogd

1.6 Network Services Available

There will be a small number of active network ports on the machine. Administrators will have access to the server via ssh (port 22) for system administration. Big Brother Web pages will be available on port 80 through http services using Apache. The Big Brother server listens on port 1984 for Big Brother traffic. Port 123 will be activated to leverage NTP for system time synchronization. System logging will be done by syslog-ng running on port 514. DNS traffic will be allowed outbound to facilitate domain-name lookups, although the bind/named applications are not permitted to run locally.

Service	port	direction	business purpose
ssh	22:tcp	inbound/outbound	command line access, bb data fetch
http	80:tcp	outbound	serve bb Web pages
ntp	123:udp	outbound	sync system clock with ntp servers
syslog-ng	514:tcp	outbound	transfer data to log server
bbd	1984:tcp	inbound	receive bb monitored data

1.7 System Access – Who/why/how

The system will be engineered after a security conscious model; command line access will be limited to one access server and the system console. Direct root access is limited to the system console. Network services will not be controlled by xinted; which means tcpwrappers is not an option for our applications. Ingress and egress access to our system will be controlled via iptables, a host based firewall application. Administration staff will not have access to the root password. Only two staff members have knowledge of the root password, the Unix system group manager and the Security Officer. The root password is stored in a safe in the Security Office. Privileged access is granted via sudo; a security side effect of sudo is the logging of all privileged commands. System access will be limited to the Unix administration staff, command line access will not be granted to any other Foobar staff member. Excess system and

group accounts will be removed during the hardening process; an example would be the nobody account.

1.8 Network services access – Who/why/how

There are three network services offered by this machine: Big Brother, Apache and ssh. Big Brother clients connect to the Big Brother server process bbd and transfer monitored data over socket port 1984. There is an exception to the rule; DMZ machines will have their data transferred via ssh. The Big Brother server will connect to the DMZ machine and transfer the monitored data. Big Brother has a tcpwrappers like facility built into the application; only machines defined and monitored by Big Brother will be permitted to contact the Big Brother daemon. Big Brother pages will be served via a password protected Web server, Apache accounts are granted by department manager requests to the Unix system manager. The Apache Web server will be protected with MD5 digest passwords, only essential personnel will have access to the monitored data. Initially the number of users interested in Big Brother data should be small; this policy may have to be modified in the future when the popularity of Big Brother increases. All network ports will be protected by a host based firewall to help lessen or prevent network attacks to the system. Finally, as discussed earlier ssh is available on the server. System logins will be available only to the Unix administration staff, no other staff will be granted a login to the server. There is a single point of access to the ssh daemon, and that is through a Unix access server. Since ssh is not using tcpwrappers, iptables will control access to the ssh daemon. Having one access server is a single point of failure, however it is an acceptable risk because of the control it gives our environment.

1.9 Risk Mitigation and Analysis

A large security risk is staff walking away from a working area with Big Brother actively being displayed. The nature of system monitoring requires the monitoring of processes, disk space, CPU and messages files. The Big Brother Web pages would be a gold mine for attacker looking to infiltrate and compromise the infrastructure. Unfortunately, social engineering is a very effective method for gaining access to infrastructure information. The production and disaster recovery servers are located in a locked data center requiring badge access for entry. The computer room has a raised floor, but the cables strewn and bundled under the floor would pose a challenge even to the Delta Forces. In fact, there is a rumor Jimmy Hoffa may be buried under the tiles in the data center. There is no threat from anybody crawling under the floor to attack a system. Due to the chaos of the floor underworld, there is an exposure if an attacker either cut or pulled a handful of network cables. This is issue is actively being worked by network services.

1.10 Risks, Server Roles, Compromise Sources, Attack Prevention

The Big Brother server is well protected from the Internet with firewalls and routers; the main threat is the Foobar Intranet. Although background checks are required for employees, vendors and visitors are not subject to the rigors of a background check. Our most likely source of attack would be an internal network attack designed to knock out the monitoring system. The attacker could be a disgruntled employee, a vendor who feels slighted, or an intern trying some new hacking tricks found on the Internet. For

an attacker to be successful in this environment, first the monitoring server must be quietly disabled before the primary target is attacked. The sensitive nature of insurance data increases the attractiveness of the data to an attacker. Social security numbers, names, addresses, and birthdays, are all lucrative products on the black market. Our mission will be to prevent unauthorized access to network services. The largest exposure on the server is the http server; a large amount of resources will be used to secure network services. Data integrity of the system is a primary objective, Web pages need to be protected from alteration or hijacking. The system needs to be protected from people loading unauthorized material such as pornography onto the monitoring system**.** The security of the system will be anchored by a host based firewall application, iptables. The purpose of this layer of defense is to make the system an unattractive hacking target. Configuration controls will be used to prevent direct root access to the system. Privileged access will be granted to a controlled group with sudo, additionally all commands will be logged with the system logging facility. Apache defense will be bolstered with the addition of mod_security, a third party IDS module for Apache. Monitoring data from DMZ machines will be gathered with ssh, limiting the number of holes in the firewalls, and encrypting the data during transport. All unnecessary network services will be disabled, and their login accounts and groups will be removed to prevent exploits from unused network services. The core application, Big Brother, will be monitoring the system locally, as well as all Unix servers for any anomalies or resource exhaustion attacks. If Big Brother detects any conditions out of the normal, staff is alerted to investigate the issue.

1.11 Server Criticality, Data Integrity and Hardening Effort

At this point the server is not classified as mission critical by management and staff of the company because the project is in it's infancy. We have the anticipation that the Big Brother paradigm of enterprise monitoring will be embraced quickly; at that point the server becomes classified as mission critical. At this point management may not consider the server critical; however administration staff does since it monitors the health of the Unix server infrastructure. Data integrity is very important for several reasons: during problem analysis the data will play an important role in the Root Cause Analysis. During problem post mortem analysis the data will play an important role for lessons learned. This will help staff learn how to prevent an incident of similar ilk from repeating itself. Big Brother data such as disk consumption will be used for trending and analysis. The data gathered by Big Brother will aid capacity planning and budgeting metrics. When staff is notified of a problem the information must be correct, faulty notification will reduce the confidence staff has in Big Brother to detect a problem. Although the Big Brother server is not directly involved in revenue generation, its primary business function is to prevent small problems from escalating into large problems. Since the insurance industry has tight deadlines for many financial processes; system availability is critical. Loss of service at the wrong time of a business cycle may result in loss of revenues and public relations damage. The data must be protected. If an intruder compromises sensitive data, a huge financial liability is placed on Foobar. The system has a narrow business scope and functions as an appliance server. Due to the server criticality staff will spend extra effort to harden the system.

1.12 Network Access, Firewall Protection and Reasonable Risks

This server has no direct access to the Internet. Externally the server is protected by firewalls from the outside world; internally a host based firewall application residing on the server protects the server. There will be no network services such as http available to the outside world. Network services are available to a tightly controlled group of internal staff. The closest the server will come to the Internet is ssh contact to a DMZ server. The data transmission will be initiated from the Big Brother server to the DMZ via ssh; iptables will be used on the server to prevent unauthorized access to the server. The biggest risk we will have will be the user community. Users have a bad habit of leaving terminals unlocked when they vacate their working areas. Selected support staff will have Big Brother running on their workstations. An attacker could walk by and start looking at the Big Brother pages gathering critical infrastructure information. One of the greatest threats is social engineering from a friendly person walking by somebody's desk. The “friendly visitor” may initiate a conversation by saying something like, “hey that's really cool, what it is.” Our customer service focused staff will naturally feel obligated to give a quick demonstration showing the power and flexibility of Big Brother. Our kind staff member has shown a potential attacker the road map to our infrastructure. We will address this user exposure with ongoing security awareness education. We can’t remove users since we need them to administer the machines. The practice of installing development tools such as compilers and kernel source is not an accepted best practice for a secure server. However, our server was purchased with a non-standard NIC not supported by the Linux kernel. Every upgrade requires the NIC be compiled into the kernel. The time required to install, compile, and finally remove tools, is a major resource drain. The number of Linux servers in our enterprise is small; we will accept the exposure of leaving development tools on our monitoring server. There are also reports on the Big Brother mailing list of standard Red Hat 9.0 kernels suffering from a severe memory leak. Not everyone has experienced this phenomenon, users who compile their own kernel, or pieces of the kernel do not seem to suffer this affliction. Another risk is Big Brother by default polls a system every 300 seconds/five minutes. Between polls the system is vulnerable if an exploit occurs and can be covered up during the five minutes interval. There is no easy way around this risk, we will note the risk and accept it.

1.13 Toplology Maps

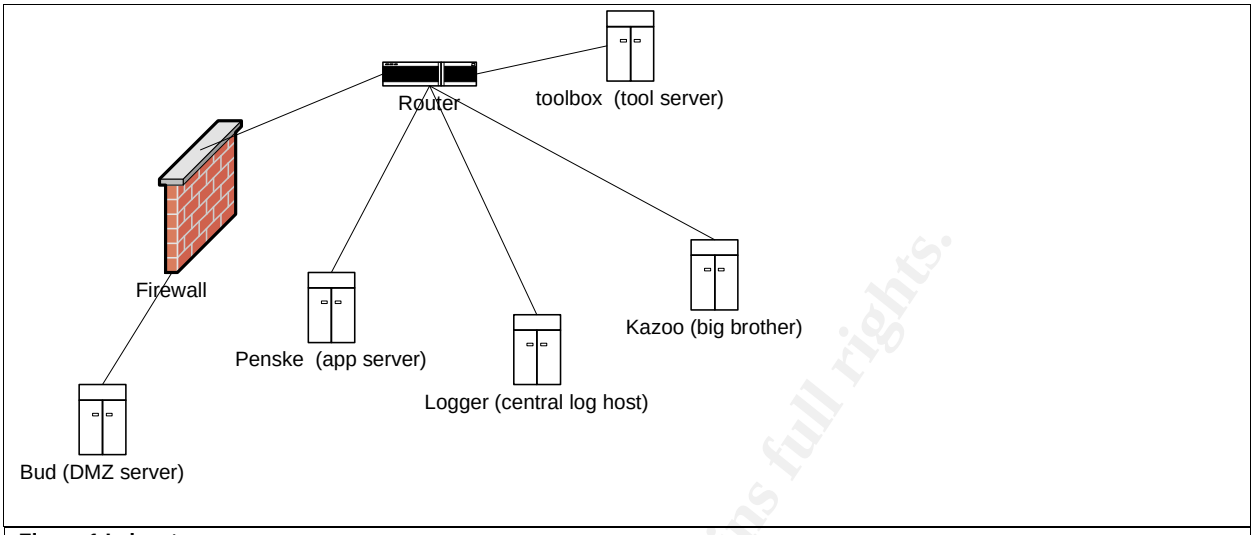


Figure 1 Lab setup

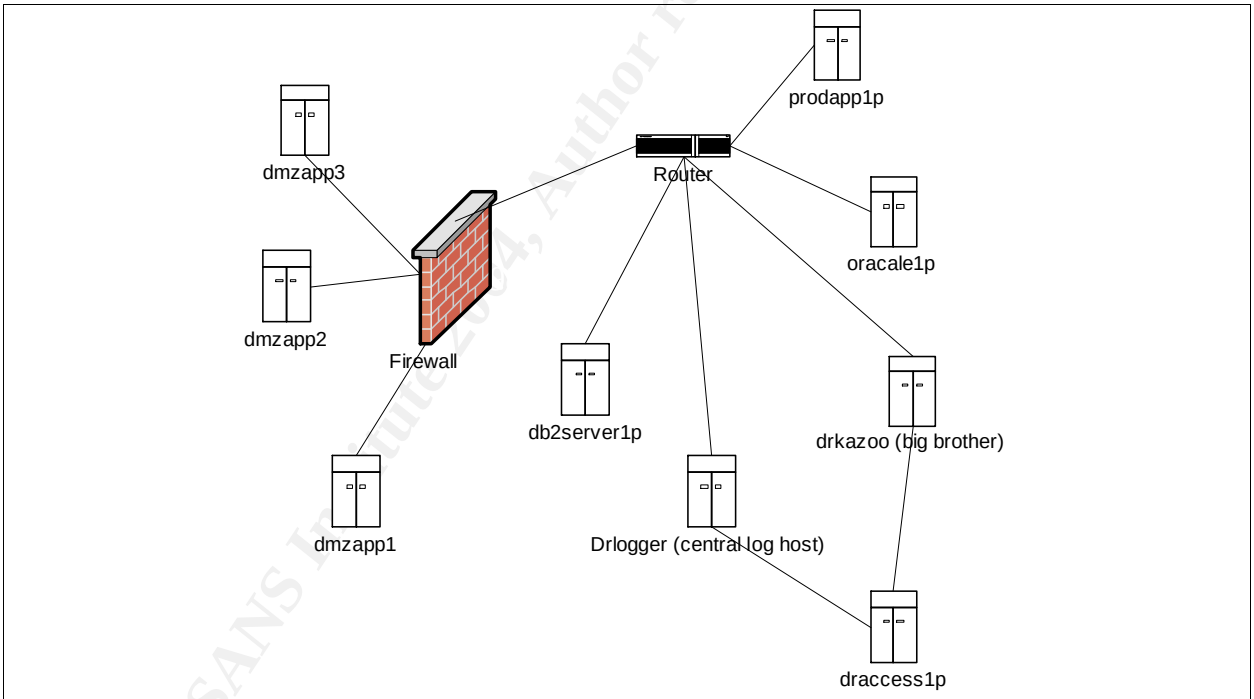


Figure 2 Disaster Recovery Center

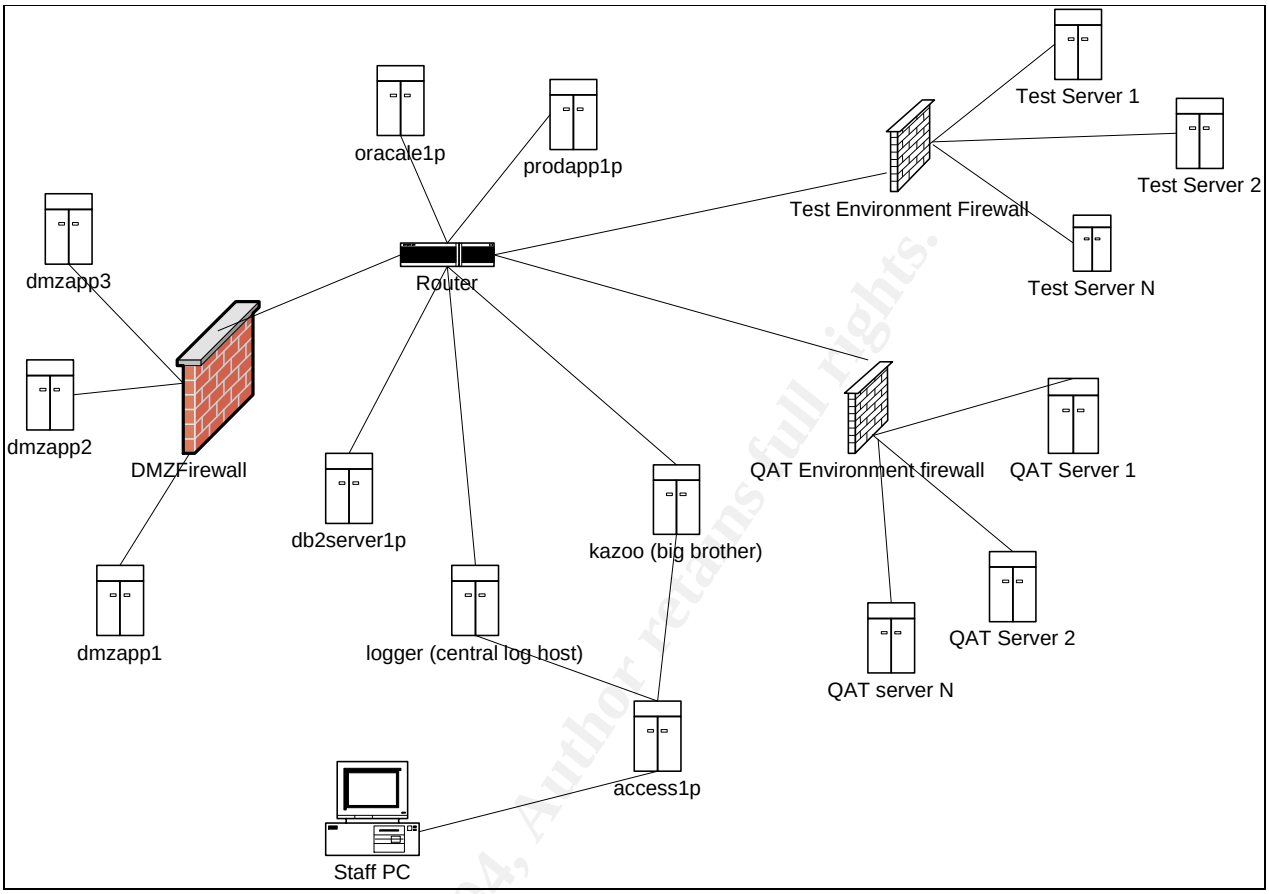


Figure 2 Production setup

SECTION II: Steps to Install and Harden the Server

The operating system we choose for this project was Red Hat 9.0. The system was procured for a very low price, and the source code is freely available. This is both a blessing and curse. Since the source code is freely available, exploits are easier to write, however exploits are patched more quickly since the source code is available.

2.1 Basic Hardening Principles

- Install only the software required by the business, install the minimum amount of software to keep meet the business needs
- Keep all software up-to-date with security and bug patches
- Delete any user and group accounts not needed on the system
- When user and group accounts are needed for system or daemon functions, grant login shell's only when absolutely necessary
- Run only a minimum number of services needed to meet customer and business needs
- If possible chroot any publicly accessible daemon

- Be careful with SUID files lying around the system; keep these files only if necessary
- Administration staff will use sudo to obtain root privileges
- Configure logging locally and to a central logging server
- Configure each host to have its own host based firewall
- Comment configuration files making it clear why something is being done.
- Follow the security guidelines published by the Center for Internet Security.

2.2 Operating System Installation

The operating system was purchased at a major electronics chain during an end-of-year holiday sales. The operating system cost less than $30 U.S. dollars after rebates and coupons. I was not able to get screen prints for everything I did, I will describe each screen as well as the how's and why's of each selection. If I was lucky enough to include screen prints, I feared this document would become unmanageable in length. During this part of the document we will need to use our imaginations.

Power on the machine and insert the first cd-rom.

First screen
We are asked if we want to install or upgrade Red Hat Linux.
Response: Press return and advance to the next screen

Second Screen
Name: Choose CD media test
Response: Skip

Third Screen
Name: Welcome screen, documentation location, and brief explanations
Response: Next

Fourth Screen
Name: Language Selection
Response: U.S. English

Fifth Screen
Name: Mouse Configuration
Response: 3 Button Mouse (PS/2)

Sixth Screen
Name: InstallationType
Response: Custom
Discussion:

We are going to choose the Custom option because it allows us greater control of the packages installed on the system. Our system has a non-standard NIC that requires compilation of the proper driver into the kernel. We will need to choose packages to meet security and business needs.

Seventh Screen
Name: Disk Partitioning Setup
Response: Manually partition with Disk Druid
Discussion:
This single purpose-monitoring machine will not require huge amounts of disk space. That being said, we will still allocate enough disk space to accommodate present and future business needs.

Eigth Screen
Disk Setup
Discussion:
The file system requirements do not require a large amount of disk space even though our server was shipped with a large hard drive. Although our server will not require large amounts of disk space, enough disk space will be allocated to each file system avoiding present and future space issues. Here is a snap shot of how the disk was partitioned on the server.

Partition	Size
/	1 GB
/boot	100MB
/home	100MB
/var	1 GB
/usr	5 GB
/tmp	5 GB
/opt	1 GB
/usr/local	5 GB
Swap	5 GB

/boot – The Red Hat recommendation of 100 MB was followed. The choice of keeping /boot as its own file system allows more control over the mount options. The file system will be mounted ro.
/ - There is plenty of space so root was created as 1 GB, this will be large to avoid any space issues. A larger size was chosen to avoid any disk exhaustion attacks on the root file system.
/usr – The user file system was made large to accommodate any future growth, trying to avoid any file system space issues. We need to do everything we can to help this project succeed; running out of disk space will hinder our project.
/home – The home directory partition was created on the small side, there is no business reason staff should be storing files on this server. The size of 100 MB is plenty of space for home directories.

/var – The var file system was created large enough to handle any log files that may accumulate on the system. If we have a DOS attack, or we are scanned by Nessus there will be a log of data created from the attack. A larger size was chosen to avoid loss of logging service from an attack that generated a lot of log traffic.
/tmp – This file system was created on the large size in case we need to perform any administration work in the future. We have a large transitory file system to use.
/opt – Nothing special here, the file system was created 1 GB to avoid file system problems. This file system will be lightly used.
/usr/local – This file system will be the home of Apache and BB; the file systems were sized on the larger side to accommodate the applications.
Swap – Typically swap is 2xmemory, however since disk space is plentiful on this machine, swap was made 5 GB.

Ninth Screen
Name: Boot Loader Configuration
Response: Click on the "Use a boot loader password" box. A dialog box will appear prompting you for a password, and verification of the password.
Discussion: GRUB passwords prevent people from getting a root-shell at the boot prompt by typing "kernel-2.4.20-24.9 init=/bin/sh" at the prompt. The person would have root access to the system without us knowing about it. This prevents unauthorized access to the system at the boot prompt.

Tenth Screen
Name Firewall Configuration
Response: Select the security level High, allowing coming SSH.
Discussion:
A host based firewall is powerful, often overlooked application. Security conscious administrators should enable host-based firewalls on production servers. Later, during the system hardening we will replace the stock Red Hat firewall configuration with a more secure configuration. The server will be protected from exposure during configuration by the host-based firewall.

Eleventh Screen
Name Additional Language Support
Response: English was selected based on the target audience.

Twelfth Screen
Name: Time Zone Selection
Response: The server is in America/New_York time zone.

Thirteenth Screen
Name: Set Root Password
Response: Set root password in the dialog boxes, the root password must be confirmed.
Discussion:
The root password should be at least 8 characters; consisting of a mixture of alphanumeric characters, avoiding words, phrases and easy to guess strings. The root

password grants privilege to the system, the root password should be handled with extreme care. This will be discussed later.

Fourteenth Screen
Name: Authentication Configuration
Response: Check enable MD5 passwords and enable shadow passwords. The other options: NIS, LDAP, Kerberos 5 and SMB are not selected.
Discussion:
MD5 passwords can have a length of 256 characters making them a more secure choice, and shadow passwords store password in a secure manner.

Fifteenth Screen
Name: Package Group Selection
Response: We will select a minimum number of packages to meet our business needs. Choices are discussed by section below.

Desktops Section
X Window System
Do not select anything

Gnome Desktop Environment
Do not select anything

KDE Desktop Environment
Do not select anything

Discussion:
Our system will be engineered with security in mind from the beginning. We will build a skinny system and fatten the system as needed to meet business deliverables.

Applications Section
Editors
Under editors vim-enhanced was selected. Most of the Unix administrators do not know how to use the basic Unix editor ed, so a graphical editor is needed on the system. The editor Emacs is out, vi/vim is the editor of choice.

Engineering and Scientific
Do not select anything

Graphical Internet
Do not select anything

Text-based Internet
Do not select anything

Office/Productivity

Do not select anything

Sound and Video
Do not select anything

Authoring and Publishing
Do not select anything

Graphics
Do not select anything

Games and Entertainment
You're kidding right? Isn't problem solving and delivering solutions in a pressure packed environment enough fun? ☺

Discussion:
The only thing interesting in this section is the choice of editor. The choice of editor will not increase or decrease the security of the system, or help meet customer needs from a monitoring perspective.

Servers Section
Server ConfigurationTools
Do not select anything

Web Server
Do not select anything

Mail Server
Do not select anything

Windows File Server
Do not select anything

DNS Name Server
Do not select anything

FTP Server
Do not select anything

SQL Database Server
Do not select anything

News Server
Do not select anything

Network Server

Do not select anything

Discussion:
Nothing from this entire section should be installed on the server. The monitoring server has a narrow scope functioning as an appliance server; for security purposes unnecessary packages are not installed on the system. The server will be running a version of Apache that we will download, configure, compile and install for security reasons.

Development Section

Development Tools
Select this package, click on the details and do not select any of the optional packages. There were 23 packages selected.

Kernel Development
Select this package, click on the details and do not select any of the optional packages. There should be 3 base packages selected.

X Software Development
Do not select anything.

GNOME Software Development
Do not select anything.

KDE Software Development
Do not select anything.

Discussion:
On the surface selecting development tools seems like a contradiction to security best practices. Installation of development tools, especially kernel source, is a bad idea opening the system to unnecessary exposure. Big Brother needs development tools like Perl, and the tools make system management much easier because of their programming power. The choice of purchasing an inexpensive system with a non-standard NIC forces the presence of kernel source on the system. The NIC needs to be compiled into the kernel. Without the kernel source, every upgrade will require the addition of the kernel source prior to the NIC being added into the kernel. Afterwards the kernel source could be removed. This constant adding and removing of software leaves an exposure because eventually something will be missed causing a possible business interruption. If the kernel source is known to be on the system, the administration staff will need be more careful on the system. Since this is the first Linux server in the enterprise, a Linux tool server doesn't exist. There have been reports of Big Brother not running well on a stock Red Hat 9.0 kernel. Compiling the kernel seems to alleviate the memory problems some users of Big Brother seem to experience.

System Section

Adminstration Tools
Do not select anything.

System Tools
Do not select anything.

Printing Support
Do not select anything.

Miscellaneous
Minimal
Do not select anything.

Everything
Do not select this.

Discussion:
Choosing one of these selections will override the previous selections. We have chosen an installation containing a few more packages than the minimal installation, but not nearly as bloated and unsecured as selecting everything installation. A minimal install would have worked for us if the small number of development tools and kernel source were not required.

The package size is 804 MB.

Sixteenth Screen
Name: About to Install
Response: Click on next
Discussion:
Time to configure system and install software

Seventeenth Screen
Name: Boot Diskette Creation
Response: No, I do not want to create a boot diskette.
Discussion:
Twenty U.S. dollars was saved by purchasing the system without a diskette drive.

The GUI disappears and the system reboots. Please note during the installation we were not prompted for an IP address due to the non-standard NIC not being recognized. The network will be configured after the NIC driver is installed.

2.3 Secure the Operating System

Patch the system:

The base operating system has been installed and needs to be updated with patches and bug fixes.

Copy the patches to a separate directory

```
# mkdir –p /usr/local/patches/20040118/kernel
# mkdir –p /usr/local/patches/20040118/system
# mount /mnt/cdrom
# cd /mnt/cdrom
# find . –print | cpio –pduvm /usr/local/patches/20040118/system
#  cd /usr/local/patches/20040118/system
```

Move the kernel update to a separate directory

```
# mv *kernel* ../kernel
```

Update the applications

```
# rpm –F *.rpm
```

Check the machine still boots.

```
# cd /
# shutdown –r now
```

Update the kernel

```
# cd /usr/local/patches/20040118/kernel
# rpm –F *.rpm
```

Check the machine boots with the new kernel

```
# cd /
# shutdown –r now
```

Remove the old kernel

```
# rpm –e kernel-2.4.20-8
```

2.3a) RPM updates to Address Vulnerabilities

The base operating system is now installed, but it is far from secure. Before our energies are directed towards the hardening process we must install a NIC driver into the kernel. Our inexpensive Dell system was shipped with a NIC card not supported by the Linux 2.4 kernel. We need to download and install a Broadcom 5400 NIC driver. The NIC driver will continue to be an issue; every kernel upgrade will require the NIC driver to be re-installed. This is one reason we allowed development tools and kernel source to be installed on the system.

Here is the procedure to update the Broadcom 5400 NIC.

Install the source RPM Package:

Download the Broadcom driver from:
http://www.broadcom.com/

This was done on another PC and a CD-Rom was created with the Broadcom network RPM.

```
# mount /mnt/cdrom
# cd /mnt/cdrom
# rpm -ivh bcm4400-bcm4400-3.0.7-1.src.rpm
# cd /usr/src/redhat
# rpmbuild -bb SPECS/bcm4400.spec
# insmod bcm4400
# netconfig
# umount /mnt/cdrom
# shutdown -r now
# rpm -q -a | grep bcm
bcm4400-3.0.7-1
```

The Operating System is now installed and patched. We now turn our attention to making the system secure. In our initial statement we stated our goal was to build an appliance machine with a CIS rating of 9.5 or better. When everything is installed the CIS benchmark tool will be run to verify the security rating of the system.

2.3b) Boot Services

The umask for daemon processes is set in the /etc/rc.d/init.d/functions. The daemon umask is verified by using the grep command against the file /etc/rd.d/init.d/functions. The grep command displays the umask value set in the file, and it is set correctly. Any third party software packages added later will not be affected by a restrictive umask. This check verifies files created by daemon processes have the proper permissions.

```
# grep umask /etc/rc.d/init.d/functions
# Make sure umask is sane
umask 022
```

2.3c) Sendmail

The business function of this server is enterprise monitoring, the server needs to send mail but not receive mail. Outbound mail will be sent via a relay server. Even though the sendmail daemon will not be started on the server, this will not prevent the server from sending mail. E-mail will be the notification vehicle used by Big Brother to notify staff when there is a Big Brother issue. The sendmail daemon is configured not to start as a daemon and to interrogate the mailq every 15 minutes. In theory this should send mail every 15 minutes.

```
# cat /etc/sysconfig/sendmail
DAEMON=no
```

```
QUEUE=15m
# chown root:root /etc/sysconfig/sendmail
# chmod 644 /etc/sysconfig/sendmail
# ls -l /etc/sysconfig/sendmail
-rw-r--r--    1 root     root          20 Feb  7 21:17 /etc/sysconfig/sendmail
```

2.3d) Prevent Anonymous Shutdowns

Out of the box Red Hat operating systems are configured to reboot the system with the key sequence <ctrl><alt><del>. This window's like behavior is not acceptable in a secure environment. Even though our server is secured in a locked room, this behavior is not desirable. A disgruntled administrator can give a 3-fingered salute and reboot the monitoring node. Worse, an attacker can salute the machine and disable the monitoring system. The chances of the system being maliciously disabled through this key sequence are remote, but disabling this behavior fits into our paradigm of defense in depth and not taking chances.

Change this:

```
# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

to:

```
# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/usr/bin/logger -p authprov.info 'ctrl-alt-del trapped'
```

2.3e) Password protect single-user mode

Red Hat Linux allows users to enter single-user mode by typing "*Linux single*". This is a security risk allowing users to acquire root-level access without a password. This behavior does not fit into our security minded business model. If a user were allowed to type "*Linux single*" into the GRUB loading program, root access would be granted without any authentication or accountability. In reality, physical access equals root access. If a user has physical access to the system console, a password is merely a nuisance that can be circumvented easily by booting from cd-rom. However, our job is to make the attackers life as difficult as possible. To protect single-user access with a password we need to edit /etc/inittab.

Edit the /etc/inittab file and add these two lines:

```
# Password protect single-user mode
~~:S:wait:/sbin/sulogin
```

Verify the file has no syntax errors:

```
# init q
```

2.3f) Warning Banners

The login banner was modified to explain the acceptable use policy on Foobar's computing resources. The banner was added for protection of Foobar in the event of

legal action. The acceptable use policy is displayed to all users at login time, this banner warns users their actions may be monitored and/or recorded.

```
# cat /etc/motd
****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************
```

2.3g) SSH Configuration

Company security policy forbids the use of clear text, unsecured protocols such as telnet and rlogin. Hacking and exploitation tools are readily available on the Internet allowing attackers to leverage unsecured protocols to their advantage. The use of clear text protocols represents a security risk that is easily mitigated by using encrypted access methods like ssh. Ssh is the protocol for users connecting to our Big Brother machine. Even though this is a secure communications protocol, we cannot be too careful with the setup of ssh. A simple rpm query will list what version of ssh we are running:

```
# rpm -q -a | grep ssh
openssh-3.5p1-6
openssh-server-3.5p1-6
openssh-clients-3.5p1-6
```

There was a cert bulletin released concerning an exposure in OpenSSH, version 3.5p1-6. The latest stable, bug free, security patched RPM was downloaded and applied to our system. The RPM's were downloaded from http://ftp.lug.udel.edu/pub/OpenBSD/OpenSSH/portable/rpm/RH90/. One of our hardening principles is keeping all software up-to-date with security and bug fixes.

```
# rpm -F openssh-3.7.1p2-1.i386.rpm
# rpm -F openssh-clients-3.7.1p2-1.i386.rpm
# rpm -F openssh-server-3.7.1p2-1.i386.rpm
# rpm -q -a | grep ssh
openssh-3.7.1p2-1
openssh-clients-3.7.1p2-1
openssh-server-3.7.1p2-1
```

There are two configuration files for ssh, /etc/ssh/ssh_config and /etc/ssh/sshd_config. The system wide configuration file is /etc/ssh/sshd_config, and the per user configuration file is /etc/ssh/ssh_config.

/etc/ssh/ssh_config

This is the per use configuration file.

Protocol 2
Ssh has the ability to run either protocol 1 or protocol 2. Due to several exploits for protocol 1, the ssh daemon will only accept protocol 2 connections. The default value is 1, 2.

/etc/ssh/sshd_config
This is the system-wide ssh configuration file.

Protocol 2
This directive specifies the protocol version sshd will use for connections. Several exploits against protocol 1 have rendered it insecure by the security community. Therefore protocol 1 should be avoided at all costs. The default value is 2,1.

PermitRootLogin no
This directive prohibits root from connecting directly to the system via ssh. This does not affect the ability of root access through the system console, only direct root logins using ssh. This aligns nicely with our goal of not allowing root access anywhere except the system console. The default value is yes.

LogLevel VERBOSE
This specifies the verbosity level of message logging from sshd. The next level of verbosity is DEBUG, which would violate the privacy of users and will not be used. The default value is INFO.

LoginGraceTime 90
The server will disconnect after 90 seconds if the user has not finished authenticating. The default is 600.

IgnoreRhosts yes
Setting this directive to yes will cause ssh not to use .rhosts or .shosts in RhostsAuthentication, RhostsRSAAuthentication or HostBasedAuthentication. The default is yes.

RhostsAuthentication no
Authentication using only /etc/hosts.equiv or a rhosts files is not sufficient, this method is insecure and will not be permitted. The default value is no.

RhostsRSSAuthentication no
This option specifies whether authentication using rhosts or /etc/hosts.equiv, together with RSA host authentication is allowed. This directive is available only under protocol 1. Even though we are using protocol 2, we are adding this directive to tighten things up a bit. The default value is no.

HostBasedAuthentication no

This option specifies whether /etc/hosts.equiv or rhosts authentication together with successful public key client host authentication is allowed. The default value is no.

PermitEmptyPasswords no
If password authentication were allowed, the server would permit accounts to login with empty password strings. Any sane administrator would not allow access with an empty password string. Although the default value of no is what we want, removing the comment makes our intention more clear and serves as improved configuration documentation. The default value is no.

X11Forwarding yes
This directive allows X11 forwarding. On the surface it would appear disabling this option would increase security, in reality it is very easy for a user to install software permitting X11 forwarding. X11 forwarding is automatically disabled if UseLogin is enabled. The default value is no.

Banner /etc/motd
This is the warning message sent prior to user authenticating; our banner defines the acceptable usage policy for Foobar. The content of this file is sent to the user's terminal before authentication is allowed. This option is available only for protocol 2.

StrictModes
StrictModes is set by default; in the paranoid mindset this option should be set. Later ssh will be used to retrieve data from Big Brother client machines, this option is very important to prevent an escalation attack. A trust relationship is setup between the Big Brother node and the DMZ clients in the enterprise. As a security professional we must verify this relationship is secure. This option instructs sshd to verify the permissions and ownership on the remote ssh authentication files. If the permissions on the user's directories or ssh files or set too loose a password will be used for verification and the trust relationship is bypassed. Nice security feature.

The configuration of ssh is complete; the daemon is recycled to verify the configuration.

```
# /etc/rc.d/init.d/sshd stop
Stopping sshd:                             [  OK  ]
# /etc/rc.d/init.d/sshd start
Starting sshd:                             [  OK  ]
```

If you see anything other than "[OK]" during daemon initiation, there is a configuration error that must be corrected immediately.

Root Access
One of our prime goals as stated in section one is protection of the root account. Root access is the ultimate goal of an attacker; once the root account has been compromised, the game is over and the attacker wins. Tasks requiring direct root access are limited to the system console. There is no valid business reason any staff needs to log directly into the machine as root. The /etc/securetty file is edited to limit

the number of consoles available for root access. Our system console is limited to four tty's, the contents of the file is now:

```
# cat /etc/securetty
tty1
ttty2
tty3
tty4
```

The number of direct root avenues is now restricted to 4 tty's. This information needs to be kept from any outside prying eyes; access to the file will be restricted to root. [1]

```
# chmod 0400 /etc/securetty
# chown root:root /etc/securetty
# ls -l /etc/securetty
-r--------    1 root     root           20 Feb  7 11:32 /etc/securetty
```

2.3h) Kernel Tuning

The network related kernel parameters are stored in the /etc/sysctl.conf file. On Red Hat 9.0 the parameter's are loaded at boot time with the command **/sbin/sysctl -p /etc/sysctl.conf**. If an attacker were to disable the monitoring node, their attacks would be done in stealth mode since the monitoring node is disabled. The Big Brother server needs protection from network level attacks and exposures by tuning kernel parameters; this will increase the node's security. This section demonstrates how we add depth to our defenses by configuring kernel parameters protecting our machine against network attacks and exploits. The Big Brother server is protected from the outside world with firewalls and routers, if an attack is launched from the inside, this server would be a prime target due to its business function of infrastructure monitoring. If an attacker successfully disables monitoring to the infrastructure, he/she can roam about with one less method of detection. Increasing network security begins with intelligent kernel configuration set at boot time by the parameters selected below. The settings chosen are based on the recommendations from the SANS Track-6 Practicum by Hal Pomeranz. The reference guide for these parameters can be found at: http://lxr.linux.no/source/Documentation/networking/ip-sysctl.txt. This online documentation is excellent. The entire /etc/sysctl.conf file is listed in the appendix.

Below is the list of kernel parameters modified on the Big Brother server and the reason for the modified value.

ip_forward = 0

This parameter controls IP packet forwarding between interfaces. The server will not be acting as a gateway or router; in fact this is a behavior we would like to avoid. If the machine is dual homed, the machine can pass traffic from one interface to another

[1] Hudak, Typler, Sibley, Brad. OpenSSH A Survival Guide for Secure Shell Handling Version 1.0. Washington. The SANS Institute, 2003 This book was used as the refernce guide for section 2.3g.

bypassing the routing and switching equipment. Another effect is this can cause our machine to act as a launching pad for attacks to other machines on the network. This may also cause a DOS attack since our machine may be too busy forwarding packets to do monitoring work for us. In theory, a machine may become so overloaded servicing packets it will be unable to do anything else. In effect the monitoring function has been disabled, allowing an attacker to roam our infrastructure without being noticed by Big Brother. In reality, with today's more powerful hardware this type of DOS attack may be difficult to achieve. The default value is disabled.

rp_filter = 1
Several of the configuration parameters selected have dealt with IP spoofing, and here is another anti-spoofing defense. The parameter rp_filter is a kernel parameter protecting the machine against IP spoofing via source route verification. This reverse path filtering examines the source address of incoming packets against the interface the packet should be sent on; if the packet and the interface do not match the packet is dropped because it may be a spoofing attempt. Our machine has no business accepting packets from suspicious sources; this practice leaves an unnecessary exposure to our monitoring machine. A caution is issued in the documentation warning this option may cause troubles for complicated networks, networks utilizing slow, unreliable protocols, or networks using static routes. The default value is 0, no source validation.

tcp_max_syn_backlog = 4096
This is a parameter to protect syn floods by allowing 4096 remembered connections that have not received an acknowledgement from the connecting client. This is a common DOS attack method for hackers to render a machine unusable by using all the TCP connections of a machine. This is important to add to the configuration to prevent an attack from disabling the monitoring machine. The default value is 1024 for servers with 128 MB or more of memory, and a value of 128 for machines with memory less than 128 MB.

accept_source_route = 0
This option addresses packets with a SRR option enabled forcing IP source routing; this will be disabled in our configuration. The default is TRUE for a router and FALSE for a host.

accept_redirects = 0
The business function of our machine is monitoring the Unix infrastructure of the company; this machine does not function as a router or a gateway. Aligning with our business function of this machine, we disable the ability to do ICMP redirects. Enabling this option would allow our machine to act as a router or a gateway, exposing our machine to launch pad attacks where an attacker uses our machine to attacker another. Worse, a flood of ICMP directs could cause a DOS attack, our machine would be too busy redirecting ICMP to monitor the infrastructure. The default value is enabled if local forwarding is disabled and disabled if local forwarding is enabled.

send_redirects = 0
The default behavior of this parameter does not meet our machine's business function as a monitoring server. This parameter should only be set if the machine is functioning as a router or gateway, obviously a monitoring server does not meet those criteria. By default this parameter is set to send ICMP redirects to another destination. If this is enabled our machine could be used as a launching pad for an attack. If the machine were flooded with packets a DOS attack could result since the machine may be so busy servicing packets it wouldn't have time to do anything else. As mentioned earlier, with today's more powerful hardware this DOS attack may be difficult to achieve. The default value is TRUE.

tcp_syncookies = 1
One of the most likely attacks against our machine could be a syn flood attack. This option helps prevent a syn flood attack, but is only available to a kernel compiled with the CONFIG_SYNCOOKIES option. This parameter will send out syncookies when the syn backlog queue of a socket overflows. The default value is 0.

log_martians = 1
Okay I have to admit, I am very fond of this parameter's name. I am reminded of the television show My Favorite Martian. Cute name aside, this parameter is very important for the security conscious administrator. Enabling this parameter logs packets arriving from impossible source addresses, typically sent from an attacker trying an address spoof on our machine. The default value is 0.

secure_redirect = 0
Accept ICMP redirect messages only for the gateways listed in the default gateways list. Our machine is being engineered to avoid possible attacks where we forward packets to other addresses on the network. Enabling this parameter exposes our machine to a DOS style attack. Any DOS attack on this machine will disable the monitoring to the infrastructure allowing an attacker to roam the network without being detected by Big Brother. The default value is true.[2]

The machine is up and running and we are on the console. The network services will be stopped and the new values will be loaded into the kernel. Stopping and starting the network services will check our kernel file /etc/sysctl.conf, and report any errors in the file. Verifying the network settings will avoid a hung machine during reboot if there is an error in the file.

```
# /etc/rc.d/init.d/network stop
Shutting down loopback interface:                [  OK  ]
# /etc/rc.d/init.d/network start
Setting network parameters:                      [  OK  ]
Bringing up loopback interface:                  [  OK  ]
```

[2] http://lxr.linux.no/source/Documentation/networking/ip-sysctl.txt This document was used as a reference guide for the entrire section under 2.3h Kernel Tuning.

2.3i) Network services on out of the box

By default there are too many active network services on the Red Hat operating system, an appliance server requires only a small number of active network services. One of our hardening principles is running only the services needed by the business; any unnecessary services will be disabled. Following is a table listing the current active services.

Active

```
# chkconfig --list | grep :on
```

Service	Runlevel 0	Runlevel 1	Runlevel 2	Runlevel 3	Runlevel 4	Runlevel 5	Runlevel 6
kudzu	0:off	1:off	2:on	3:on	4:on	5:on	6:off
syslog	0:off	1:off	2:on	3:on	4:on	5:on	6:off
netfs	0:off	1:off	2:on	3:on	4:on	5:on	6:off
network	0:off	1:off	2:on	3:on	4:on	5:on	6:off
random	0:off	1:off	2:on	3:on	4:on	5:on	6:off
rawdevices	0:off	1:off	2:on	3:on	4:on	5:on	6:off
gpm	0:off	1:off	2:on	3:on	4:on	5:on	6:off
pcmcia	0:off	1:off	2:on	3:on	4:on	5:on	6:off
saslauthd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
keytable	0:off	1:on	2:on	3:on	4:on	5:on	6:off
apmd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
atd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
xfs	0:off	1:off	2:on	3:on	4:on	5:on	6:on
autofs	0:off	1:off	2:on	3:on	4:on	5:on	6:off
irda	0:off	1:off	2:on	3:on	4:on	5:on	6:off
nscd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
isdn	0:off	1:off	2:on	3:on	4:on	5:on	6:off
sshd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
portmap	0:off	1:off	2:on	3:on	4:on	5:on	6:off
nfs	0:off	1:off	2:on	3:on	4:on	5:on	6:off
nfslock	0:off	1:off	2:on	3:on	4:on	5:on	6:off
crond	0:off	1:off	2:on	3:on	4:on	5:on	6:off
anacron	0:off	1:off	2:on	3:on	4:on	5:on	6:off
xinetd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
ntpd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
snmpd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
snmptrapd	0:off	1:off	2:on	3:on	4:on	5:on	6:off

Below is a desired list of services required to meet the business and security needs of our customers.

Desired

Service	Runlevel 0	Runlevel 1	Runlevel 2	Runlevel 3	Runlevel 4	Runlevel 5	Runlevel 6

network	0:off	1:off	2:on	3:on	4:on	5:on	6:off
random	0:off	1:off	2:on	3:on	4:on	5:on	6:off
pcmcia	0:off	1:off	2:on	3:on	4:on	5:on	6:off
keytable	0:off	1:on	2:on	3:on	4:on	5:on	6:off
iptables	0:off	1:off	2:on	3:on	4:on	5:on	6:off
sshd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
crond	0:off	1:off	2:on	3:on	4:on	5:on	6:off
anacron	0:off	1:off	2:on	3:on	4:on	5:on	6:off
ntpd	0:off	1:off	2:off	3:on	4:off	5:on	6:off

Here is a simple script to modify the boot services only starting the services we need.

```
# for service in kudzu syslog netfs pcmcia apmd \
atd gpm autofs isdn portmap nfslock sendmail \
rhnsd xfs xinetd cups
do
        chkconfig  ${service} off
done

# chkconfig –list | grep :on
# cat /tmp/services
```

Service	Runlevel 0	Runlevel 1	Runlevel 2	Runlevel 3	Runlevel 4	Runlevel 5	Runlevel 6
network	0:off	1:off	2:on	3:on	4:on	5:on	6:off
random	0:off	1:off	2:on	3:on	4:on	5:on	6:off
pcmcia	0:off	1:off	2:on	3:on	4:on	5:on	6:off
keytable	0:off	1:on	2:on	3:on	4:on	5:on	6:off
iptables	0:off	1:off	2:on	3:on	4:on	5:on	6:off
sshd	0:off	1:off	2:on	3:on	4:on	5:on	6:off
crond	0:off	1:off	2:on	3:on	4:on	5:on	6:off
anacron	0:off	1:off	2:on	3:on	4:on	5:on	6:off
ntpd	0:off	1:off	2:off	3:on	4:off	5:on	6:off

Another guideline used in our hardening process is conforming to the standards set forth by the Center for Internet Security. I have used the CIS automated tool many times and I am familiar with what is expected by the tool. From past experience I know the csi-scan tool will flag the daemon sgi_fam, we have no need for the daemon, and so it will be disabled.

```
# cat  /etc/xinetd.d/sgi_fam
# default: on
# description: FAM is a file monitoring daemon. It can \
# be used to get reports when files change.
service sgi_fam
{
        type         = RPC UNLISTED
        socket_type  = stream
        user         = root
        group        = nobody
        server       = /usr/bin/fam
        wait         = yes
```

```
        protocol     = tcp
        rpc_version  = 2
        rpc_number   = 391002
        bind         = 127.0.0.1
        disable      = yes
}
```

Please notice the syslogd daemon is deactivated at this point; a security conscious server running without central logging does not fit the best practices for security. Later we will be adding syslog-ng to replace the standard Unix syslogd. As a starting point the services running are minimal, and a good starting point for adding business critical secure services.

2.3j) System login cleanup

The Big Brother server has a very narrow business function. Exposure is being reduced as much as possible by removing unused services, applications, users, and groups. By default the system contains user and group accounts responsible for running daemons and system services. The goal is to run the system with a minimum number of services and applications; users and groups associated with disabled or removed applications will be expunged from the system. One of our hardening deliverables is the removal of user and group accounts not needed on the system. A weekly maintenance script will be run listing all current users. This documentation will be our beacon during maintenance and upgrades. A hurdle faced during an upgrade or maintenance is users and packages being added to the system after they were removed during the initial setup. A small password file means decreased risk to our system by reducing the amount of avenues into our system. The only Foobar employees granted command line access to the server will be Unix system administrators.

We are going to remove the following users:

```
for user in mail news uucp operator games gopher \
        ftp nobody rpm vcsa nscd rpc rpcuser nfsnobody \
        mailnull smmsp pcap xfs gdm
do
        echo "removing ${user}..."
        userdel -r ${user}
done
```

At this point the only user with a valid shell is root.

Below is our modified and tightened password file.

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/dev/null
daemon:x:2:2:daemon:/sbin:/dev/null
adm:x:3:4:adm:/var/adm:/dev/null
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
```

```
halt:x:7:0:halt:/sbin:/sbin/halt
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/dev/null
ntp:x:38:38::/etc/ntp:/dev/null
```

We are going to remove the following groups:

```
for group in mail news uucp operator games gopher \
        ftp nobody rpm vcsa nscd rpc rpcuser nfsnobody \
        mailnull smmsp pcap xfs gdm
do
        echo "removing ${group}..."
        groupdel -r ${group}
done
```

Although we have some development tools on our server, the server will never be used for development purposes. Therefore, there is no business reason to generate core files, have a large number of open file descriptors, or a large number of processes. The limits.conf file has been modified to protect the system from a user exhausting resources and causing a DOS attack. Below is our modified /etc/security/limits.conf file.

```
/etc/security/limits.conf

*               soft    core            0
*               hard    core            0
*               hard    rss             102400
*               soft    nofiles         512
*               hard    nofiles         1024
*               soft    nproc           512
*               soft    maxlogins       3
```

2.3k) Syslog-ng and Stunnel

In a security focused environment logging is a mission critical business function. In the event of an incident, log files are analyzed and saved as evidence. On a more basic level, logging is extremely useful for root cause analysis when there is an application or system problem. System log files are the pulse of our systems; judicious logging can help keep a system functioning well by spotting problems early. The standard logging facility on most flavors of Unix is the venerable syslogd daemon. Although syslogd does an admirable job, it really doesn't fit into our secure computing model. The reasons why syslog does not fit into our model are listed below. Below is a comparison of syslog and syslog-ng. Our replacement choice is syslog-ng written by Balazs Scheidler. It is a widely accepted replacement for the aging, vanilla syslogd facility.

Log messages from standard syslogd are not consistent between applications and operating systems; this makes log analysis difficult and parsing a nightmare. Syslog-ng allows the administrator greater control of the message format, creating a standard message format controlled amongst heterogeneous operating systems and applications. Message parsing becomes a much more manageable task.

Log filtering by priority and facility is not very flexible, critical messages are often lost among informational messages that can be ignored. Syslog-ng allows the administrator to group messages into different categories allowing message parsing to be more manageable. Messages are less likely to get lost since they are grouped together, and log files are sorted by host instead of one unwieldy, gigantic log.

If syslogd is sending messages to a central log host the transport protocol is UDP, which is not reliable or guaranteed. In a secured environment guaranteed delivery and storage of messages on a central logging host is critical. Our logging facility must guarantee delivery and be secure during transport. If a system is compromised and the case is taken to court, a smart lawyer could use UDP protocol against us, since it is not guaranteed.

Syslogd uses UDP as its transfer protocol, and the data is delivered in a clear text format to the logging host. An attacker using a sniffer such as Ethereal would have no trouble catching packets on the way to the logging host. The information being sent to the logging host is sensitive and would be highly desirable to an attacker. Information such as IP addresses, user ids, and application names would be highly prized information of interest to an attacker. The data transport protocol used to transfer the data must be secure to protect the data. Syslog-ng uses TCP for its protocol, which works well with stunnel creating an encrypted transport mechanism. The combination of syslog-ng running over stunnel creates a desirable logging combination, and one that will be running in our enterprise.

It is possible to send messages through multiple hosts on its journey to the central logging host. Standard syslog will only display the last hostname in the journey to the central logging host before arrival. In the event of a security incident, security and administration staff will have to trace the path of a message back to its original host. Valuable time could be lost, or worse data could have been molested on its way to the host. The true path of the message may never be known. Syslog-ng has the ability to show the entire path traveled en route to the central logging host.

Log files are clear text files vulnerable to modification by a hacker; our desire is to store one copy of the log files locally, and another copy on the hardened central logging host. The central logging host responsible for storing the logs must be a very secure, hardened system.

Based on the criteria above there is a very strong case for implementing syslog-ng over standard syslogd. [3]

Syslog-ng is freeware downloaded from http://www.balabit.com/downloads/syslog-ng/1.5/src/. As of this writing the latest stable version is 1.5.26. Syslog-ng requires the installation of a library that also can be downloaded from Balazs Scheidler. The library

[3] Cyrus Peikari,Cyrus, & Chuvakin, Anton, Security Warrior. Sabastopol, O'Reilly & Associates, 2004 pp 412- 413 Security Warriors was used for the syslog-ng section as a reference guide.

is libol and the latest stable version as of this writing is 0.3.9. The library can be downloaded from http://www.balabit.com/downloads/syslog-ng/libol/0.3/.

In the interest of space the command output was not included.

```
#  tar xvfz libol-0.3.9.tar.gz
# cd lib-0.3.9
# ./configure
# make
# make install
```

The library for syslog-ng is installed and ready to use, we now turn our attention to the installation of syslog-ng.

```
# tar xvfz syslog-ng-1.5.26.tar.gz
# cd syslog-ng-1.5.26
# ./configure
# make
# make install
```

The code for syslog-ng is installed at /usr/local/sbin/syslog-ng. I didn't change any of the defaults so the configuration information will need to be stored at /usr/local/etc/syslog-ng/syslog-ng.conf

The file permissions should be very restrictive to limit preying eyes from looking at files they have no business viewing. Permissions are very restrictive to allow only root access to the directory and configuration file.

```
# mkdir -p /usr/local/etc/syslog-ng
# cd /usr/local
# chown -R root:root etc
# chmod 0500 -R etc
```

Enterprise configuration files are stored on a central tool server. The central tool server is very secure and stores all enterprise wide configuration files such as syslog-ng.conf, sudors, ntp.conf, etc. The master copy of the syslog-ng.conf file is copied from the tool server to the monitoring server.

```
# cd /usr/local/etc/syslog-ng
# scp jhorwath@toolbox:/usr/configuration/syslog-ng.conf .
# scp jim@toolbox:/var/tmp/syslog-ng.conf .

******************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
```

```
********************************************************************************

jim21349@toolbox's password:
syslog-ng.conf      100% |*****************************************************************|  3418      00:00

# cat syslog-ng.conf
```

Since the server configuration deals with implementing a central logging host, and is beyond the scope of this discussion the finer points will not be expanded upon. One point for improvement would be the development of a RPM for syslog-ng and libol. During a software inventory they will not be displayed in a RPM query, this is not really the best choice for a security conscious environment.

We need to modify our /etc/services file to define where syslog-ng runs. This will be important because our goal is to run syslog-ng over stunnel for a secure, encrypted logging mechanism. Stunnel will have to know which port syslog-ng is running on. The standard is to use port 514/tcp for syslog-ng over stunnel for secure communication.

Add this line to /etc/services:

```
syslog-ng           514/tcp         # syslog-ng new generation syslogger
```

The final step in configuring secure syslog-ng is to wrap the logging data with stunnel. Stunnel will wrap our syslog data with an encrypted SSL tunnel during transport from our Big Brother host to the central logging server. The central logging host is a hardened for maximum security. A technical review of the central logging host is beyond the scope of this discussion. The completed syslog-ng file can be referenced in the appendix section at the end of this document.
Start syslog-ng

```
# /etc/rc.d/init.d/syslog-n start
Starting /usr/sbin/stunnel:
stunnel[27851]: stunnel 4.04 on i386-redhat-linux-gnu PTHREAD+LIBWRAP with OpenSSL 0.9.7a Feb
19 2003
stunnel[27851]: FD_SETSIZE=1024, file ulimit=1024 -> 500 clients allowed
Starting Kernel Logger:                    [  OK  ]
```

The system now has a reliable, encrypted method for transferring log data between the monitoring host and the central syslog server.

2.3l) NTP (Network Time Protocol)

A secured environment is not complete without NTP running. From a security and administration perspective, time synchronization of machines is important for log files, machine communications and administration. In the event of the environment being compromised, piecing together the when, how, why, and what happened is more difficult without the aide of NTP. The CSO mandated all computing equipment run NTP; as a

result we have a generic enterprise wide configuration file we will load. Based on the business function of the machine: production, development or UAT, each network uses different devices for time synchronization. Cisco routers function as NTP servers for Foobar. NTP uses very little resources in terms of CPU cycles or network bandwidth. Each network segment uses three different routers for time synchronization.

Following are the steps used to copy over the configuration file and start the service.

```
# cd /tmp
# scp jhorwath@toolbox:/usr/configuration/ntp.conf /tmp
# mv /tmp/ntp.conf /etc/ntp.conf
# chmod 440 /etc/ntp.conf
# chown root:root /etc/ntp.conf
# /etc/rc.d/init.d/ntpd restart
Shutting down ntpd:                    [  OK  ]
Starting ntpd:                         [  OK  ]
```

The corporate direction of using NTP on all computing resources makes good business and security sense. Time stamps will be synchronized across the enterprise. Analysis and tracking during troubleshooting is made easier due to time stamp synchronization on all the computing resources.

2.3m) DNS (Domain Name Service)

DNS is a mission critical operating system utility on the monitoring server. By default Big Brother uses DNS for name resolution. Name/IP entries in host files are not reliable and need to be avoided. The tool server contains the DNS configuration file for Foobar. Here are the steps used to copy over the configuration file:

```
# cd /tmp
# scp jhorwath@toolbox:/usr/configuration/resolv.conf   /tmp
# scp jhorwath@toolbox:/usr/configuration/host.conf /tmp
# mv /tmp/ntp.conf /etc/resolv.conf
# mv /tmp/host.conf /etc/host.conf
# chmod 444 /etc/resolv.conf
# chmod 444 /etc/host.conf
# chown root:root /etc/resolv.conf
# chown root:root /etc/host.cont
```

DNS will not be started on this machine until it is placed on the production network. The configuration goal is a machine ready to plug into the production network without having too much addition configuration.

2.3n) TSM (Tivoli Storage Manager)

Tivoli Storage Manager is the enterprise backup solution for Foobar; TSM is a network backup solution. When the backup data arrives at the TSM server, it is stored on DLT tape and SAN attached storage. Since the backups are network based, using TCP protocol, it is possible to encrypt the backups with stunnel. Using stunnel for backups in a medium to large production environment is not a good idea. The monitoring files will never be more than a few Gigabytes on the TSM client and will place very little load on

the monitoring server. The TSM server is a different story, there are nearly 2000 machines (Wintel and Unix) backed up nightly. Scheduling two thousand machines over a twenty-four hour period is difficult enough, adding a CPU intense decryption process makes scheduling nearly impossible. A couple hundred machines sending encrypted data requiring a decryption process on the TSM server will kill the server. The server will suffocate under the CPU intensive load. Given the alternatives, there isn't a large exposure because we are sending data to the TSM server without stunnel. The final production TSM installation must be coordinated with the TSM administrator because our client needs to be registered on the TSM server. Although the TSM client is installed, it is not activated until the machine is on the production network where it can access to the TSM server. The TSM rpm is stored on the tool server. Below are the steps used to copy over the configuration file:

```
# cd /tmp
# scp jhorwath@toolbox:/usr/configuration/resolv.conf   /tmp
# scp jhorwath@toolbox:/usr/configuration/host.conf /tmp
# mv /tmp/ntp.conf /etc/resolv.conf
# mv /tmp/host.conf /etc/host.conf
# chmod 444 /etc/resolv.conf
# chmod 444 /etc/host.conf
# chown root:root /etc/resolv.conf
# chown root:root /etc/host.cont
```

2.3o) SUDO

The Unix su command grants privileged access to the system, or allows one user to execute commands as another user. In order to have privileged access on machine a user must know the root password; one of our security guidelines is protection of the root password. This really creates a dilemma, administration staff needs the root password for job duties, but we are operating on the least privilege principle. The principle of least privilege is violated since using su to root gives all or none privileged access, and the privileged commands are not logged. For auditing and forensic purposes this does not fit into our model. Our desire is to have all privileged commands issued by users logged to the system logging facility. In the event of an incident that requires a Root Cause Analysis, command logging will aid with forensics. Only two staff members should know the root password: the security officer and the Unix system manager. The root password is sealed in an envelope and locked in a safe. The sudo configuration file is very simple allowing administrators access to all commands on the system. This model presents an exposure, we are relying on staff to be honest and not execute a shell escape to become root. If the user executes a shell escape the commands are no longer logged and our security measure has been bypassed. The sudo configuration file is stored on the tool server. Following are the steps used to copy the configuration file to the monitoring node:

```
# cd /tmp
# scp jhorwath@toolbox:/usr/configuration/sudoers   /etc/sudoers
# chmod 400 /etc/sudoers
# chown root:root /etc/sudoers
```

The sudoers file is large since it encompasses 300 operating systems and several groups, here are the sections pertaining to the administration group and the node kazoo.

```
#
# Alias to lump together all the administrators
#
Group_Alias             Admins = jhorwath, fflintstone, bbunny, brubble, \
                        funger, omadison
```

Leveraging the Group_Alias token allows easier maintenance of the sudoers file. Modification is made easier since all users are defined in one place. If a staff member changes responsibility it is easy to remove them from privileged access.

```
#
# Allow the administrators blanket access
#
Admins          ALL = ALL
```

Administrators are given unstricted access to all commands on the system.

```
#
# Under the defaults section
#
Defaults                    syslog=auth                 # syslog facility
Defaults                    !lecture                    # disable initial lecture
Defaults@HOST_ALIAS         log_year, log file=/var/adm/sudo.log
```

All commands issued by the sudo facility are logged via syslog-ng in the system log facility and in the /var/adm/sudo.log file. The messages logged via syslog-ng are stored locally and on the central syslog server. The lecture option of sudo is disabled; the first use of sudo will not spew the responsibility lecture about sudo. If a user leaves the company, or their responsibility changes the security office will contact the administration staff notifying them change in duties. This is done to allow administration staff to modify the sudoers file. In production the sudoer's file is modified then distributed via ssh over rsync throughout the enterprise.

Apache With mod_security

Big Brother is a Web based monitoring tool. The Big Brother display node requires an http service available to users allowing Big Brother to be displayed to users. The Big Brother display host can be on any machine. The simplicity and flexibility of Big Brother is a blessing and a curse; implementation of a Big Brother server is easy, however you are exposed in several areas if you are not careful with the system configuration. At a SANS conference in December 2003, I was introduced to mod_security. Ivan Ristic developed an Apache module based on SNORT, building an IDS into Apache. Although our Web server is shielded from the Internet by firewalls and routers, we need to protect ourselves from the internal black hats with mischief on their minds. The

addition of mod_security heavily influenced our decision towards using Apache for our Web server. Apache is the most widely deployed Web server in use today. Most Apache exploits are the result of either poor configuration or poorly written third party vendor software. This section will address the hardening of Apache for our monitoring server.

There are several http servers that were considered for our purposes, however Apache was chosen for the following reasons:

- Apache is free
- The source code is freely available allowing more control over how Apache is configured
- Compared with other Web servers Apache has experienced only small number of exploits since 1997
- Apache is a mature product widely used on the Internet. This high volume of usage will be used to our advantage by building on the experiences and mistakes of others.
- One of the most powerful additions to Apache is mod_security, an IDS system that plugs into Apache for an extra layer of defense.
- Did I mention Apache was free

As mentioned earlier Big Brother is easy to deploy. The ease of deployment can be the root of many security holes due to configuration errors.

We must use all of the features available through Apache and mod_seurity to our advantage. They will help us engineer a secure, but usable Web server. Our Web server has a very narrow business scope; it will serve only Big Brother data pages. We will not use this http server to list data center schedules, cafeteria menus, family pictures, MRTG graphs, off-hour schedules, or the myriad of other things you can display on Apache. As of this writing the Center for Internet Security has an Apache scorecard in the works, but it is not available to the general public. I have used the GCUX paper written by Ryan Barnett as a guideline for Apache security. Additionally I have used the SANS Institute Track 6 – Securing UNIX 6.4 Running UNIX Applications Securely book by Hal Pomeranz. Ryan's work is heavily referenced on the Internet whenever Apache security issues are discussed. I had the pleasure of attending a seminar hosted by Ryan and found it very lucid, informing and entertaining. Since we are interested in best practices, Ryan's road map was a natural fit. Ryan's guide may be found at: http://www.cgisecurity.com/lib/ryan_barnett_gcux_practical.html.
The latest stable release of Apache is 2.0.48, which is available at:
http://httpd.apache.org/download.cgi .
We will also need to download the mod_security code from:
http://www.modsecurity.org/download/index.html
The documentation for mod_securityis available at:
http::/www.modsecurity.org/documentation.html

Through this section Ryan's document was used as a road map along with the mod_security documentation provided by Ivan Ristic. The combination of Apache with mod_security will provide a formidable defense against intruders. Apache will be modified and secured to meet our business need of monitoring the Unix enterprise. Apache must fit in with our security initiative of a dedicated server for enterprise-wide monitoring. Due to source code customizations and a single installation, a RPM was not generated for Apache. This may change in the future, for right now Foobar does not have a RPM installation for Apache.

User and Group Configuration

During the system hardening process, the user and group nobody were removed from the system. The account nobody was originally created to work with NFS; a user of less notoriety will be chosen to run the httpd server. Somehow the user and group nobody has become the default user and group for Apache Web servers. Apache has a dedicated business function of serving Big Brother Web pages. Big Brother and Apache and must share Web files, the easiest method to meet this goal is to use Unix file permissions to share Web pages. If a compromise happens through the http account, it is not privileged and would result in minimal damage. Although Apache runs on a privileged port, once the startup is complete a non-privileged user can run the httpd server. We need to lock the account, preventing the user from accessing the system by assigning /dev/null as the user shell. Adding defense to depth we do not want to risk this account being compromised or exploited.

```
# groupadd bbgroup
# useradd -g bbgroup -d /usr/local/apache2 -c "Apache account" -s /dev/null bbweb
#  passwd -l bbweb
Locking password for user bbweb.
passwd: Success
```

Prepare for Apache installation by downloading and coping the tarball to the source directory.

```
# cp httpd-2.0.48.tar.gz mod_security-1.7.4.tar.gz /usr/local/src
# gunzip httpd-2.0.48.tar.gz
# tar xvf httpd-2.0.48.tar
```

Unpack and untar the **mod**_security source.

```
# tar xvfz mod_security-1.7.4.tar.gz
```

Configuration of Apache

Apache like most open source software allows the user flexibility to tailor the application to meet the business needs of an enterprise. A shell script will be created for the command line configuration options for our Apache installation. Although the code can be built via the command line, a few reasons are presented why a shell script was chosen to handle the configuration. The shell script will function as documentation for

the specific options used during the compilation process. This is important for upgrades requiring recompilation or deployment to other servers. As the number of options used to configure Apache grows in complexity, the opportunity to miss an option increases. The possibility exists of building a Web server with a missing key ingredient, or leaving a potential exploit open because of a forgotten option. Here is the configuration script.

```
./configure --enable-auth-dbm \
--enable-auth_digest \
--enable-rewrite \
--enable-usertrack \
--enable-vhost-alias \
--enable-ssl \
--disable-status \
--disable-mod_info \
--disable-include \
--disable-autoindex \
--disable-userdir \
--disable-so \
--disable-auth \
--disable-auth_anon \
--with-module=mappers:security \
CPPFLAGS=-I/usr/kerberos/include LDFLAGS=-L/usr/kerberos/lib
```

Option Discussion The How's and Why's

Enabled Modules
auth_dbm
Big Brother Web pages will be password protected to strengthen the access and integrity of information from outsiders. The password feature is compiled by in adding the auth_dm module. This stores username and passwords in a DBM file.

auth_digest
Our user community will be using Internet Explorer for their Web browser. This configuration allows the more secure MD5 authentication, instead of the basic authentication scheme.

vhost-alias
Needed SSL (Secured Socket Layer) module support, for future support of Web traffic over https.

ssl
Secure Socket Layers will allow us to encrypt our data as it travels to the user's browser, using https.

rewrite
Used to control URL redirection and manipulation.

usertrack

This module will track users who make malicious requests. For a security minded deployment this is a necessity, any malicious activity will warrant further investigation for staff.

Disabled Modules
Below is a discussion regarding Apache modules not included in our configuration due to security concerns. The module and the reason for exclusion are briefly discussed below.

mod_auth
The mod_auth module implements user access via text files. Although this module forces authentication, storing user information in a plain text file is a bad practice. A text file that is not encrypted is ripe for comprise if a configuration error is made. User credentials should be stored in a DBM type file with MD5 authentication; this combination is much more secure and fits the security model we are engineering.

mod_auth_anon
Inclusion of the mod_auth_anon module would grant anonymous user access to authenticated areas, anonymous access does not fit into our security model. Security professionals want to know the who, what and why's of our systems. The system is not some sort of Intranet rest stop for users; anonymous users are not welcome on this web server.

mod_status
The mod_status module provides information regarding server activity and performance. On the surface this data is interesting from a performance and metrics point of view; from a security perspective our goal is to avoid advertising any information regarding our server to the outside world. Performance and metrics will be collected with other tools within the confines of the operating system.

mod_info
The mod_info module falls into the same category as mod_status, another module that does not fit into our security paradigm. Soon will we will obfuscate our Apache release and version levels by changing the source code. The mod_info module would compromise much of the work we are trying to accomplish by announcing server configuration to the outside world.

autoindex
Inclusion of this module allows a directory listing when the index page is missing; this is too much information displayed to the outside world. When the user requests the document root directory in the browser, and the index.html files does not exist, a directory listing is displayed in the browser window. The directory structure of our document root directory is valuable infrastructure information to an attacker. The attacker would have valuable directory content information that may be used to their benefit during an attack or an exploitation attempt.

userdir
Our Web server has a very specific business purpose. Earlier we stated the server would not host family pictures, cafeteria schedules, personal date books, etc. We are not allowing user directories within the web site. This will lessen the chance for exploitation and keep our Web server free from extraneous, non-business clutter.

mod_so
The module mod_so allows the loading of executable code and modules into the server at runtime. Inclusion of this module would provide the opportunity for an attacker to load a rogue module into the system. Obviously allowing a user the ability to load modules into the system presents a security threat.

Special Included Modules

-with-module=mappers:security
This directive informs the compilation process to include the mod_security module.

CPPFLAGS=-I/usr/kerberos/include
The CPPFLAGS I discovered from the school of hard knocks. Compiling SSL into Apache requires Kerbos for some odd reason. This fact was missing from the documentation, luckily a quick google search lead me in the right direction.

Edit the header file
In an effort to obfuscate as much information as possible, the Apache header is modified to hide server configuration information. This command sequence can reveal a lot to an attacker:

```
telnet www.foo.org 80
Trying X.y.z.q
HEAD / HTTP/1.1
Host www.foo.org
HTTP/1.1 200 OK
Date: Sun, 21 Mar 2004 12:00:13 GMT
Server: Apache/2.0.48 (Unix) mod_perl/1.25
Last-Modified: Wed, 01 Oct 2003 13:35:32 GMT
ETag: "27cc2-654-3f7ad824"
Accept-Ranges: bytes
Content-Length: 1620
Connection: close
Content-Type: text/html
```

This will spew more information than we want to reveal. The default value returned by Apache includes version and release numbers, as well as the identity of some modules included. Later during the configuration of the httpd.conf file, further modifications will be made to prevent information like the above from being revealed. We will edit the ap_release.h file in the Apache include directory. This edit will change the displayable server information, while still giving credit to The Apache Software Foundation.

```
# cd/usr/local/src/http-2.0.48/include
# vi ap_release.h
```

From

```
#define AP_SERVER_BASEVENDOR "Apache Software Foundation"
#define AP_SERVER_BASEPRODUCT "Apache"
#define AP_SERVER_MAJORVERSION "2"
#define AP_SERVER_MINORVERSION "0"
#define AP_SERVER_PATCHLEVEL "48"
```

To

```
#define AP_SERVER_BASEVENDOR "Apache Software Foundation"
#define AP_SERVER_BASEPRODUCT "Some Server"
#define AP_SERVER_MAJORVERSION "99"
#define AP_SERVER_MINORVERSION "99"
#define AP_SERVER_PATCHLEVEL "99"
```

The intent of this change is to confuse and frustrate a less skilled attacker. Credit needs to be given to Apache Software Foundation for the product, so the AP_SERVER_BASEVENDOR will remain. There is a small exposure if an attacker compromises the machine and runs strings on the httpd executable, he/she will see the string Apache Software Foundation in the output. If the attacker is able to strings the httpd binary, we have larger issues because the permissions on the Apache binary will be restrictive. [4]

Precompilation

There are a few steps required to build mod_security into Apache. Although the module is easy to compile, the documentation on how to build it is still a bit sketchy. Ivan Ristic discusses this in the mod_security documentation. There is the possibility we may chroot the Apache server. To make life much simpler a static compilation is chosen instead of a dynamically compiled executable. This is taken directly from the mod_security documentation.

```
Inform the compilation process to include mod_security
-with-module=mappers:security
Copy mod_security.c to modules/mappers
Edit modules/mappers/modules.mk
add:
mod_security.la: mod_security.lo
        $(MOD_LINK) mod_security.lo
static = mod_security.la[5]
```

[4] http://www.cgisecurity.com/lib/ryan_barnett_gcux_practical.html. This document served as the referencem for the entire Apache configuration section.

[5] http://modsecurity.org/documentation/modsecurity-manual-1.7.4.pdf

Compilation

```
# make
```

Compilation checking

The compilation step completed without incident. The following steps will verify the Apache version, modifications, and the modules compiled into Apache.

Verify the mangled Apache version:

```
# ./httpd -v
Server version: Some Server/99.99.99
Server built:   Feb 14 2004 10:29:06
```

Verify the compiled modules:

```
# ./httpd -l
Compiled in modules:
  core.c
  mod_access.c
  mod_auth_dbm.c
  mod_auth_digest.c
  mod_log_config.c
  mod_env.c
  mod_usertrack.c
  mod_setenvif.c
  mod_ssl.c
  prefork.c
  http_core.c
  mod_mime.c
  mod_security.c
  mod_asis.c
  mod_cgi.c
  mod_vhost_alias.c
  mod_negotiation.c
  mod_dir.c
  mod_imap.c
  mod_actions.c
  mod_alias.c
  mod_rewrite.c
```

Installation

```
# make install
```

This command moves the Apache code to the /usr/local/apache2 directory, it is now ready for configuration. I had thought about installing Apache into a different directory such as /usr/local/http, or /usr/local/someserver. After going through the trouble of modifying the Apache source code to hide critical server information, the path

/usr/local/apache2 will appear in the process table. It doesn't take a rocket scientist to figure out what version and type of http server we are running. We will accept this risk.

Apache Configuration
With the Apache software now installed, our attention is focused on configuring Apache to meet our business needs. We will be using the following configuration principle: minimize the security risk while meeting the business need of a web server for Big Brother. Apache is configured by modifying the /usr/local/apache2/httpd.conf file.

Below is a discussion of various Apache configuration options.

HostnameLookup on
This feature will cause Apache to execute a DNS query to verify the hostname of the requesting client's IP address. The benefit of this setting is the ability to parse log files for suspicious client requests for information from our Web server. The range of addresses requesting information should be very small. There is a downside since each client request will result in a longer response time because of the DNS query. Each request verifies the requesting client's information. Unless we are under attack this should not be an issue, our server will be available only on the Intranet to a small number of addresses.

User bbweb
Group bbgroup
The default user and group used to run Apache is nobody. This behavior needs to be modified since the user nobody is associated with NFS. Apache will run as a user with the least amount of privilege to perform its business function of serving Big Brother Web pages. By default Apache listens on the privileged port 80. After the port is opened by root, all child processes spawned execute as bbweb:bbgroup. Running a Web server as root would present a security risk, any exploit to the Web server could compromise the root account.

Port 80
This is the default port Apache listens on. There is no compelling reason to change the port number of Apache. A change would frustrate the user community because they would have to type in extra characters in the Web browser. We are trying to sell the monitoring system to staff, therefore we need it to be as possible to use.

ServerAdmin WebSecurity@foobar.com
The security implications of this directive are often overlooked. The first impulse during configuration is to assign the e-mail address of staff supporting the Apache application. In reality this isn't a good security practice. Although the administrator feels helpful to the user community by doing due diligence, a real e-mail address is revealing too much information. If an attacker penetrates the firewalls and finds our server a valid e-mail is handed to them. This address can be used for malicious acts such as e-mail spoofing. The user WebSecurity will be defined as a corporate mail alias. Our user community is small, all the users know who the administrator of this application is.

ServerRoot "/usr/local/apache2"
This is the Apache root directory where log files, configuration files, binaries, and Apache libraries reside. This will not be modified, however it is very important to know where files reside.

ServerName http://kazoo.foobar.com
This directive sets the server name for self-referential URL's, keeping the naming consistent throughout the Web site. This name should match the DNS name of your underlying server; in this case the server name is kazoo.

UseCanonicalName On
The UseCanonicalName directive and the ServerName directive discussed above work together. When UseCanonicalName is active, the server name returned to client is the name listed in the ServerName directive. It is not the hostname and port supplied by the client. This will keep naming consistent throughout the Web site.

LogLevel debug
This directive controls the level of Apache logging and is similar in ilk to syslogd configuration. We have selected a highly verbose level of logging; this was selected for ingress and egress traffic tracking. The log files will require rotation on a regular basis to prevent the file system from filling up. A nice benefit of log rotation is that log files stay manageable in size. In theory, an attacker could overwhelm the server with activity creating a large Apache log file; this would exhaust file system space and create a DOS attack on the Apache server. However, Big Brother will be monitoring the disk space usage on the server and will catch this sort of malicious activity.

ErrorLog syslog:local7
Part of our security model is implementation of a central logging host; this directive specifies the logging facility. Local files are more susceptible to tampering than centrally logged files. In a previous step we defined our logging program to be syslog-ng with all output being logged to a central logging host. This adds another feed into syslog-ng.

CustomLog logs/access_log combined
This directive defines where to log access attempts, and what type of format the file should use. This directive says the access_log will contain both the referrer and the user_agent information. This information will be useful when we receive alerts for malicious http requests. [6]

Directory

<Directory />

[6] http://www.cgisecurity.com/lib/ryan_barnett_gcux_practical.html p 26

```
   Order allow,deny
   Deny from all
   Satisfy all
</Directory>
```

The Directory container controls access to the web server's file systems, in this case the root node. This directive protects the entire file system defined in the directive from a directory level attack by taking a default deny stance for the entire file system. This will prevent users from being able to walk the Apache document root file system. By taking a hard stance in the beginning we can relax our protection for other resources on the Web server without having to worry about exposing our server to exploitation. The **Order** directive controls the way **Allow/Deny** statements are processed. The **Order** directive also defines the default policy for connections that do not match explicit Allow/Deny statements. Our Allow/Deny directs Apache to process the Allow statements first, then the Deny statements, with the default denying requests that are not explicitly allowed.

Options FollowSymLinks SymLinksIfOwnerMatch
The Big Brother system operates with symbolic link to the HTML files. We need to allow the Web server the ability to access files via symbolic links. This is obviously a security risk since a (mis)configured (intentional or unintentional) link may expose critical system files. This option forces a tightening of permissions on critical system files. Since the web server is running as a user of least privilege, we will be protected from the bbweb user manipulating system files. In an effort to further tighten security we are using the SymLinksIfOwnerMatch directive. This directive will only open a file referenced by a symbolic link, if the owner of the symbolic link and the owner of the file match. This will prevent a file such as /etc/passwd from being displayed in the browser since bbweb shouldn't own the password file.

AllowOverride None
This directive instructs the web server how to handle any .htaccess files encountered on the server. The configuration option “None” tells the web server to ignore any .htaccess files found in sub-directories. These files are used to override the access control on directories. Allowing the web server to follow the direction of .htaccess files is a security risk opening our system to exploitation.

IncludesNoExec
The directive prevents Server Side Includes from being used in the html code of a Web page. Server Side Includes are operating system commands embedded in HTML. A smart attacker can use server Side Includes to display or download critical system files. Protecting critical system files is basic security 101.

```
<Directory "/usr/local/apache2/htdocs">
Order allow,deny
Allow from 192.168.1
Deny from all
```

This block defines access control to the document root directory. Defined here is the DHCP address space of the company, 192.168.1. This will only allow users with IP addresses from within the company VPN to access the web server, everyone else is denied.

ServerSignature Off
This directive will protect the web server by not disclosing the software we are using to the outside world. This will change the default server http response to hide the fact that we are using Apache. By using the **On** directive we will allow the web server and version to be displayed on error pages. In effect we undo our modification of the Apache header file that we did prior to compilation.

ServerTokens Prod
This is another directive dealing with hiding server information from users or attackers. We are configuring the Web server to obfuscate as much information as possible. We are displaying as little information about our infrastructure as possible. This demonstrates our defense in depth principle by adding layers of redundancy and obfuscation.

Next we will discuss Apache directives not meeting our business or security needs. These directives will be commented in the configuration file. I am leaving them in the configuration instead of removing the directives for documentation purposes and to prevent upgrades from enabling these options.

#UserDir public_html
We did not load the mod_userdir module, however this directive showed up in the configuration file. This directive is commented because we don't have the mod_userdir module loaded. It would be nice if Apache would add/remove directives as modules are added and deleted. There are a series of directives requiring comments due to restricting directory level access. The index and directory directives were not expected to be present since the controlling modules were not added in. The shaded box contains directives we commented in the configuration file.

```
UserDir public_html
DirectoryIndex index.html index.html.var
IndexOptions FancyIndexing VersionSort
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
```

```
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^
DefaultIcon /icons/unknown.gif
ReadmeName README.html
HeaderName HEADER.html
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

Check the configuration

We completed a large number of modifications to the httpd.conf file. If there are any syntax errors they need to be fixed before we can start the web server. The following command verifies the syntax of the /usr/local/apache2/conf/httpd.conf file.

```
# cd /usr/local/apache2/bin
# ./httpd -t
Syntax OK
```

File permissions

File permissions on Apache files is adjusted.

```
# cd /usr/local/apache2
# chmod 755 conf
# chmod 644 conf/httpd.conf
# cd /usr/local/apache2/bin
# chmod 511 *
# cd /usr/local/apache2/cgi-bin
# chown bbweb:bbgroup *
# chmod 750 *
```

The stock cgi files sent with Apache are removed from the system. They have nothing to do with the business purpose of this machine.

```
# cd /usr/local/apache2/cgi-bin
# /bin/rm *
```

Password protect the server

The Big Brother data displayed by Apache will be password protected and access granted on a need to know basis. All the Unix nodes in the enterprise are monitored. This information would provide an infrastructure schematic, obviously valuable information to an attacker. The attacker could easily determine what products are running where, and plan an attack strategy based on application deployment.

```
# ./htdigest -c /usr/local/apache2/etc/passwd “EM Monitoring” jhorwath
Adding password for jhorwath in realm realm.
New password:
Re-type new password:
# cat /usr/local/apache2/etc/passwd
jhorwath:EM Monitoring:dacf0b753bdb0aa4c7aa58dc82b11fab
```

When an invalid user is tried a message is sent to the /var/log/boot.log file.

```
Feb 29 19:25:22 kazoo httpd[18101]: [error] [client 192.168.1.200] Digest: user `baduser' in realm `EM Monitoring' not found: /bb/bb.html
```

Add this to the httpd.conf file:

```
#
# This should be changed to whatever you set DoumentRoot to.
#
<Directory “/usr/local/apache2/htdocs”>

#
# Password informationm for the site
#
AuthType Digest
AuthName “EM Monitoring”
AuthDigestFile /usr/local/apache2/etc/passwd
Require user jhorwath fflintstone junior rockroll
```

Cleaning up the clutter.

Prior to version 1.3 of Apache the configuration was spread across three files: httpd.conf, srm.conf and access.conf. With the release of Apache 1.3 the configuration files were merged into http.conf, but in the interest of backwards compatibility srm.conf and access.conf were preserved. These two legacy files will be removed in the interest removing any clutter to the system. It's also one more file an attacker can modify or a file we may miss during an upgrade.

Mod Security Additions

Earlier the development idea behind mod_security was discussed, adding SNORT like protection directly into Apache. Ivan Ristic allowing SNORT rules converted into a form that can be inserted into the Apache httpd.conf file met this goal. The addition of IDS rules dramatically increases the security of Apache. This was chosen to increase security on our Web server by protecting against common attacks. We may have a staff member who read about a “nifty” attack against Web servers, and try said attack against our Apache implementation.

Mod_security directives:

The mod_security web site publishes a large list of predefined rules ready for inclusion into an Apache configuration file. These rules will help protect against many common http security problems such as buffer overflows, command execution and directory traversal. A full discussion of the mod_security rule base is beyond the scope of this paper. The complete Apache configuration file is included in the appendix. For a thorough discussion of the rules please reference

http://www.modsecurity.org/documentation/snortmodsec-rules.txt

Chrooting Apache

A powerful side effect of using mode_security is the ability to easily chroot Apache. As per the mod_security reference manual, by using a static installation of Apache chrooting can be accomplished in a few minutes. This delivers all the benefits of chrooting without the tedious and time-consuming task of creating and coping system files. As mentioned at the top of section II, we will chroot system daemons when the return meets the effort. Apache seemed like a perfect candidate for a chroot jail. As luck would have it, I was the first person running Red Hat 9.0, kernel 2.4.20-28.9 attempting to chroot Apache. Over a period of several days I tried several patches supplied by Ivan Ristic to accomplish chrooting. Finally a fresh patch combined with the creating of /dev/zero in the chroot jail successfully chrooted Apache. The option of chrooting Apache exists if we choose to use it. A few steps taken prior to compilation, plus a configuration file directive yield a chroot jail for Apache. Once the environment was ready, simply adding

SecChrootDir /chroot

To httpd.conf file configures the chroot jail.
Let's verify the chroot jail:

```
# /etc/rc.d/init.d/apache start
Starting httpd:                                [  OK  ]

# lsof -d rtd | grep http
httpd    29757   root  rtd    DIR   3,10 4096 50559 /chroot
httpd    29760  bbweb  rtd    DIR   3,10 4096 50559 /chroot
httpd    29761  bbweb  rtd    DIR   3,10 4096 50559 /chroot
httpd    29762  bbweb  rtd    DIR   3,10 4096 50559 /chroot
httpd    29763  bbweb  rtd    DIR   3,10 4096 50559 /chroot
httpd    29764  bbweb  rtd    DIR   3,10 4096 50559 /chroot
```

We will comment the chroot directive in the httpd.conf and implement a chroot jail for BB after we get things working.

Initialization scripts

We did not install Apache by default onto the system during the system build. As a result we need to manually add Apache start and stop scripts.

```
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc0.d/S85apache
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc1.d/S85apache
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc2.d/S85apache
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc3.d/S85apache
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc4.d/S85apache
# ln -s /etc/rc.d/init.d/apache /etc/rc.d/rc5.d/S85apache
```

Test the init scripts

Stop Apache

```
# /etc/rc.d/init.d/apache stop
```

```
Shutting down http: httpd
# ps -ef | grep http[d]
```

Start Apache

```
# /etc/rc.d/init.d/apache start
Starting httpd:                                [  OK  ]
# ps -ef | grep http[d]
root     11143     1  6 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
bbweb    11146 11143  0 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
bbweb    11147 11143  0 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
bbweb    11148 11143  0 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
bbweb    11149 11143  0 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
bbweb    11150 11143  0 15:26 ?          00:00:00 /usr/local/apache2/bin/httpd
```

Apache web server stops and starts as expected. Apache is ready; we are now ready to configure Big Brother for this machine.

Big Brother Setup

Big Brother (henceforth referred to as BB) is a system-monitoring tool available in two flavors, the better-than-free version (BTF), and the professional version. As of Big Brother 1.9c there wasn't a huge difference between the professional and better-than-free versions, however there will start to be differences between the versions in future releases. Cost and functionality were the largest variables in the system monitoring decision equation. The Big Brother better-than-free version is free, fitting perfectly into our cost model. However, if Big Brother is used for monitoring revenue-generating systems, a server license is required. The server license cost $695 U.S. Dollars and was purchased from http://bb4/com/purchase.html. With the server license in hand, we focus our energies towards Big Brother.

Project Plan

BB is an extremely flexible Web based monitoring system. The tool is easy to configure and flexible enough to meet almost any business or personal need. Our project plan requires rapid deployment and immediate results. The deliverables promised to management are the following: an increase in customer satisfaction, SLA's metrics, and more efficient business systems. The initial phase of this project will deploy a basic, powerful monitoring system. During the first few weeks the Unix staff will be busy correcting problems that have festered from months and years of neglect. As Foobar staff becomes more acclimated with BB, additional functions will be added to meet current and future business needs. The initial period will allow the security and administration personnel bonding and education time. Each group can educate the other to the trials and joys of their jobs.

Overview

The BB application is extremely flexible and easy to implement. A glance at the BB home page immediately relays the infrastructure status to the user. Staff with varying technical skills can easily determine the status of the infrastructure due to the easy to understood color codes. The status of monitored entities is reflected in the color of the background:

green is good
yellow is warning
red is a major problem
purple is lost contact
blue is maintenance mode
clear is no status
Colors propagate from the child to parent page, so the background color is always a worst case report of the company.
The goal of the monitoring project is to leverage BB's power and flexibility for the monitoring of the Foobar infrastructure.

Download the software
The software can be downloaded from http://bb4.com/download.html

Configuration for compilation
First thing we need to do is unpack the tar ball.

```
# tar xvfz bb-1.9c.tar.gz
bb19c.tar
```

Now unload the Big Brother tar file.

```
# tar xvf bb19c.tar
```

The directory we are interested in is bb19c.

```
# cd bb19c.tar
```

Like most open source software, BB requires configuration prior to compilation. The configuration primes the software with any system specific idiosyncrasies. The operating system type is supplied on the configuration command line, most popular type of unices are supported. If an operating system is not supplied on the command line, a pick list is presented with the supported operating systems. We are using Red Hat, a popular version of Unix, as a result our Unix distro will be specified on the command line.

Following is a discussion regarding the options chosen during the configuration process. The responses supplied to the questions are bolded in the shaded sections.

```
# ./bbconfig redhat
README.FIRST
Compilation
---> OK, we'll try redhat...


*** WARNING: Don't run BB as root ! ***
          Executing BB as root is not recommended


Prevent the execution of BB as user 'root' (y/n) [y]: y
---> OK... BB is NOT ALLOWED to run as root
```

Discussion:

Running the Big Brother daemons as root is an obvious security risk, to mitigate risk the daemon should not be run as root. Running daemons as a non-privileged user fits into our security principle of "least privilege." Since the Big Brother daemon is engineered not to run as root, it cannot listen on a port lower than 1024, by default the tool listens on port 1984.

```
        BB will only start under a designated user id. The
       startup script will verify that the current user ID
       and the designated user ID are identical.
       Note: This check is only performed during the startup script.
            It does not prevent the execution of other BB binaries/scripts
            while working using another user ID. It only prevents
            you from starting BB while working another user ID.

What will be the user ID of BB [bb]: bbuser
---> BB will only run from user 'bbuser'
```

Discussion:

In this step we selected a non-privileged user bbuser to run the BB application. Applying the principle of least privilege, this account will have no special authority on the system. The account was created earlier, it cannot login to the system and is used only to execute and configure the monitoring tool.

```
        Making sure BBHOME </usr/src/practical/bigbrother/bb19c> is writable...

---> OK, /usr/src/practical/bigbrother/bb19c is fine...


       Do you want to preserve the old style directory structure ?
       You may want to do so if you use BB extensions or
       externals that do not understand the new directory structure.
       This option is *NOT* recommend as keeping the old directory
       around represents a security risk.

Old-style directory structure (y/n): [n] n
```

Discussion:

This is a legacy question. Originally the BB Web files were not separated from the BB application. Storing data, configuration and executable files in same directory tree is a security risk. Being near to the htdocument root tree exposed non-data files to exploitation.

```
        When you set up your machines, you should use Fully Qualified
       Domain names, this means you use the whole name, like www.bb4.com,
       instead of just 'www'.  This is recommended.

Use FQDN (y/n): [y] n
---> Good, we'll use FQDN
```

Discussion:

Using the fully qualified domain name is recommended if BB is monitoring multiple domains. In this case there exists the possibility of having the same machine name in different domains. If you are going to monitor multiple domains you will need to set the

FQDN option to Y (TRUE). Once you choose to either use or not use the FQDN, a large effort is required to switch the other method. Thoughtful planning must be used when configuring this option. Foobar will never have multiple domains within the company. The non-qualified machine name will be used on the display, creating a display less busy with shorter machines names.

```
      Big Brother creates HTML pages with the status of your network.
      You'll need a web server to publish this information.

What machine will be the BBDISPLAY [localhost.localdomain]: kazoo
---> OK... kazoo will be a BBDISPLAY
```

Discussion:

This step defines the BBDISPLAY node. This is a mission critical piece of the BB application. This node is responsible for displaying BB Web pages, receiving status messages from clients, and creating the HTML pages displayed in the Web browser. The BBDISPLAY node listens on port 1984 for incoming status reports from BB clients. This host runs the bbd program listening on port 1984 for incoming status messages. The BBDISPLAY host receives incoming status messages from clients and converts said status messages to html. Two shell scripts are responsible for generating the two main BB status pages. These shell scripts are bb.sh and bb2.sh. The BBDISPLAY node must be running an http server in order to display BB Web pages. BB allows the flexibility of having redundant display hosts. In the event the main BBDISPLAY host is unavailable, the secondary host will receive status requests and generate HTML pages. Although having a backup display host is a solid business continuity plan, this is not feasible given our resource and budget situation. Foobar is in the midst of a server consolidation initiative, purchasing three servers for the monitoring initiative was all Foobar could budget for. The disaster recovery site will function as the hot backup for production and could server as a redundant BBDISPLAY host. If the disaster recovery site were mobilized, production would be in the midst of severe problems and most likely couldn't contact another site due to fire, implosion, electrical outage, etc. Initially our site will have only a single BBDISPLAY node.

```
      Big Brother sends important messages to a pager server.  This
      machine will at a minimum to be able to send mail.

What machine will be the BBPAGER [kazoo]: kazoo
---> OK... kazoo will be a BBPAGER
```

Discussion:

This step defines the BBPAGER node. The BBPAGER node functions as the notification server, another mission critical piece of our BB configuration. This is the host responsible for notifying staff (a.k.a. paging, e-mail notification) when there is a problem detected by BB. The BBPAGER host receives alerts and processes said alerts against notification rules and takes actions based on the notification rules. When BB is notified of an infrastructure problem, staff requires notification of the event to triage the issue. Engineering Big Brother with multiple BBPAGER nodes would result a page sent from every active BBPAGER node. Defining two BBPAGER nodes would send two notifications to staff for every event. Administration staff would not be happy receiving

multiple pages for each event. Our monitoring node will function as the both the BBDISPLAY and BBPAGER server. The BBPAGER and BBDISPLAY functions can reside on any host; however, it is common practice is to use the same host for both functions. Our notification vehicle will be sendmail. Although sendmail works well, the optimal solution is an analog phone as either the primary or backup notification method. Using an analog phone line would increase success rate of notification bypassing the reliance on the network. Foobar has a security policy forbidding analog phone lines attached to computers, plus the added expense of an additional phone line does not fit into our budget. Our paging solution is dependant upon network connectivity and sendmail robustness, if the network is disabled our monitoring node is rendered useless.

```
      Some questions regarding the current host
    (localhost.localdomain) will be asked.


Is this host a BBDISPLAY host (y/n): [y] y
```

Discussion:
This question asks if this node is a BBDISPLAY host. Refer two questions prior for a discussion on the BBDISPLAY server.

```
Is this host a BBPAGER host (y/n): [y] y
```

Discussion:
This question asks if this node is a BBPAGER host. Refer two questions prior for a discussion on the BBPAGER server.

```
Enter the default recipient: [root@localhost] UnixOnCall@foobar.com
```

Discussion:
This e-mail address is used a last ditch contact point. This e-mail address is used to contact a staff member through the BB Web page for help. Although this address is not viewable from Web pages, an alias is created to keep the address in sync with on-call rotations. There are no real security worries with this address because it is hidden well.

```
      Since Big Brother produces results to be displayed on web
    pages, we need to know where to view these results.


Enter the base URL for BB [/bb]: /bb
---> OK... Big Brother will live under http://kazoo.Foobar.com/bb
```

Discussion:
The step above configures the BBWEB variable that is responsible for creating Big Brother html pages. This environment variable is used by code responsible for the creation of HTML pages. Unlike other BB environment variables, the BBWEB variable is not configurable. In version 1.9c, the cgi-bin is hard coded into all C code and shell scripts. Future releases will allow users the ability to configure the BBWEB variable.

Since the html files are always written into the same place I was not able to leverage the chrooting of Apache, this would have improved the security of our system. The BBWEB variable always points relative to the BBHOME directory, so the user needs to create a symbolic links between the BBWEB directory and the document root directory. The way the symbolic link acts, it was not feasible to chroot the Apache server. Using a symbolic link to another directory defeats the purpose of the chroot. The effort to change the source code to allow html creation in a different directory was not worth the effort. Future upgrades would be a nightmare since every upgrade would require a non-supported code change. After all the time and effort needed to chroot Apache, I was really disappointed I couldn't use it in the end. Once I have the ability to use the chrooted Apache server with BB, I will.

```
      Big Brother also uses CGI scripts to create dynamic output.
     What directory do these scripts live in?

Enter CGI directory [/home/www/httpd/cgi-bin]: /usr/local/apache2/cgi-bin
---> OK... CGI scripts will live at /usr/local/apache2/cgi-bin
Enter the base URL of the CGI scripts [/cgi-bin]: /cgi-bin
---> OK... The base URL location of CGI scripts is in /cgi-bin
```

Discussion:
This step defined the location of the cgi-bin, nothing special here.

```
--> UPDATING Makefile
--> UPDATING runbb.sh
--> UPDATING bbsys.local
--> UPDATING bbsys.local for RedHat/SuSE requiring -w3 option in PINGPAR1
--> CHECKING COMMAND PATHNAMES
*** Verifying pathnames to necessary commands...
*** All pathnames OK.
--> UPDATING bbdef.sh
--> UPDATING URL location
--> INSTALLING CGI scripts

      BB needs to set the group name of the www/rep directory
      to the group name of the web server by using its user name

Enter web server user id [nobody]: bbgroup
```

Discussion:
During Apache configuration we assigned the user bbweb, group bbgroup to the Apache application. This choice was made to avoid root running our Web server, this fits our principle of least privilege. Having root run the Web server is exposing the server to an increased security risk. There are only a small number of directories the Web server needs read or write access. The group selected here will connect the Web server to the BB server allowing the two applications to share files.

```
      You may override the group name determined by the previous step.
```

```
Enter group name [users]: bbgroup
```

Discussion:
See previous step discussion.

```
--> SETTING WRITE PERMISSION FOR OWNER AND GROUP FOR www/rep
--> CHANGING THE GROUP ID OF www/rep--> UPDATING pager scripts

------------------------------------------------------

--> Done.  Now do
            cd ../src
            make
            make install
            cd ../..
            chown -R bbuser bbvar bb<ver>
                where bb<ver> is the new version's directory name
            su - bbuser
                to continue installation using that user ID

Note to RedHat 6.1 users, ping is broken, please use ping from
http://bb4.com/netkit-base-0.10-29.i386.rpm.gz
```

Discussion:
The default connectivity test for BB is via ICMP. Red Hat version 6.1 contains a ping bug right out of the box requiring a patch. This warning will not affect our configuration since we are running the latest, patched version of Red Hat, and won't be using ping in our configuration. In fact, ICMP traffic will not be allowed into or out of our server, this will be discussed later in more detail. This warning can be safely ignored.

```
# cd ../src
# make
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bb.o bb.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbsend.o bbsend.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o utils.o utils.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o bb bb.o bbsend.o utils.
o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o dohostsvc.o dohost
svc.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbd.o bbd.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbpage.o bbpage.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o bbd bbd.o dohostsvc.o
b
bpage.o bbsend.o utils.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbnet.o bbnet.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o bbnet bbnet.o utils.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o touchtime.o toucht
ime.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o touchtime touchtime.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o dumphostsvc.o
dump
hostsvc.c
```

```
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o dumphostsvc
dumphostsvc
.o dohostsvc.o utils.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o getipaddr.o getipa
ddr.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o getipaddr getipaddr.o u
tils.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbstat.o bbstat.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o bbstat bbstat.o
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH   -c -o bbrun.o bbrun.c
cc -O -DREDHAT -DGETTIMEOFDAY -DSIGSETJMP -DREGEXEC -DTIMEH -o bbrun bbrun.o utils.o
```

Discussion:

The Big Brother code was built without incident.

```
# make install
mv bb bbd bbnet touchtime dumphostsvc getipaddr bbstat bbrun ../bin
```

Discussion:

This step copies the BB binaries and shell scripts in the bin directory. The makefile also cleans up after itself removing clutter leftover from the build process.

Move code and setup the BB home directory

```
# find ./bb19c ./bbvar -print | cpio -pdvum /usr/local
# cd /usr/local
# ln -s bb19c bb
# cd /usr/local/bb/etc
```

Discussion:

At this point the system is ready for final configurations. The code was copied from the development area to the production directory. During configuration, if the BB gets mangled beyond repair, a pristine version is stored in the development directory and will be used as a fresh starting point. After the code is copied, a symbolic link is created between /usr/local/bb19c and /usr/local/bb. By referencing BBHOME, all external references point to the latest version of BB; BB upgrades are a matter of switching symbolic links, and external references require no configuration. The fewer the number of modifications needed during an upgrade, the less chance there is for an error. The most heavily referenced variable in BB is BBHOME; everything is relative to BBHOME. The BBHOME variable is set to /usr/local/bb.

Important Big Brother Parameters

The master configuration and environment file for BB is /usr/local/bb/etc/bbdef.sh; this file defines many of the important environmental variables for BB. BB is driven by many environment variables defined in this file. BB is designed to test many infrastructure components such as disk space, connectivity, network services, log files, etc. Although the scope of monitoring possibilities is nearly endless, our initial deliverable is a basic, robust monitoring system. Basic environment variables, values, and reasons for setting their values are discussed below. These variables are discussed in greater detail in the BB online documentation located at http://demo.bb4.com/bb/help/bb-man.html.

BBUSER=”bbuser”

Discussion:

This is the user id that runs the BB application. This variable is used to verify the BB daemons run as the non-privileged user bbuser. This is a huge security gain since a non-privileged user will be running the BB programs. Using a non-privileged account fits into our security guideline of least privilege. This variable is only used in the initialization program $BBHOME/runbb.sh.

STOPROOT="TRUE"
Discussion:
This variable is used in the BB shell script $BBHOME/runbb.sh to verify the user id 0 is not trying to start the BB daemons. This is a huge security gain since only a non-privileged user will be running the BB programs. Using a non-privileged account fits into our security guideline of least privilege.

FQDN="FALSE"
Discussion:
Host names listed on the BB Web pages can be listed in a fully qualified manner or by hostname (kazoo.foobar.com or kazoo). The recommendation in the documentation is to use the fully qualified domain name. This is useful in this case where there the possibility of having the same machine name in different domains. We have a single, tightly controlled domain, using a fully qualified domain name is not necessary. As was cautioned earlier, one should think long and hard about this option. Once it is set there is a lot of work to use the opposite method once your site is running.

DFWARN=90
DFPANIC=95
Discussion:
These two environmental variables are the default values for disk space monitoring. Disk space monitoring is a constant battle administrator's face, users need more space and administrators have no space to give. These variables will help prevent a disk exhaustion attack. If an attacker can exhaust the disk space used by an application, a DOS attack will result since the application can no longer write data. The operating system may crash if the file system experiencing space issues is root (/). DFWARN is the warning value for disk space checking; when this value is reached a yellow/warning condition is reported to BB. A yellow condition is a warning to staff; it does not generate a notification message but gives a gentle nudge that something is wrong. This is a warning to the administrators regarding disk space consumption. There is a potential problem brewing, but it is not severe enough to warrant intervention. DFPANIC is the value requiring immediate attention from staff, signaling pending disaster may be around the corner. When this value is reached a red/major condition is reported to BB. A red condition requires immediate attention to correct a serious disk problem. These are the default values and can be overridden by using the etc/bb-dftab file. We will be populating etc/bb-dftab file for our uses.

CPUWARN=500
CPUPANIC=750
Discussion:

The CPU threshold is based on the system load average returned by the uptime command. The value reported is the load average multiplied by 100. The CPU utilization will vary from machine to machine, based on business purpose and hardware capacity. In this day and age with systems containing multiple Gigahertz CPU's, a machine with a 400 for a load average is probably nothing to worry about. We have bumped these parameters up quite a bit from the default values. The CPU workload is a check for possible run-away processes and processes consuming large amounts of CPU. A user can disable the machine by running a poorly coded program or a cleverly crafted program designed to suck the life out of the CPU. This would result in a DOS attack and an irate user base. If a CPU exhaustion attack is attempted BB will catch it, and notify staff. These are the default values and are overridden by populating the $BBHOME/etc/bb-cputab file.

PROCS=""
Discussion:
This environment variable defines default processes causing a yellow or warning condition. This signals processes that are not running, but would be nice to have running. The process is not mission critical, only desirable. This is a legacy variable from the early days of BB, the functionality is now replaced by a configuration file allowing greater flexibility. An argument can be made regarding the benefits of having default processes which are monitored on every system, but the maintenance is much simpler using the external process table file. The configuration file is $BBHOME/etc/bb-proctab.

PAGEPROC="cron syslog-ng stunnel bbrun "
Discussion:
This environment variable defines default processes causing a red or major condition. If the process defined by this variable is not present in the process table, this is considered a condition require immediate staff attention. I have changed the value to syslog-ng, stunnel and bbrun as defaults. If an attacker tries to modify the system by stopping one of the above programs staff will be paged. These processes will be defined in the external process table as well; this is a depth in defense action. This is a great security and business continuity feature. This is a legacy variable from the early days of BB, the functionality is now placed in a file $BBHOME/etc/bb-proctab allowing for greater flexibility and manageability.

PAGELEVELS="red purple"
Discussion:
This is the ultimate love/hate environment variable. This variable defines the notification conditions for staff. Nothing is sure to generate morning conversation faster from an administrator than a page being received off-hours. The staff at Foobar will complain loudly when notification is not sent when there is a problem, and they complain when their pagers go off for a problem. This is definitely a love/hate relationship with staff. The default condition out of the box is "red purple." BB will report a purple condition when a test does not report back to the BB server in a certain time interval. By default this is 30 minutes and a tunable parameter we will not modify.

The argument can be made against paging for purple conditions, however if a test has not reported to the BB server in 30 minutes there is definitely something wrong. If an attacker compromised a machine BB may have been disabled as part of their attack. With a bit of education the administration staff will see the benefits of being notified when the monitoring system isn't running. Major/red conditions require immediate staff intervention and are pagable events.

CHKMSGLEN="TRUE"
Discussion:
This environment variable verifies messages files are not zero length. A common hacker technique is to remove message files from the system during system exploitation. This variable will verify monitored log files are not zero length. The BB log monitoring function works in a manner similar to logwatcher. Our log rotation scripts need to add some data to a message files after rotation to avoid pages from BB. This verification test is a nice security feature leveraged by our implementation. This is a desirable setting since we can have a 24x7 log watching facility with BB; this requires no additional overhead and provides a nice security service.

PAGEMSG="NOTICE"
Discussion:
This environment variable is used to monitor strings in the message files. This defines the default strings triggering a red/major condition in BB. The NOTICE token signals a condition reported to the system logging facility requiring immediate staff attention. If an attacker is trying to exploit a hardened system, the exploitation attempts will be logged in system log files and the central log host. BB should catch the attempts and notify staff of the activity. This is a desirable setting since we can have a 24x7 log watching facility with BB; this requires no additional overhead and provides a nice security service. This test will need to be disabled during system scans; otherwise the oncall pager will melt from all the activity. This is a legacy variable from the early days of BB, the functionality is now placed in a file $BBHOME/etc/bb-msgstab allowing for greater flexibility and manageability.

CONNTEST=FALSE
Discussion:
This variable defines the use of ICMP to test for machine availability. Most BB sites use ICMP to test for machine availability; the argument could be made this is not a true test from a user perspective. There are times when an ICMP will succeed but the machine is unusable from a user perspective; for example a process is chewing up all the system resources. The user community cannot use the system but BB is reporting the system as available. The confidence staff and management will have in BB may wane if problems are not caught by BB. Another reason we're not using ICMP tests is the network administrators prevent ICMP traffic cross networks, i.e. from production to test networks. This network policy prevents BB from using ICMP to test for machine availability where the ICMP traffic must cross networks. Additionally the BB node will be running iptables prevent ingress and egress ICMP traffic. Instead of using ICMP for

availability, we are going to test machine connectivity by testing port 22, the ssh network service. If we can't ssh to the machine, the machine can't be administered.

BBNETTHREADS=5
Discussion:
This variable defines how many concurrent bb-network.sh tests should run. We increased this variable to reduce the length of time to complete network testing. The production system will be handling the monitoring of 300 operating systems; the BB network testing program, bb-network.sh, will not be able to handle all 300 systems at once. This variable will allow BB to run five concurrent network test daemons, spreading the workload across multiple programs. If the system can't complete a network test the result will be a false positive. If the BB system is reporting false positives, staff will not have confidence in the integrity of BB tests and may start ignoring BB notifications. This would be dangerous since an attacker could attack our systems under the complacent watch of staff.

BBNETSVCS="noconn telnet ssh ftp smtp"
Discussion:
This is the environment variable defining what networks services BB will verify. Any service listed in the /etc/services file is capable of being tested by BB. A nice feature of this test is verification of a service running or not running. From a security perspective, this is helpful to verify security challenged services like telnet are not running. The network tests are performed from the BBNET machine to the client. During phase one of the project only a small number of services will be tested, later this list may expand to meet other business initiatives. The noconn directive disables BB from executing an ICMP test for system connectivity. The daemon responsible for performing network tests is $BBHOME/bin/bnet. The network tests are defined in the $BBHOME/etc/hosts file on a host-by-host basis.

BBNETTIMER1=15
BBNETTIMER2=15
BBNETTIMER3=30
Discussion:
These variables affect the BB network tests. Each network test is given three chances to succeed; if the test fails on the third attempt a notification is reported. The network test is performed from the BBNET machine which is defined in the $BBHOME/etc/bb-hosts file. The numeric value is the number of seconds a network test is allowed before a timeout stops the test. The first two attempts each are given 15 seconds, and the third test is given 30 seconds to succeed. This amounts to 60 seconds being given to a network test. The default values are too low and will result in too many false positives, especially during full backups. If the administration staff receives too many false positives they will become desensitized to network service notifications. The downside

is network sluggishness may go undetected by the tool; however the user community will be sure to complain if the network is slow.[7]

One last environmental variable is the polling period for BB defined as 300 seconds or five minutes. In between polling periods a node is vulnerable to a timing attack. If an attacker notices BB has just run they could edit/mangle a system log file and have it go undetected since BB checks every five minutes. This is a risk we have accepted earlier and there isn't an easy way around it, tripwire won't help for certain files. The environmental parameters are set to sane values meeting our business and security needs. Our monitoring goal is to catch problems before they become a major crisis.

Cleanup and Permission tightening

The permissions and cleanup on the BB home directory can now commence. There are a lot of files in the BB directories not needed, and we need to modify the file permissions. In an effort to reduce the BB footprint, removing clutter and increase security by removing files not needed. Here are the files in the BB home directory.

```
# cd /usr/local/bb
# ls -c
etc  FILES.LIST  README.CHANGES   README.SUPPORT  www     tmp
bin  LICENSE     README.INSTALL   runbb.sh        install  web
ext  README      README.SECURITY  runbb.sh.DIST   src
```

The directories src and install along with the README and DIST files are removed since the originals are preserved in the development directory.

```
# /bin/rm -r FILES.LIST LICENSE README* runbb.sh.DIST install src
# ls -C
bin  etc  ext  runbb.sh  tmp  web  www
```

Configuration Files

There are a series of configuration files located in the $BBHOME/etc directory requiring customization. These files allow administrators to customize and tune BB to better serve their customers. These configuration files will override the default values set in the bbdef.sh file chosen during the initial configuration. The configuration files allow the flexibility of having one file for the entire enterprise. These files allow administrators to centralize the BB installation, keeping tunable values for all nodes in one file, or to define node specific values. Although this may seem like a good idea, for security reasons I am against it. The files discussed below contain sensitive information. Listing all the business critical processes, or file systems in one file is too concentrated with sensitive information. If this information falls into the wrong hands a roadmap of our infrastructure is given out. The $BBHOME/etc directory is the main configuration repository for Big Brother. This section will review a few of the more important files

[7] http://demo.bb4.com/bb/help/bb-man.html This document was used as a reference for the entire BB compilation and configuration section.

and the modifications made to meet the business and security needs of Foobar. These files are discussed in great detail in the BB documentation located at http://demo.bb4.com/bb/help/bb-files.html.

bb-hosts
Discussion:
The bb-hosts file controls communications between BB servers and clients. In the BB system documentation the recommendation is made to have one master bb-hosts file distributed to every node. The recommendation is based easing host management by keeping the file the same on every node. This is one area of disagreement I have with the documentation. Every time a node is added or removed from BB, every bb-hosts file in the enterprise would require updating. That involves an awful lot of work for no benefit; in fact I see this as a security problem. The main bb-hosts file needs to be populated with every host, IP address, and service monitored. This model does not fit the defense in depth philosophy, or least privilege philosophy. The bb-hosts file on the BBDISPLAY node must contain a complete list of the Unix machines in the Foobar infrastructure. Using the central file distribution model, if a BB client machine is compromised the attacker has a network diagram of Foobar's Unix nodes. The minimum amount of information needed in the client $BBHOME/etc/bb-hosts file is the BBDISPLAY host, BBPAGER host, local hostname and IP address. There is an additional BB environmental variable defined in bb-hosts, BBNET. This is the node responsible for originating network tests. Our BBNET node will be the same as our BBDISPLAY and BBPAGER nodes. It is possible to have a separate node for BBNET tests, or to have multiple BBNET nodes. By default BB does a DNS name resolution on the hostname or FDDN provided in the bb-hosts file, if network tests are desired without consulting DNS, use the ***testip*** directive. This directive instructs the BB daemon bbnet not to use DNS for network tests, BB will use the ip address provided. Please refer to this URL http://demo.bb4.com/bb/help/bb-man.html#2.0 for an in-depth discussion on the bb-hosts file.

bb-proctab
Discussion:
This file defines the business critical processes of a machine. This configuration file allows the flexibility to be warned (a yellow condition) or paged (a red condition) when a process is or is not running (the process is missing from the process table). BB verifies the existence of a process in the process table based on parameters defined in the bb-proctab file. The system can alert staff based on whether a process is running or not running. For example, BB can page staff if the program telnetd is found in the process table. A flaw with the process monitoring system is the ability to assign processes to different groups. A nice feature would be the flexibility to assign the Oracle processes to the DBA's, the operating system processes to the administrators, etc. Processes are treated as an entity and not as an aggregate; one group is ultimately responsible for all the monitored processes on the machine. Initially only mission critical operating system processes will be monitored. BB will not monitor processes deemed minor; our monitoring will be limited to business critical processes. From a security and business continuity standpoint, the monitoring of processes is critical. System security is

dependant on system processes running and security challenged processes not running. For example, if syslog-ng is not running there is an exposure because our logging facility is not functioning. From a business perspective, if revenue generating Web servers are down, money is lost. This is a business problem. Note: BB checks the system every 300 seconds, if a process dies and restarts between polling periods, BB will not catch the stopping and starting of a daemon. Here is the format of the file:

```
NODE: minor processes : major processes
NODE can be the node name such as prodoracle or localhost.
localhost: : cron syslog-ng !jim
```

The bb-proctab file allows specific nodes to be named or the generic token localhost that matches every host. In our example the node "localhost" is monitored as follows: a major alarm is generated if cron or syslog-ng isn't running, and a major alarm is generated if the process jim is running. Being able to check if a process is running, when it shouldn't be is a nice feature. There are no minor alarms generated from this test. This will allow us to disable unsecured network protocols without having to worry about people enabling services. As I mentioned earlier we are not going to use this file for central monitoring purposes due to security concerns.

bb-dftab
Discussion:
This is the configuration file for disk space monitoring. This file defines file systems and their corresponding space thresholds, overriding the default values set during configuration. Here are the default values as defined at configuration time: 90% full generates a minor alarm, and 95% full generates a major alarm. This file strengthens security by watching for disk exhaustion attacks and helps with business continuity because there aren't disk resource issues. A simple but effective attack can be a disk exhaustion attack against either an application or the operating system. If an attacker is successful in exhausting the disk space for an application, the application may become unavailable because there is no place to write data. An attack on the root file system may cause the system to freeze and likely crash due to a lack of resources. Keeping judicious thresholds will prevent or lessen disk exhaustion attacks. The bb-dftab file allows a user to ignore disk thresholds for a file system by setting the thresholds over 100%, like the example below for /moreswap. This technique is useful for database file systems designed to run at 100%. Very large file system may be difficult to monitor because of their incredible size. A 180 GB that is 99% full actually has quite a bit of space left, however on the surface it appears ready to burst. There is a BB disk space plug-in available on http://www.deadcat.net which uses blocks or inodes for threshold values. In the case of a large file system, using blocks for threshold may make better business sense.

```
NODE:/moreswap:101:102
localhost:/:80:90
```

The bb-dftab file allows specific nodes to be named or the generic token localhost that matches every host. Using the example above, disk space is monitored as follows: a minor alarm is generated when the root file system becomes at least 80% full. When

the root file system becomes at least 90% full, a major alarm is generated. The file system /moreswap will never generate alarm; it would need to be 101% full to generate a minor alarm. This effectively removes monitoring from the file system. Monitoring disk space improves security and business continuity. As mentioned earlier we are not going to use this file for central monitoring purposes due to security concerns.

bb-cputab
Discussion:
This is the configuration file for CPU thresholds. This file allows administrators to customize CPU thresholds based on the resources and business use of a machine. During the initial configuration we defined the following values: 500 raises a minor alarm, and 750 raises a major alarm. In today's modern computing environment, hardware has become substantially more powerful, and the load average isn't always a reliable metric to monitor. For example, on an 8 way CPU machine the load average may look high, however the system is running fine. During backups the corporate TSM servers reach load averages over 5000, and they still meet their business needs. The purpose of this monitor is to catch any run-away processes, or to prevent an attacker from executing a program that does nothing more than monopolize the CPU causing a DOS attack

```
localhost : : 500 : 600
```

The bb-cputab file allows specific nodes to be named or the generic token localhost that matches every host. Using the example above, CPU processes are monitored as follows: a minor alarm is generated when the system load average exceeds 500 (5.0). When the system load average exceeds 600 (6.0), a major alarm is generated. As I mentioned earlier we are not going to use this file for central monitoring purposes due to security concerns.

bb-msgstab
Discussion:
This file contains text patterns searched for in monitored message/log files. The file is responsible for producing warning and major alerts based on text patterns found in a monitored log file. The initial text pattern will be small and will grow as staff becomes more acclimated with the files being monitored. This test is a great feature of BB; it eliminates the need for a monitoring tool such as logwatcher, and log files are guaranteed to be monitored as long as BB is running. Critical alerts are not missed since log files are monitored 24x7. Monitoring log files is great for security and business continuity, log files are the pulse of a computer system. Monitoring log files will help prevent or lessen attacks and help reduce business outages.

```
loclalhost: /var/log/messages : : WARNING : NOTICE :
kazoo: /var/log/maillog : : refused.*from ; failed connection : error ERROR : from localhost
```

The bb-msgstab file allows specific nodes to be named or the generic token localhost that matches every host. Using the example above, the message/log files are monitored as follows: a minor alarm is generated when the pattern WARNING is found in a

monitored log file. A major alarm is generated when the pattern NOTICE is found in a monitored log file. As I mentioned earlier we are not going to use this file for central monitoring purposes due to security concerns.

bbwarnsetup.cfg
Discussion:
The bbwarnsetup.cfg file is a mission critical file defining and driving the behavior of notification. This file defines notification conditions, host groupings, and staff groupings to mention only a few. The wealth of options available in this file is beyond the scope of this document; only basic options are covered to get the system up and running. Please refer to http://demo.bb4.com/bb/help/bb-man.html#4.0 for an in depth discussion on how BB notification works. Here are a few examples from our setup:

```
bbwarn: TRUE
```

This variable enables notification in BB; bbwarn defined as any string other than TRUE will disable paging. If there is a major outage affecting a large number of machines, this variable should be modified to stop notification and prevent pagers from being smashed by owners.

```
pagelevels: red purple
```

Status conditions triggering pages are defined by the variable pagelevels. This variable defines the status conditions triggering a notification message from BB to staff. In our environment pages are sent when a red (major) or purple (didn't report back in X minutes) condition is reported to the BB server. The reasoning behind the choice was discussed earlier during configuration.

```
pagerecovered: TRUE
```

This variable sends a notification when a test condition (red/purple) returns to a normal (green) state. Notifying staff when a problem is corrected is a nice feature and a great communication vehicle.

```
pagedelay: 20
```

This variable controls the paging delay of BB. If an alarm is not acknowledged or corrected by pagedelay minutes (20 minutes), a fresh notification is sent concerning the existing alarm. When staff receive a notification, the recipient should acknowledge an alarm through the BB Web page. The acknowledgement will suppress notification for a user defined time period, or until the condition is corrected. This prevents repeated notifications from repeatedly being sent to staff.

```
hg-notesprod: notesprod*
hg-notestest: notestest*
```

The hg- directive allows staff to group nodes into a meta-group. The directive allows for consolidation of hosts, improves readability of paging rules, and finally simplifies and shortens the rule section. Regular expressions are accepted in host group definitions. The above definition example defines a host group notesprod that matches the regular expression nodesprod*. For a site with ten servers named notesprod1 through

noteprod10, this simplifies things. Host groups allow paging rules to be much simpler and more readable. The second host group is all of the lotus notes test servers.

```
pg-pageunix: JimPager, FredPager, BatmanPager,  BatgirlPager
```

The pg- directive allows grouping of staff into page groups. This allows paging groups to be defined, allowing the flexibility of assigning different resources to different computing environments. This directive allows machine to staff assignment to be modified in one place.

bbwarnrules.cfg
Discussion:

The bbwarnrules.cfg file works in conjunction with bbwarnsetup.cfg to define and control the notification behavior of BB. This file defines the notification rules comprised of recipients, nodes and monitored aggregates. The full breadth of the notification rules is beyond the scope of this document; please refer to http://demo.bb4.com/bb/help/bb-man.html#4.0 for a more detailed discussion on BB notification rules. Below is a simple example of notification rules for Lotus Notes server for Foobar.

```
#########################################################################
#
# Lotus Notes Servers
# During business hours all Unix pagers are notified,
# off-hours all Unix people get e-mail once an hour.
#
# Format of the notification definition
# hosts;exhosts;services;exservices;day;time;recipients
#
#########################################################################
#
# Sunday reboots
# Exclamation point means please ignore alarms on Sundays from 05:45
# to 06:30 am
#
!hg-notesprod;;*;;0;0545-0630;*

#
# Production nodes instances 24x7 support
#
hg-notesprod;;*;;1-5;0800-1659;ext-FoobarPage-pg-page_unix
hg-notesprod;;*;;1-5;0000-0759 1700-2359;ext-FoobarPage-pg-production:60
hg-notesprod;;*;;06;*;ext-FoobarPage-pg-production:60

#
# Test Lotus Notes instances business hours support
#
hg-notestest;;*;;1-5;0800-1659;ext-FoobarPage-pg-page_unix
hg-notestest;;*;;1-5;0000-0759 1700-2359;ext-FoobarPage-pg-email_unix:240
hg-notestest;;*;;06;*;ext-FoobarPage-pg-email_unix:240
```

The syntax and functioning of bbwarnrules.cfg is similar to the Unix crontab. The grammar of the notification rules allow for day, time, monitored test, and recipients to be defined with a nice degree of granularity. The rules in the gray box above define the

support matrix for a group of Lotus Notes mail servers. Following is a brief explanation of the sample rules. During normal business hours a group defined by the token pageunix is sent notifications. This group was defined in the bbwarnsetup.cfg file. By default the alarms are resent every 20 minutes as defined by the pagedelay variable. During non-business hours the production machines notify the pg-production group, and the test machines send notifications to the group pg-email_unix. The token ext- defines a script used for notification purposes. The script FoobarPage is the script used to notify staff, it is listed in the appendix. During non-business hours the time delay between pages for test machines is four hours or 240 minutes. Test and quality control machines are classified as non-business critical by the business owners, as a result they do not receive 24x7 support. Production machines are classified as business critical and receive 24x7 staff support. The argument could be made for 24x7 support across all machines. If a test machines falls under attack off-hours, said attack could go undetected for several hours. This is an acceptable risk. On Sunday's notification is turned off from 05:45 to 06:30. The ! negates the rule.

bb-bbexttab
Discussion:
Earlier I mentioned if you can script it, you could monitor it. One of the more powerful features of BB is the ability to create an external monitor (a.k.a. script) and send the results to the BB server. Using the correct syntax new monitored tests appear on the BB Web page. The bb-bbexttab file defines the external scripts external to execute and the frequency to execute each script on the system. Our system will be running one external script, a memory checker. By default the scripts should be located in the /usr/local/bb/ext directory.
The format of the file is:

```
Node: white_space : script1 script2…..scriptN;timevalue
```

This syntax defines either the node or the generic token localhost matching every host. Each script is separated by white space. It is possible to specify the frequency of how often the script runs by using a ;X after the script name. This specifies the interval (in minutes) to the frequency of the script.

security
Discussion:
The file security enables security on the BB system. BB was not engineered to run from xinetd/inetd, therefore using tcpwrappers is not an option. In an effort to increase system security, the authors engineered a tcpwrappers like facility into BB. The security file defines which IP addresses or networks are allowed to transfer data to the BB server daemon, $BBHOME/bin/bbd. The format of the file is one IP address or network per line. To specify an entire network range a zero is used for the subnet. For example, to allow the IP range from 192.168.1.1 to 198.168.1.254, said range would be specified as 192.168.1.0. Once the security file is activated on the BBDISPLAY servers, connections from nodes are dropped unless their IP address or networks are listed in the etc/security file. When a new node is added to the security file, the BBDISPLAY

server must be stopped and started to reflect the changes. This file should only reside o on the BBDISPLAY server; client nodes do not need this file. [8]

Tighten the directory and file permissions

The default file permissions on BB aren't secure enough to meet increased security demands. All mission critical machines located in a DMZ require monitoring, file permissions need to be verified to avoid exposure. File permissions too loose will result in exposure, permissions too stringent hinder our monitoring effort by making the process too complex. The file permissions must be just right. The monitoring of DMZ machines is critical because they are a primary interface to our customer base. An outage to a mission critical machine could result in a PR nightmare along with a potential revenue loss. A significant amount of money may be lost if a hard deadline is missed because of a computing failure. BB file permissions on the client as well as the server are very important. The initial impulse is to have all the BB files owned by user bbuser and group bbgroup. This logic sounds good until the situation is examined more closely. If the bbuser account is compromised there is the possibility of an escalation attack on another machine since the bbuser will be connecting to DMZ machines with a trust relationship. Therefore, all files under the BBHOME directory (/usr/local/bb) must have strict access permissions. The ownership will be as follows: all files will be owned by root and executable and readable by the group bbgroup. These permissions allow bbuser to execute the BB software, and it allows the Apache web user bbweb to execute the files**.** Modification of the BB files will require root level access, if the files are modified somebody has root level privileges and we have a major problem if it isn't a member of the Unix staff.

File and directory permissions

As per the discussion in the previous section, all BB files will be owned by root, group bbweb. All files needing execute permission will be mode 0550, files used exclusively for reading will have permissions 0440. BB files should not require modification frequently; modification is restricted to a privileged account (root or sudo). The only vehicle an attacker has for changing BB files is exploitation of the root account. If a person has root authority, the game is over. Temporary directories will have file permissions of 2770, allowing members of the group to read and write files in the directory. Below are tables listing the file permissions used on the BB server kazoo. This is the footprint for the server, a client running BB will have fewer files since it does not directly notify people, display Web pages, etc.

Tighten ownership and file permissions:

```
# chown root:bbgroup  bb*
# chmod 550 bb*
```

[8] http://demo.bb4.com/bb/help/bb-files.html This document was used for the entire section discussing the BB etc directory.

BBOUT	**600**	**root:bbgroup**
bin	**550**	**root:bbgroup**
etc	**550**	**root:bbgroup**
ext	**550**	**root:bbgroup**
runbb.sh	**550**	**root:bbgroup**
tmp	**2770**	**root:bbgroup**
web	**550**	**root:bbgroup**
www	**2770**	**root:bbgroup**

The file BBOUT listed above contains chatter from the BB daemons and is created at runtime. Following is a complete listing of the Big Brother directory and file permissions on the server.

/usr/local/bb

-rw-------	1	bbuser	bbgroup	1023570	Mar	9	23:18	BBOUT
dr-xr-x---	2	root	bbgroup	4096	Feb	17	21:19	bin
dr-xr-x---	2	root	bbgroup	4096	Mar	2	23:12	etc
dr-xr-x---	6	root	bbgroup	4096	Feb	22	15:19	ext
-r-xr-x---	1	root	bbgroup	8025	Feb	15	16:49	runbb.sh
drwxrws---	2	root	bbgroup	4096	Mar	9	23:18	tmp
dr-xr-x---	2	root	bbgroup	4096	May	3	2002	web
drwxrws---	9	root	bbgroup	4096	Feb	15	16:30	www

/usr/local/bb/bin

-r-xr-x---	1	root	bbgroup	24453	Jan	18	20:48	bb
-r-xr-x---	1	root	bbgroup	4560	Mar	12	2002	bb-combo.sh
-r-xr-x---	1	root	bbgroup	5677	Mar	25	2002	bb-cpu.sh
-r-xr-x---	1	root	bbgroup	84487	Jan	18	20:48	bbd
-r-xr-x---	1	root	bbgroup	6365	Mar	12	2002	bb-disk.sh
-r-xr-x---	1	root	bbgroup	576	Mar	12	2002	bb-display.sh
-r-xr-x---	1	root	bbgroup	2537	Mar	12	2002	bb-local.sh
-r-xr-x---	1	root	bbgroup	5443	Mar	12	2002	bb-mailack.sh
-r-xr-x---	1	root	bbgroup	10086	Mar	12	2002	bb-msgs.sh
-r-xr-x---	1	root	bbgroup	5993	Jan	18	21:14	bbmv
-r-xr-x---	1	root	bbgroup	26646	Jan	18	20:48	bbnet
-r-xr-x---	1	root	bbgroup	21259	May	3	2002	bb-network.sh
-r-xr-x---	1	root	bbgroup	1456	Mar	12	2002	bb-ping.sh
-r-xr-x---	1	root	bbgroup	6776	Mar	12	2002	bb-procs.sh
-r-xr-x---	1	root	bbgroup	8536	Jan	18	21:14	bbprune
-r-xr-x---	1	root	bbgroup	2689	Jan	18	21:15	bbrm
-r-xr-x---	1	root	bbgroup	23423	Jan	18	20:48	bbrun
-r-xr-x---	1	root	bbgroup	15104	Jan	18	20:48	bbstat
-r-xr-x---	1	root	bbgroup	30651	Jan	18	20:48	dumphostsvc
-r-xr-x---	1	root	bbgroup	21554	Jan	18	20:48	getipaddr
-r-xr-x---	1	root	bbgroup	1842	Mar	12	2002	getipaddr.sh
-r-xr-x---	1	root	bbgroup	4133	Mar	20	2001	sendmsg
-r-xr-x---	1	root	bbgroup	14401	Jan	18	20:48	touchtime

/usr/local/bb/etc

-r--r-----	1	root	bbgroup	1082	Feb	21	7:50	bb-bbexttab
-r-xr-x---	1	root	bbgroup	6138	Mar	13	2002	bbchkcfg.sh
-r-xr-x---	1	root	bbgroup	4460	Mar	13	2002	bbchkcmds.sh
-r-xr-x---	1	root	bbgroup	4439	Mar	13	2002	bbchkhosts.sh
-r-xr-x---	1	root	bbgroup	600	Jan	18	20:48	bbchkwarnrules.sh
-r--r-----	1	root	bbgroup	1333	Jan	25	7:11	bb-cputab
-r-xr-x---	1	root	bbgroup	11894	Feb	16	15:40	bbdef.sh
-r--r-----	1	root	bbgroup	764	Mar	2	22:19	bb-dftab
-r--r-----	1	root	bbgroup	239	Feb	21	7:40	bb-fetchtab
-r--r-----	1	root	bbgroup	314	Feb	16	20:55	bb-hosts
-r-xr-x---	1	root	bbgroup	9084	May	15	2002	bbinc.sh
-r--r-----	1	root	bbgroup	1451	Feb	29	20:57	bb-msgstab
-r--r-----	1	root	bbgroup	2012	Mar	2	21:33	bb-proctab
-r--r-----	1	root	bbgroup	650	Jan	18	20:48	bbsys.local
-r-xr-x---	1	root	bbgroup	2818	Mar	2	21:59	bbsys.sh
-r--r-----	1	root	bbgroup	6629	Mar	2	21:51	bbwarnrules.cfg
-r--r-----	1	root	bbgroup	10135	Mar	2	23:12	bbwarnsetup.cfg
-r--r-----	1	root	bbgroup	409	Feb	29	20:04	security

/usr/local/bb/etc

-r-xr-x---	1	root	bbgroup	7349	Feb	21	8:45	bb-fetch.sh
-r-xr-x---	1	root	bbgroup	11381	Feb	22	6:55	bb-memory.sh
-r-xr-x---	1	root	bbgroup	4509	Mar	13	2002	failover
dr-xr-x---	2	root	bbgroup	4096	Mar	13	2002	hist
dr-xr-x---	2	root	bbgroup	4096	May	3	2002	mkbb
dr-xr-x---	2	root	bbgroup	4096	Mar	2	22:02	pg
dr-xr-x---	2	root	bbgroup	4096	Mar	13	2002	rep

CGI FILES

Several BB scripts need to be copied to the cgi-bin. The files we need to copy are:
bb-ack.sh bb-hist.sh bb-rep.sh bb-histlog.sh bb-hostsvc.sh bb-replog.sh
These files are located in the /usr/local/bb/web directory and need to be copied to the cgi-bin directory. Copy the files to the cgi-bin and modify the permissions. These programs allow alarm acknowledgment and historical reporting for BB.

```
# cd /usr/local/bb/web
# cp bb-ack.sh /usr/local/apache2/cgi-bin
# cp bb-hist.sh /usr/local/apache2/cgi-bin
# cp bb-rep.sh /usr/local/apache2/cgi-bin
# cp bb-hostsvc.sh /usr/local/apache2/cgi-bin
# cp bb-replog.sh /usr/local/apache2/cgi-bin
# cd /usr/local/apache2/cgi-bin
# chown root:bbgroup *
# chmod 550 *
```

Downloading external test from deadcat.net

The flexibility of BB allows almost anything imaginable to be monitored, the motto is, if you can script it, you can monitor it. This flexibility along with the community spirit has

produced a rich set of external monitoring scripts. The external monitoring scripts are referred to as plug-ins. The scripts added by brothers are field-tested and typically work with very little modification or effort. We are going to download a script to check memory resources on our machines. The script is bb-memory.sh, maintained by Henrik Storner. The code was downloaded from http://www.deadcat.net. This script will monitor memory usage on our machines. If a program experiences memory management issues, the code will start consuming large amounts of memory. This BB plug-in will be installed to detect memory issues. Our goal is to stop outages that are a direct result of poor memory management. It is feasible a user program may consume all the memory on the machine causing heavy paging and a possible DOS attack. This monitor will give admins peace of mind that a machine won't freeze due to malice or stupidity. Following are the steps to install the BB plug-in:

```
# tar xvfz bb-memory-3.0.tar.gz
x bb-memory-3.0
x bb-memory-3.0/bb_memory.exe, 62464 bytes, 122 media blocks.
x bb-memory-3.0/README, 1075 bytes, 3 media blocks.
x bb-memory-3.0/Changes, 434 bytes, 1 media blocks.
x bb-memory-3.0/bb-memory.sh, 11381 bytes, 23 media blocks.
x bb-memory-3.0/LICENSE.BEN, 1907 bytes, 4 media blocks.
x bb-memory-3.0/INSTALL.UNIX, 1863 bytes, 4 media blocks.
x bb-memory-3.0/source
x bb-memory-3.0/source/hpux
x bb-memory-3.0/source/hpux/Makefile, 93 bytes, 1 media blocks.
x bb-memory-3.0/source/hpux/bb-hp-memsz.c, 715 bytes, 2 media blocks.
x bb-memory-3.0/source/win32
x bb-memory-3.0/source/win32/bb_memory.c, 12480 bytes, 25 media blocks.
x bb-memory-3.0/memory.html, 6296 bytes, 13 media blocks.
x bb-memory-3.0/INSTALL.WIN, 2627 bytes, 6 media blocks.
# cp bb-memory-3.0/bb-memory.sh /usr/local/bb/ext
#  chmod 550 /usr/local/bb/ext/bb-memory.sh
# chown root:bbgroup /usr/local/bb/ext/bb-memory.sh
Add this line into /usr/local/bb/etc/bb-bbexttab
localhost :  : bb-memory.sh
```

Monitoring machines

Earlier we started to discuss the challenge of monitoring high visibility DMZ machines. DMZ machines require monitoring as much as machines inside of firewalls. The challenge is to monitor DMZ machines without exposing the infrastructure to unnecessary exposures. The firewall administrators rightfully want to limit access to the DMZ machines. The machines are on the fringes of the company; this presents a great deal of exposure if the machines are allowed unrestricted communication to the production network. Luckily I wasn't the first person with this opportunity to excel, the challenge of monitoring DMZ machines without exposing the company resources. There is a BB plug-in available on the http://www.deadcat.net site that solves this problem. The code gathers monitored data via ssh allowing us to monitor DMZ machines without sacrificing security. Download the code from http://www.deadcat.net, and unpack the tarball.

```
# tar xvfz bbfetch2.3.tar.gz
bbfetch/
```

```
bbfetch/README
bbfetch/bin/
bbfetch/bin/bb-file.sh
bbfetch/etc/
bbfetch/etc/bb-fetchtab.DIST
bbfetch/ext/
bbfetch/ext/bb-fetch.sh
bbfetch/ext/bb
bbfetch/TODO
```

The README file written by Jeroen Nijhof contains simple and easy to follow instructions. Below are the consolidated installation instructions from the README file. The README file does not describe the steps for setting up a ssh trust relationship, I will include those instructions below.

Set up the trust relationship
The goal is to allow logins from the BBDISPLAY node kazoo to DMZ nodes without password authentication. This will facilitate automated data transfer via ssh.

General a public-private key-pair
On node kazoo the BB node:

```
# su - bbuser
-bash-2.05b$ pwd
/usr/local/bb
$ mkdir .ssh
$ cd .ssh
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/usr/local/bb/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /usr/local/bb/.ssh/id_rsa.
Your public key has been saved in /usr/local/bb/.ssh/id_rsa.pub.
The key fingerprint is:
bb:54:d1:23:64:52:d2:5f:f5:34:39:51:51:55:3a:72 bbuser@kazoo
$ ls -l
total 8
-rw-------    1 bbuser   bbgroup       883 Feb 21 06:33 id_rsa
-rw-------    1 bbuser   bbgroup       222 Feb 21 06:33 id_rsa.pub
$
```

Copy the key the proper node

```
$ scp id_rsa.pub jim@192.168.1.199
$ id
uid=500(bbuser) gid=500(bbgroup) groups=500(bbgroup),4(adm)
$ scp id_rsa.pub jim@bud:/home/jim
The authenticity of host 'bud (192.168.1.199)' can't be established.
RSA key fingerprint is e3:23:77:14:f3:9c:3e:34:1b:dc:90:62:54:a1:4a:76.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'bud,192.168.1.199' (RSA) to the list of known hosts.
```

```
****************************** WARNING ***************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
******************************************************************************



jim@bud's password:
id_rsa.pub          100% |*****************************************************************************|   222       00:00

$ telnet bud
$ su – bbuser

# pwd
/usr/local/bb
# mkdir .ssh
# cd .ssh
[root@bud .ssh]# mv /home/jim/id_rsa.pub authorized_keys
[root@bud .ssh]#

Change permissions
# cd /usr/local/bb
# chown root:bbgroup .ssh
# chmod 750 .ssh
# ls -las | grep .ssh
   4 drwxr-x---    2 root     bbgroup        4096 Feb 21 07:53 .ssh

# chown root:bbgroup authorized_keys
# chmod 440 authorized_keys
# ls -l authorized_keys
-r--r-----    1 root     bbgroup         222 Feb 21 07:51 authorized_keys
#
$ cat aut*
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEArlB6YWep4XkNQV6tS7b8P3XioYXGYYSdIZ55Zfegl2aH+9uzHNk
BnQvb0mA5IlzrN8tcf6wNsJ5eN9h7+qk44ioE1BCoYay63AtR2hBPQfrkaaLzFwapohA9WRRDALpVSnWZ
sAYeT75T0KacJT0XpyOQSp0ZzU5aO+tIdh7ns8s= bbuser@kazoo
-bash-2.05b$
```

Discussion:

A trust relationship is a responsibility requiring careful handling and setup. A trust relationship is a huge responsibility to the administrators, users and company who have confidence in my abilities not to expose the company's systems and data. When dealing with security issues, I like to think of how the "black hats" would exploit the system being set up. Most people take the approach of trying to stop the black hats from exploiting a system. Remember the saying it takes a thief to catch a thief. I like to think of how a black hat would exploit what is configured, and then prevent that method from being used. By compromising the bbuser account on the server kazoo, I could attempt an escalation attack on any of the nodes sharing a trusted relationship via the bbuser account from the node kazoo. Once the bbuser account is compromised on the

node kazoo, I can connect to any of the trusted nodes, and hopefully compromise the root account (remember I am thinking/acting as a black hat now). Once connected to a trusted node I can do my best to compromise any privileged or exploitable account. Obviously this is an exposure we must avoid. Now it's time to switch back to the white hat, good guy, and security conscious administrator. On the trusted node I am going to change the permissions of the .ssh directory to owner root and group bbgroup, with a mode of 550. These permissions allow bbuser to cd into the directory and read files in the directory, but not modify or delete files in the directory. The authorized_keys file will be owned by root, group bbgroup, and have a mode of 440. The bbuser account is a member of bbgroup, so the key files can be read but not modified or removed. Security has many variables which must be considered, are business needs being met, are we leaving exploitable holes, are the systems usable, etc. Our system will meet the goal of monitoring DMZ machine without exposing the infrastructure to risk.

Test the connection

```
$ ssh bbuser@bud

****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************


****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************


$ exit
```

The connection was made without password authentication. The trust relationship worked well despite stringent permissions on directories and files. We now turn our attention to setting up BB to retrieve data via ssh.

Setup of bbfetch – Client side

With the trust relationship setup between the BB server and BB client, now we need to configure the client to use bbfetch. The client side of BB needs very little modification to get bbfetch working. Only a few files on the client side need modification to get BB data retrieval via ssh working. The program responsible for sending data to the BB server is

$BBHOME/bin/bb, this program will need to be replaced. Under normal conditions the $BBHOME/bin/bb program sends data to the BB server (BBDISPLAY and BBPAGER nodes) over TCP port 1984. The DMZ machines will collect BB data and store said data in a file on the system that is later picked up the BB server with ssh. The replacement program will package the BB data into a file that is copied by the BB server node with ssh through a trus relationship.

```
# mv bb bb.orig
# mv /home/jim/bb-file.sh .
# ln -s bb-file.sh bb
# chown root:bbgroup bb-file.sh
# chmod 550 bb-file.sh
```

The original $HOME/bin/bb program is renamed bb.oring, it is replaced with $HOME/bin/bb-file.sh. This simple shell script gathers and stores monitored data is gathered and stored on the bbclient. The replacement bb program is included in the bbfetch.tar archive we downloaded from http://www.deadcat.net. It's really easy, copy, link, set permissions, and were ready to go.

Setup of bbfetch – Server side

The server side setup of bbfetch is very straightforward. The monitored nodes using the ssh method of data collection are defined in the file ${BBHOME}/etc/bb-fetchtab on the BB server (BBDISPLAY) node kazoo. This file defines the node name, the id used to fetch the data, and the location of the BB data file on the client. The permissions will be tightened on the file bb-fetchtab since it contains sensitive information regarding trusted accounts and BB nodes. When the file is configured with the correct ownership and permissions we add this line:

```
bud:bbuser:/usr/local/bb/tmp/BBSTAT
```

Here are the steps to copy over the bb-fetchtab; this is the file that drives ssh data gathering for BB.

```
# cp etc/bb-fetchtab.DIST /usr/local/bb/etc/bb-fetchtab
# cp ext/bb-fetch.sh /usr/local/bb/ext
# cd /usr/local/bb/ext
```

Modify the $BBHOME/ext/bb-fetch.sh file:

In /usr/local/bb/ext/bb-fetch.sh change:

```
SCP=/usr/local/bin/scp
```

to

```
SCP=/usr/bin/scp
```

on Red Hat 9.0 change

```
SCP_OPT="-2 -p -q -i"
```

to

```
SCP_OPT="-p -q -i"
```

Even though we are using version 2 protocol, the -2 flag is not avail on the version of scp running on Red Hat 9.0 and needs to be changed. During our ssh configuration we locked ssh into using version 2 protocol.

Update the ownership and permissions.

```
# chown root:bbgroup bb-fetch.sh
# chmod 550 bb-fetch.sh
```

Add this line to the bb-fetchtab file:

```
bud:bbuser:/usr/local/bb/tmp/BBSTAT
```

Change the permissions so only root and members of the group bbweb can read the file, modification of the file by any user other than root is not permitted.

```
# chown root:bbgroup bb-fetchtab
# chmod 440 bb-fetchtab
```

Add the external script bb-fetch.sh to the BB external script file $BBHOME/etc/bb-bbexttab. By default this script will run every 600 seconds (five minutes).

```
kazoo: : bb-memory.sh bb-fetch.sh
```

As discussed earlier the file bb-bbexttab (BB External Table File) is meant for central distribution across all BB nodes, defining which programs run on which nodes. The first column of the bb-bbextab file defines the programs that execute on the node defined in column one of the file. This defines which programs execute on which nodes. Since this is a more sensitive issue this version of the file will not be placed on the client nodes. Most BB files allow you to define a host directive specifying which node(s) this directive is effective for; in this case the program bb-fetch.sh is effective only for the node kazoo.

Stop and Start BB:

```
# su – bbuser
# cd /usr/local/bb
# sh runbb.sh stop
# sh runbb.sh start
```

In a few minutes the new BB client will appear on the BB Web page along with a new column, **fetch**. The column fetch reports the status of the ssh data copy. This solution is very nice resulting in a win-win situation for all parties involved. The firewall administrators keep the environment locked down, while the administration staff and customers have computing resources under the watchful eye of BB.

What computing resources being monitored and why
CPU

The CPU test monitors the load average on the BB client system. The CPU value is based on the system load average returned by the uptime command. The value reported is the load average multiplied by 100. The CPU workload is an interesting entity to monitor from a security and administration standpoint. If programs are monopolizing the CPU, this behavior warrants investigation. The CPU maybe attacked through malice, poor programming, or bad judgment; in any case high CPU usage needs to be investigated. A machine executing a CPU intensive code could become unusable for it's user community. This would result in a DOS attack and an irate user base. If a CPU exhaustion attack is attempted BB will catch it, and notify staff.

Disk

As mentioned early, BB will monitor disk resources in our Unix infrastructure. The default disk thresholds are as follows: a warning is generated at 90% full, and a major alarm is generated when a disk exceeds 95% of capacity. The disk thresholds may need tuning on a server-by-server and application-by-application basis. Some database file systems run at 100% all the time, other file systems are so large that 99% still leaves several GB in the file system. In this case a disk being 99% full isn't as critical. Disk space is critical to monitor from an administrative point of view, filled file systems cause unpredictable behavior with applications. From a security perspective, if an attacker is successful in filling disk space a DOS attack may occur. This is a mission critical resource for machines such as a central logger. If an attacker spoofs an IP address and is successful in sending large volumes of fictitious data, the logging server could be disabled from unavailable disk space.

Fetch

This is the status of ssh data retrieval for a client machine. If an escalation attack is attempted by modifying permissions on the BB client's authorized_hosts file, the loose permissions will trigger an alarm. This monitored entity is very important, if we can't fetch data, we can't monitor the system.

FTP

This test verifies an ftp service is not running on the BB client. The BBNET server attempts a connection to the ftp port on the client. This test verifies the ftp service is not running. If the BBNET server is successful in connecting to the client's ftp port, an alert is sent to staff reporting an active ftp daemon. The ftp service should not being running on any machine, file transfer is done with scp. Ftp maybe the vehicle used by an attacker either to download or upload files from a system. Due to BB checking every 5 minutes, there is the possibility an attacker activates ftp, download files, and deactivates ftp before BB would detect the active ftp service. This test is redundant since the BB client verifies ftpd is not present in the process table. However, this test will catch an attacker trying to run a program that offers a service such as ftpd, but has an obfuscated name such as foo.

Memory

This BB plug-in monitors memory usage on a system. This is an important resource to watch from an administrative point of view. A program with a memory leak, or a poorly

designed program consuming large amount of memory will impact system performance. When BB determines memory consumption is at a critical level, staff is alerted to investigate the issue. The administrator now has a chance to rescue the system before the system either thrashes or crashes. From a security standpoint a user may try a DOS attack on the system by running a program designed to consume all available memory resources.

MSGS
This test works similar to the open source program logwatch. BB monitors log files for the existence of strings the administrator has deemed critical. An administrator defines text patterns that will trigger notification of staff to a potential problem. From a security standpoint this is a win-win situation. Assuming proper text patterns have been defined, we are almost guaranteed 24x7 message monitoring. If the BB client fails to report a status in 30 minutes, staff is notified. Zero length log files are reported as problems. If the BB message table file $BBHOME/etc/bb-msgstab is modified, tripwire will trigger an alert.

procs
This monitor defines the processes that should or should not be running on the system. Staff can define mission critical processes that need to be present in the process table. This check does not validate said processes, only checks for the existence of the process name in the process table. Additionally, we can check the process table to verify a process is not found in the process table. For example we can verify the process telnetd is not running on the system, if the process name is present, staff will be notified. This check is beneficial for both security and administration staff. The smaller and more controlled the processes table, the better things will run. This will also prevent developers, DBA's, and users from running processes that are not business appropriate.

smtp
This monitor verifies the SMTP (Simple Mail Transport Protocol) service is or is not running, depending on the business function of the BB client. Unless the BB client is a mail server, this service should not be active. If the BB client is a mail server, the SMTP service should be active. Running SMTP where not necessary is a security risk we need to mitigate. Validating the SMTP on mail servers is critical since most people consider e-mail the most business critical application.

ssh
This monitor verifies ssh connectivity and availability with a BB client. Earlier we discussed the cons of using ICMP to test for machine availability. From a business perspective using ssh for machine availability is more reliable. If we can't ssh to a machine, we can't administer it. It is important to note this test will generate messages similar to the one in the gray box below. Speaking with developers of ssh, BB opens port 22 to check for connectivity, but since the ssh connection is never accessed a log entry is made for a half connection. This is normal traffic from BBNET testing ssh. Although this test is used to determine machine availability, it is not a true test of

availability since the application hosted on the machine may not be functioning. The user community would argue the machine was not available for business.

```
Apr  5 10:56:40 myhokeybox sshd[29326]: Bad protocol version identification 'Big-Brother-Monitor-1.9c' from ::ffff:192.168.1.200
```

telnet

This test verifies the clear text; unsecured protocol of telnet is disabled. The BBNET server attempts to open a socket connection to port 23 on the client machine. As stated in the opening security initiatives of this server, clear text protocols such as telnet are not allowed to be active. If telnet is found running on a machine, a major alarm is generated alerting staff to the security breach. Having telnet enabled is a security risk and must be disabled immediately to minimize machine exposure.

Additional Security Measures For Big Brother

Chrooting Big Brother:

The initial system design used chrooting for the two major applications on the machine, Apache and BB. The amount of work required to chroot BB, for the benefit received, did not make business sense. The chroot jail for BB would be overly complex due to the amount of system resources BB checks. Resources such as system processes, log files, memory, CPU and disk space would require a great deal of effort to implement in a chroot jail. Although possible, the benefit resulting from the effort does not justify the means. Future upgrades would increase in complexity because of the customization required to accommodate chrooting BB. This idea was abandoned in lieu of using other resources such as iptables to bolster security.

Chrooting Apache Server

Here was a place I thought a chrooting Apache would work well. As discussed earlier Apache was successfully chrooted, however the method could not be used in conjunction with this version of BB. As mentioned earlier, chrooting Apache was pretty easy thanks to Ivan Ristic's mod_security module. I tried to chroot Apache in the BB environment, it just didn't work well because BB 19c hardcode the html and web paths into the source code. Although I was competent enough to change the source code, future upgrade paths are more difficult because of the code modifications. Retrofitting code modifications into future releases is ripe for error and may be impossible if there is engineering changes. The BB developers are planning to make the html and web variables configurable in future releases of BB, once that is done chrooting Apache is recommended.

Using Stunnel with BB

I had investigated using BB over stunnel to encrypt data transmissions. On my small lab network this worked well, however implementing BB with stunnel on an enterprise-wide basis was a different situation. I was concerned about the resources required to decrypt data transmissions on the BBDISPLAY server. The BBDISPLAY server is a decent machine, but I am concerned about the consequences of having a couple hundred encrypted connections arriving at the same time. Using the BB/stunnel combination

was successful in a lab environment, but is not really justifiable in a large enterprise deployment.

Maintenance Program

BB does an excellent job of detecting problems in the enterprise; sometimes BB does this job too well. One issue not addressed by BB is alarm suppression during a maintenance window, or alarm suppression when a machine is off-line for work. Maintenance windows are known in advance since prior approval is needed from the Change Management Board. Brother Tom Schmidt wrote a BB maintenance program to address notification during maintenance windows. This cgi program allows administrators to disable notification during maintenance windows or extended outages. However there is a potential security problem with the maintenance program. If the maintenance program is compromised, alarm notification can be disabled and the infrastructure can be attacked. Conversely without the maintenance feature there really isn't an easy way to place machines or monitored entities into maintenance mode. Staff may become enraged if pages are received while a problem is being addressed, or during a maintenance window. Running a secured Apache server with mod_security, the maint.pl program was identified as a potential security problem since it writes files in the /usr/local/bb/bbvar/disabled directory. Although the target directory where files are written is outside the docroot directory, allowing a cgi program to write or create files is an exposure we can't risk. An attacker could replace the password file with a trojaned version and gain root access to the system. I really had struggled with whether this utility should or should not be used. This program has a lot of potential and performs a valuable service; on a small trial basis the administration staff loved it. An additional feature is the ability to schedule maintenance windows in BB with the at command. Since the cron facility is secured on this server, that feature wouldn't do a lot of good without relaxing the restrictions on the cron facility. In the end I decided not to use the maint.pl program for BB because of the potential security exploits. The staff really liked the BB maintenance program and was not too happy with the decision; however this node needs to be protected against security exploits. In the future this program may be allowed to help meet the business needs of the company. Maybe shared memory could be used to enable and disable machines. I don't have any suggestions on how to maintain security while allowing a maintenance program. I don't want to appear overly negative of the utility, I think the concept and functionality are great, but I have to put security before personal feelings. I have left the instructions to include maint.pl onto a system in case anyone is interested.

First download and install the CGI.pm module.

Download the module from http://www.perl.com/CPAN/modules/by-module/CGI/
Install the modules:

```
# tar xvfz CGI.pm-3.04.tar.gz
# cd CGI.pm-3.04
# make Makefile.pl
# make
# make test
```

```
# make install
```

Download the BB maintenance script from deadcat.net. Download from http://www.deadcat.net by searching for maintenance. Install the program into your cig-gin directory, adjust the ownership, permissions, and modify the $BBHOME and $BBVAR variables in the program to match your environment.

Initialization scripts

BB does not ship with an automated init script. To meet our business need of having the monitoring system running 24x7; a start and stop script needs to be created. The actual script is located in the appendix. Please refer to the appendix for the complete BB rc script. We copy the script into the /etc/rc.d/init.d directory. Modify ownership and permissions.

```
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc2.d/S99bigbrother
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc3.d/S99bigbrother
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc4.d/S99bigbrother
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc5.d/S99bigbrother
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc0.d/K99bigbrother
# ln -s /etc/rc.d/init.d/bigbrother  /etc/rc.d/rc1.d/K99bigbrother
# chown root:root /etc/rc.d/init.d/bigbrother
# chmod 755 /etc/rc.d/init.d/bigbrother
```

Test Initialization Scripts

The initialization scripts are verified below.

Start

```
# /etc/rc.d/init.d/bigbrother start
Starting Big Brother:
Starting the Big Brother System & Network monitor
      Starting Big Brother Daemon (bbd)...
      Starting Network tests (bb-network)...
      Starting Display process (bb-display)...
      Starting external scripts
            Starting external script bb-memory.sh
            Starting external script bb-fetch.sh
      Starting Local tests (bb-local)...
Big Brother 1.9c started
# ps -fu bbuser
UID        PID  PPID  C STIME TTY          TIME CMD
bbuser   15188     1  0 15:39 ?        00:00:00 /usr/local/bb19c/bin/bbd
bbuser   15192     1  0 15:39 pts/2    00:00:00 /bin/sh /usr/local/bb/runbb.sh start
bbuser   15193 15192  0 15:39 pts/2    00:00:00 sleep 30
bbuser   15194     1  0 15:39 pts/2    00:00:00 /bin/sh /usr/local/bb/runbb.sh start
bbuser   15195 15194  0 15:39 pts/2    00:00:00 sleep 90
bbuser   15218     1  0 15:39 pts/2    00:00:00 /bin/sh /usr/local/bb/runbb.sh start
bbuser   15221     1  0 15:39 pts/2    00:00:00 /bin/sh /usr/local/bb/runbb.sh start
bbuser   15224 15218  0 15:39 pts/2    00:00:00 /usr/local/bb19c/bin/bbrun -a /usr/local/bb19c/ext/bb-
memory.sh
bbuser   15225     1  0 15:39 pts/2    00:00:00 /bin/sh /usr/local/bb/runbb.sh start
bbuser   15226 15225  0 15:39 pts/2    00:00:00 /usr/local/bb19c/bin/bbrun -a /usr/local/bb19c/bin/bb-
local.sh
```

```
bbuser   16071 15221  0 15:40 pts/2    00:00:00 /usr/local/bb19c/bin/bbrun -a /usr/local/bb19c/ext/bb-fetch.sh
```

Stopping

```
# /etc/rc.d/init.d/bigbrother stop
Stopping Big Brother...
# ps -fu bbuser
UID        PID  PPID  C STIME TTY          TIME CMD
```

IBM AIX Installation

A large number of our client machines are IBM AIX Unix machines. This platform uses a facility called Source Master to start and stop applications at boot time. I have included a short script to install BB on an AIX system; this will configure the system to start/stop BB via source master commands. AIX does not use the rc style init structure.

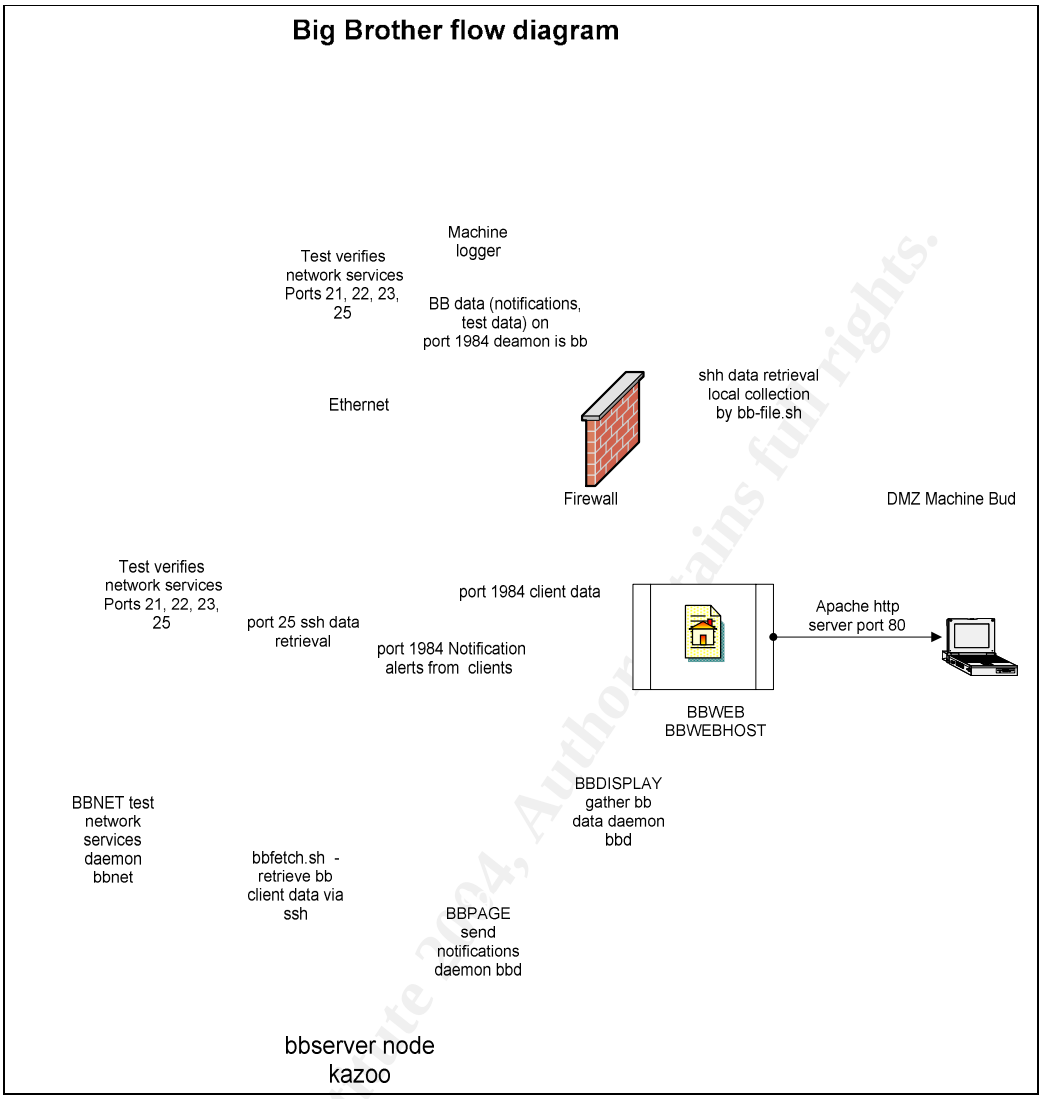
```
echo “Making the Big Brother group…”
mkgroup -A id=105 bbuser
echo "Making user..."
mkuser id=105 pgrp=bbuser admgroups=system gecos="Big Brother Account" account_locked=false
login=false rlogin=false bbuser
echo "Making inittab entry..."
mkitab "bigbrother:2:once:/usr/bin/startsrc -sbigbrother"
echo "Adding to source master...."
mkssys -u 105 -G bigbrother -s bigbrother -S -n 15 -f 9 -p /usr/local/bb/runbb.sh -a "start" -i
/usr/local/bb/BBOUT -o /usr/local/bb/BBOUT -e /usr/local/bb/BBOUT

To start BB
# startsrc –s bigbrother
To stop BB
# stopsrc –s bigbrother
```

Client Deployments:

BB Clients will be deployed via RPM based on the setup used for the BB server node. The client deployment will not be discussed here.

General Discussion:

On the BB mailing list there are users who have reported troubles using Red Hat 9.0 with BB. Some users have experienced severe memory leaks requiring reboots after a couple of weeks of operation. One suggested solution is compilation of a new kernel preferably to 2.6. Our system has not experienced any memory problems; the system has been running for 2 months without any noticeable memory problems. Big Brother has a very large scope; many features and options have not been discussed. This section concentrated on configuration of a basic, yet powerful monitoring server. Please refer to the online document available at http://www.bb4.com for a more in-depth discussion of BB and it’s wide-range of features.



Big Brother Flow Diagram

Log File Management

Several of the system daemons create and populate log files on the system. These log files are very important because they reflect the current and past state of our system. These files require maintenance; otherwise the files will grow to an unmanageable size and will consume large amounts of disk space. All of the work we did to secure our system will be compromised because the logging facilities will be disabled. Red Hat Linux is shipped with a native log management program, logrotate. This program will maintain log files defined and configured in the file in /etc/logrotate.d directory. System and Apache log files will be maintained by these maintenance scripts. As discussed earlier BB will monitor log files and notify staff when zero length log files are found. When files are rotated the date is echoed into the new log file, this is done to prevent BB from finding a zero length log file and notifying staff at midnight. The system will preserve ten generations of log files, the files are rotated on a nightly basis, after ten

days the files are no longer on the system. However, the files will be available via backup for seven years to meet company regulations.

Here is an example entry in the file /etc/logrotate.d/syslog

```
chown root:root /var/log/messages.[0]
echo "Log rotated at `/bin/date`" > /var/log/messages
/bin/kill –HUP `cat /var/run/syslog-ng.pid 2> /dev/null` 2>/dev/null || true
```

Iptables

One of our security initiatives was configuration of a host-based firewall on every node. On Red Hat Linux systems iptables is the host based firewall system built into the kernel. This packet-filtering program will be our first layer of defense against network attacks. Iptables will protect our system from harmful ingress network traffic, and prevent potential harmful egress traffic from leaving our system. Although there are many clear explanations concerning iptables on the Internet, I found the description by Michael D. Baur in Building Secure Servers with LINUX to be exceptional. The iptables configuration implemented in this document was based on the configuration Michael presented in the book. Iptables is a powerful, flexible, packet-filtering program allowing administrators to control network access to and from a machine. A good working knowledge of TCP/IP is required to be able to harness the power of this excellent, often overlooked tool. Iptables will protect the machine from IP attacks, as well as protecting other machines from an attack staged from our machine. Michael suggests administrators create a table defining which network services are allowed into and out of a node. This small but powerful table will help administrators better understand their systems. Additionally, this table will be the framework used to build the iptables rule base.

Following is a high-level view of network services allowed into and out of the machine.

Routing/Forwarding	none
Inbound service	HTTP, SSH, BB, TSM
Outbound services	DNS, NTP, Syslog-ng/stunnel, telnet, ssh, ftp, smtp

Note: telnet, ssh, ftp, and smtp are allowed outbound for testing purposes only.

Our greatest exposure to an exploit is through a network attack. Our security model was engineered around redundant layers of defenses against exploits in the hopes of reducing system exposure. The final piece is the activation of a host-based firewall, commonly known in Linux as iptables. Although iptables is being configured last, it is actually the first line of defense against network attacks. The netfilter architecture is built into the kernel in Red Hat 2.4. As mentioned earlier the stock iptables configuration file was replaced with a configuration based on the recommendations from Michael D. Bauer. The script is located on pages 74-76 of the book "Building Secure Servers with Linux." Michael's script was munged to meet our business needs. Only a small amount of modification was necessary to have the script meet our business needs. Our goal is to restrict the source, destination, and type of traffic on ingress and egress destinations, as well as logging suspicious packets to the logging facility. A

previous step configured BB to monitor system log files. BB will detect any suspicious network activity, resulting in notification to the administration staff.

Our default policy is to deny all input, output and forwarding chains. The loopback address is allowed to access the system for both ingress and egress directions. Currently the loopback address is not used for any business purpose; however we are going to permit the loopback traffic without restrictions to allow future needs such as problem determination. The rule base will attempt to block address spoofing, stealth scans, packet fragmentation attacks, and network ports not used by our system. All attack rules will log the source IP address in the log file /var/adm/kernel. This file scanned by BB and will notify staff of the attack. Input rules allow port 22 – ssh, port 80 – http, port 1984 – BB, port 4000 – TSM, everything else is logged and dropped by default. For output we allow port 21 – FTP, port 22 – ssh, port 23 – telnet, port 25 – smtp, port 53 – DNS, port 123 – NTP, port 1984 – BB, and port 4000 - TSM. For BB verification purposes we allow outbound smtp, ftp and telnet traffic. These ports are needed by BB to verify network services are or are not functioning, depending on the business need.

The iptables startup script is located in the appendix. [9]

Tripwire

The corporate standard for a system integrity checker is Tripwire. The deliverable for enterprise-wide deployment of Tripwire is currently under away, corporate policy states Tripwire must be running on all mission critical servers deployed after 01/2004. Initially our server will be running a localized copy of the freeware version of Tripwire for Red Hat. Later this year the freeware version of Tripwire for Red Hat will be replaced with the commercial version of Tripwire. The commercial version of Tripwire has the centralized management console. Luckily the freeware version of Tripwire is avail for non-commercial versions of Unix and doesn't differ significantly from the commercial version. The RPM for Tripwire was downloaded from http://www.tripwire.org/downloads/index.php. This is version 2.3.47 of Tripwire.

Install the RPM

```
# rpm –ivh tripwire-2.3-47.i386.rpm
```

Verify the installation

```
# rpm –q –a | grep tripwire
tripwire-2.3.47
```

Tripwire will be configured to monitor important operating system files, static Apache files, and static BB files.

Edit /etc/tripwire/twcfg.cfg and /etc/tripwire/twpol.cfg

[9] Michael D. Bauer, Building Secure Servers with Linux. O'Reilly & Associates, 2003 pp 68-76 The reference was used for the entire iptables section.

```
# vi /etc/tripwire/twcfg.cfg /etc/tripwire/twpol.cfg
```

Initialize the tripwire database. The default files were modified to add BB and Apache.

```
# /usr/sbin/tripwire --init
```

Use CIS to get a Security Rating

After the hardening process has been completed the CIS benchmarking tool was run to verify our system is rated high enough to be called an appliance server. Our goal for an appliance server is a CIS rating of 9.5 or better. The hardening process we followed earlier was based on the CIS standards. The tool can be downloaded from: http://www.cisecurity/org/sub_form.html.

```
# tar xvfz cis-linux.tar.gz
cis/
cis/LinuxBenchmark.pdf
cis/README
cis/CISscan-1.4.2-1.0.i386.rpm
#  cd cis
# [root@kazoo cis]# rpm -ivh CISscan-1.4.2-1.0.i386.rpm
Preparing...                ########################################### [100%]
```

Run the package to get a security rating.

```
# cd /usr/local/CIS
# ./cis-scan
# tail cis-most*
Preliminary rating given at time: Wed Feb 25 22:05:29 2004

      Preliminary rating = 9.38 / 10.00

Positive: 6.6 No non-standard world-writable files.
Positive: 6.7 No non-standard SUID/SGID programs found.
Ending run at time: Wed Feb 25 22:06:41 2004

      Final rating = 9.69 / 10.00
```

Excellent, a rating of 9.69 exceeds our goal of having a 9.50. The Negatives are the following:

```
# grep ^Negative cis-most*
Negative: 3.15 Apache web server rc-script not deactivated.
Negative: 3.15 Web server not deactivated.
Negative: 5.3 /var/log/messages should not be group-readable.
Negative: 5.3 /var/log/messages should be owned by GID 0.
Negative: 5.3 /var/log/cron should not be group-readable.
Negative: 5.3 /var/log/cron should be owned by GID 0.
Negative: 5.3 /var/log/kernel should not be group-readable.
Negative: 5.3 /var/log/kernel should be owned by GID 0.
Negative: 5.3 /var/log/secure should not be group-readable.
Negative: 5.3 /var/log/secure should be owned by GID 0.
Negative: 5.3 /var/log/boot.log should not be group-readable.
Negative: 5.3 /var/log/boot.log should be owned by GID 0.
```

```
Negative: 5.3 /var/log/maillog should not be group-readable.
Negative: 5.3 /var/log/maillog should be owned by GID 0.
```

These are all acceptable negatives. The system must run a Web server to meet the business needs of the server; therefore the items in section 3 are acceptable. Section 5 complained about log files being group readable, again this is acceptable. We are using BB to monitor local log files, the user bbuser must be able to read the log files. The files can be modified only by root account since root owns the files. The files are the group readable, bbuser is a member of the group, and therefore the log files can be monitored by bbuser/BB. This rating does not mean our system is bullet proof, it does confirm the hardening steps taken so far are correct. Personally I really like this tool, it is easy to install and use. Although the tool does not guide you through hardening process like Bastille, if you pay attention to what the tool is verifying you can learn the steps need to harden a system.

Section III: Ongoing Maintenance Procedures

Maintenance procedures discussed below are slanted more towards education, processes and procedures. At this point the server is secure and meets the needs of the business; an ongoing maintenance plan was formulated to keep the server secure while meeting future business needs. Our ongoing maintenance plan will consist of the following items:

- User Education
- Mailing Lists
- Monitoring
- Change Management
- Backups
- Patch Management
- Integrity Checks
- Vulnerability Assessment

User Education

The weakest security link in any organization is the human element; security measures are only as strong as the user. Educating the user community regarding the benefits of security is a priority. Many users view security as a process laden job impediment, one deliverable is education of the user community to the benefits of security. Technical departments will slot 15 minutes during weekly staff meetings towards security education and training. Security enlightened staff will educate the non-believers during normal working conditions and as the opportunity presents itself. Administrators will be given laminated tip sheets of commands and things to look for during daily travels on a machine. This tip sheet is similar to the tip sheet available by SANS Institute at http://www.sans.org/score/checklists/ID_Linux.pdf. Non-technical workers will also be

educated to the importance and benefits of security in the workplace. The more educated the work force, the more effective our security policies, the easier it is for implementing new or revised processes and procedures.

Mailing lists
Mailing lists are great resources to stay current with what's hot in the computing industry, one must be careful to subscribe to a small number of critical mailing lists. The volume on some mailing lists is large making it difficult to stay current with the e-mail volume.

Big Brother: bb@bb4.com
This mailing list is community of "brothers," users with practical BB experience who graciously help one another squeeze the maximum benefit from BB. The mailing list is full of extremely friendly and helpful people, including the developers Sean MacGuire and Robert Croteua. The mailing list is received in digest form to cut down on the amount of e-mail received on a daily basis.

Mod_security: mod-security-users@lists.sourceforge.net
This mailing list is a community of mod_security users who exchange information, tips and questions regarding the use and current status of mod_security. Ivan Ristic, the developer of mod_security is an active participant in the mailing list.

SSH: secureshell@securityfocus.com
The ssh mailing list allows subscribers to share ssh information with other users. The list allows users to exchange ssh questions, tips and experiences. Many interesting and powerful uses of ssh are discussed on this list.

Red Hat Watch List: redhat-watch-list@redhat.com
This list is a Red Hat driven announcement list regarding critical bug fixes and security issues found in Red Hat version 9.

Cert Advisories from SANS: **CriticalVulnerabilityAnalysis@sans.org**
This e-mail list is sent and maintained by SANS, it sends e-mail alerts to subscribers regarding security vulnerabilities found in software. Since the e-mail is delivered to us, we don't have to worry about missing an advisory because we didn't visit a website.

Cert-Advisory List: **cert-advisory@cert.org**
This e-mail list is sent and maintained by the nation Cert Coordination Center; it sends e-mail alerts to subscribers regarding security vulnerabilities found in software. Since the e-mail is delivered to us, we don't have to worry about missing an advisory because we didn't visit a website.

Monitoring
The business function of the machine is monitoring the enterprise Unix machines. By default the machine will be monitoring itself. On a 24x7 basis we will be monitoring memory, processes, disk space, CPU, log files and network services. The business of

monitoring is a constantly changing process, monitoring must change to meet current and future business needs. BB notifications are responded to in 20 minutes; although there exists no written SLA for customers, the Unix administration staff has agreed upon 20 minutes as an acceptable response time. This aligns with the pagedelay time setup in BB. During the weekly staff meeting, all BB notification alarms (those triggering a notification page) from the previous week are reviewed and discussed by the team. A monthly scorecard is delivered to business units reporting the health of machines and the service delivered by the Unix department over the past month. Potential critical problems are fixed without customers knowing a problem existed. Fixing problems before the business is impacted is great press for the Unix department. Prior to BB there was no way to demonstrate proactive action to the business because monitoring was done ad-hoc, if at all. Management is working on a formula to quantify the money saved by BB monitoring. This formula will include the cost of a full time employee, estimated expense for every minute of downtime, etc. This monitoring checks disk, CPU, network services, processes, memory and log files on a 24x7 basis.

Backups

System backups are a mission critical operation often overlooked until the occurrence of a disaster. Tivoli Storage Manager (TSM) is the enterprise backup solution for Foobar Industries. Daily incremental backups are done Sunday through Friday and a full backup is done on Saturday night. The TSM server runs a verification script daily listing all backup failures from the previous 24-hour period. There is example e-mail in the gray box below. This list is mailed to all administrators and managers responsible for systems, as well as, the group responsible for storage and backup operations. Unless there is a valid business reason, failed backups are completed during the next business day. Everything will be backed up on the system except temporary file systems, temp files and garbage files such as core dumps. Backup scheduling is handled through the TSM scheduler; nothing additional is needed on our server. On a monthly basis backups are verified by restoring random files from random servers. The group responsible for storage requests also handles backup responsibility; the Unix administration staff only uses TSM for file restoration. Because Foobar is in the insurance industry, regulations require certain backups to be retained for a period of seven years. Every six months an audit of backups is done to verify the contents and status.

```
FOOADSM MISSED AND FAILED TSM EVENTS

AIX-DOMAIN
04/08/04   21:00:00                    AIX3          AIXUNIX1        Missed
04/09/04   01:00:00                    AIX2          FOOFTPNODE      Missed

NT-DOMAIN
04/08/04   20:00:00     04/08/04   20:01:47     NT-8PM          B_1551          Failed
04/09/04   03:00:00     04/09/04   03:04:30     NT-3AM          INVBCAM1        Failed
04/08/04   22:00:00                    NT-10PM          NTNODE1      Missed
04/08/04   22:00:00                    NT-10PM          NTNODE2      Missed
04/08/04   22:00:00                    NT-10PM          NTNODE3      Missed
```

04/08/04	22:00:00	04/08/04	22:03:45	NT-10PM	NTSQLNODE1	In Progr-
04/09/04	02:00:00	04/09/04	03:01:30	NT-2AM	NTSQLNODE2	In Progr-

Change Management

Change Control is a necessary process controlling changes to the production and test environments. The change control process does not affect development systems. A change management board consisting of business owners and customers impacted by the proposed change must approve all changes impacting production systems. All changes that directly or indirectly effect a production environment must be submitted and approved via a Wednesday morning change management call. Every proposed change must have an accompanying back out plan in the event of failure. The production maintenance window for Foobar is every Sunday from 03:00 to 10:00 am. Emergency changes may be made if agreed upon by the internal business partner and mid-level management.

Patch Management

Patches are a vital ingredient for keeping systems running problem and exploit free. Patches are applied bi-monthly to the test system running on an isolated network. The system must run problem free for 3 days before a patch can be migrated to the production environment, or the internal business partner must give permission in writing to apply the new patch. Patches are applied during the scheduled Sunday maintenance from 03:00 to 10:00 am. Staff must have prior approval from the Change Management Board before patches are installed on the production environment. If a critical vulnerability is found, patches/maintenance releases may be applied immediately to production systems without prior approval of the Change Management Board. For example, if a root vulnerability for Apache is published, an emergency patch can be applied without prior approval. Administration staff, along with the security office must use good judgment and agree if the patch is needed immediately. When the kernel is updated, the NIC driver needs to be compiled into the kernel. Patches can be found at:

http://rpmfind.net/linux/redhat/updates/9/en/os/i386/

https://rhn.redhat.com/errata/rh9-errata-security.html

https://rhn.redhat.com/errata/rh9-errata-bugfixes.html

Updates will be downloaded to a corporate ftp server, transferred to a workstation, scanned with Norton for viruses, and burned onto cd-rom. To update the RPM's use the –F, freshen option.

```
Insert the cd-rom
Login as root
mount /mnt/cdrom
cd /mnt/cdrom
```

```
rpm –Fvh *.rpm
shutdown –r now
```

Integrity checks

Tripwire is the corporate standard for a system integrity checker. The enterprise-wide deployment of Tripwire is currently taking place. The free version is available only for noncommercial Unix systems, luckily Red Hat 9.0 falls into that category. Foobar has made a commitment to implement the commercial version of tripwire throughout the enterprise. Until there is an enterprise deployment of Tripwire, we will be using the free version.

Tripwire is run via cron on a daily basis at 23:45.

```
/usr/sbin/tripwire –check | mailx –s “Tripwire log for `date +%,/%d/%y`” unix_sys_requests@foobar.com
```

Vulnerability Assessment

A script has been created to save critical system information, verify the CIS Benchmark number is still acceptable, and check the system for potential exploits. This script is included in the appendix does all the above and mails the output to the system administration request mailbox. The Unix administrator is responsible for check the e-mail from the Vulnerability script. The script is run from cron, every Monday morning, below is the cron entry.

```
00 8 * * 1 /usr/local/bin/VulnerableScan.ksh >/tmp/scan.$$ 2>&1; cat /tmp/scan.$$ | mailx -s "Vulnerable Scan " unix_sys_requests@foobar.com; /bin/rm –f /tmp/scan.$$
```

Section IV: Test and Verify the Setup

4.1 Verify ssh configuration

a) Verify root access is denied

A major concern is restricting root access to the system. Direct root access is permitted only through the system console. Ssh is the accepted method of system access for administration staff; however the root user is not permitted to directly connect to the system, unless the tty is the system console.

```
# ssh root@kazoo

****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************
```

```
root@kazoo's password:
Read from remote host kazoo: Connection reset by peer
Connection to kazoo closed.
```

Examine the log files:

```
# tail secure

Feb 25 22:24:58 kazoo sshd[9762]: ROOT LOGIN REFUSED FROM 192.168.1.200
Feb 25 22:24:58 kazoo sshd[9762]: Failed password for root from 192.168.1.200 port 2566 ssh2
Feb 25 22:24:58 kazoo sshd[9762]: fatal: monitor_read: unsupported request: 24
```

Discussion:
The user should not be able to directly connect to the system with the root account. The user is not warned about trying to access the system via the root account; the connection is disconnected without a scolding.

b) Verify only protocol 2 is acceptable for connections.
During our hardening process we disabled protocol 1 access via ssh, allowing only protocol 2. Ssh protocol 1 has known exploits; as a result protocol 1 is disabled to prevent any unnecessary exposures.

```
# ssh -1 jim@kazoo
Protocol major versions differ: 1 vs. 2
# ssh -2 jim@kazoo

****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************

jim@kazoo's password:

****************************** WARNING ****************************************
You have accessed a private computer system. This system is for authorized use
only and user activities may be monitored and recorded by company personnel.
Unauthorized access to or use of this system is strictly prohibited and
constitutes a violation of federal, criminal, and civil laws. Violators may
be subject to employment termination and prosecuted to the fullest extent of
the law. By logging in you certify that you have read and understood these
terms and that you are authorized to access and use the system.
*******************************************************************************

[jim@kazoo jim]$
```

Here are the /var/log/secure entries:

```
Protocol 1 try:
Feb 28 15:54:30 kazoo sshd[26959]: Did not receive identification string from 192.168.1.202
Protocol 2 try:
Feb 28 15:54:48 kazoo sshd[26961]: Accepted password for jim from 192.168.1.202 port 44107 ssh2
```

Discussion:
Access was granted via protocol 2, but protocol 1 access was denied.

c) Logging
One of our configuration steps was logging access to the system. If forensic analysis is ever needed on the system, a record of logins is very important. Because we are logging locally and centrally we will be able to keep records of system access. We displayed log files in steps a and b.

Discussion:
In step b the /var/log/secure log file was interrogated for system access.

d) Warning Banner
A warning banner displaying the terms, conditions and consequences of system use or misuse is important to the system.

Discussion:
The warning banner was displayed in steps a and b. If there is a compromise on the system, it is important to have an acceptable use policy is in place.

4.2 Port Verification

A few times a desired port map was mentioned; here we will verify which ports are running.

Here is our desired port map for the BB server:

Service	port	direction	business purpose
ssh	22:tcp	inbound/outbound	command line access, bb data fetch
http	80:tcp	outbound	serve bb Web pages
ntp	143:udp	outbound	sync system clock with ntp servers
syslog-ng	514:tcp	outbound	transfer data to log server
bbd	1984:tcp	inbound	receive bb monitored data

Using the tools nmap and lsof we will verify which ports are running on our system.

First we verify we are only running a small number of services:

List all services running on the machine:

```
# lsof -i
```

```
COMMAND  PID   USER   FD   TYPE DEVICE SIZE NODE NAME
stunnel  655   root    4u  IPv4   1194       TCP localhost.localdomain:shell (LISTEN)
sshd     715   root    3u  IPv4   1376       TCP *:ssh (LISTEN)
ntpd     731    ntp    4u  IPv4   1427       UDP *:ntp
ntpd     731    ntp    5u  IPv4   1428       UDP localhost.localdomain:ntp
ntpd     731    ntp    6u  IPv4   1429       UDP kazoo:ntp
bbd      846 bbuser    3u  IPv4   3072       TCP *:1984 (LISTEN)
httpd   2703   root    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2706  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2707  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2708  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2709  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2710  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   2717  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
httpd   5011  bbweb    3u  IPv4 446473       TCP *:http (LISTEN)
```

This verifies we are running 5 services. Being paranoid let's verify which port each is associated with.

```
# nmap -sTUR -F -P0 -O 192.168.1.201

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Warning:  OS detection will be MUCH less reliable because we did not find at least 1 open and 1
closed TCP port
Interesting ports on  (192.168.1.201):
(The 2145 ports scanned but not shown below are in state: filtered)
Port       State       Service (RPC)
22/tcp     open        ssh
80/tcp     open        http
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SInfo(V=3.00%P=i386-redhat-linux-gnu%D=2/29%Time=4041D280%O=22%C=-1)
TSeq(Class=RI%gcd=1%SI=4ACFA2%IPID=Z%TS=100HZ)
TSeq(Class=RI%gcd=1%SI=4AD053%IPID=Z%TS=100HZ)
TSeq(Class=RI%gcd=1%SI=4AD995%IPID=Z%TS=100HZ)
T1(Resp=Y%DF=Y%W=16A0%ACK=S++%Flags=AS%Ops=MNNTNW)
T2(Resp=N)
T3(Resp=N)
T4(Resp=N)
T5(Resp=N)
T6(Resp=N)
T7(Resp=N)
PU(Resp=N)


Uptime 0.678 days (since Sat Feb 28 14:36:16 2004)

Nmap run completed -- 1 IP address (1 host up) scanned in 582 seconds
```

Discussion:

This step verifies the system has a small number of ports open on the system. The lsof program verifies there are 5 services running on the machine. The nmap program verifies there are 2 services listening, although the BB daemon was not

found during the scan. This step verifies the least privilege principle with least number of services running.

4.3 System Vulnerability Scan with Nessus

Nessus is a freely available vulnerability scanner allowing administration staff to verify the security of a system. Nessus has the potential of being very aggressive and may cause DOS attacks if the system is not configured correctly. This aggressive nature was one of the reasons the tool was chosen to test the system. If our system can survive the most aggressive attacks, then we can deliver a high degree of service to the business. Nessus was installed on a laptop, allowing staff the freedom to drop anchor anywhere on the network and test. The code was downloaded 02/28/2004, and every plug-in available at the time was enabled.

 # cat kazoo_good.txt

```
Nessus Scan Report
------------------



SUMMARY

 - Number of hosts which were alive during the test : 1
 - Number of security holes found : 0
 - Number of security warnings found : 0
 - Number of security notes found : 9



TESTED HOSTS

 192.168.1.201 (Security notes found)



DETAILS

+ 192.168.1.201 :
 . List of open ports :
   o general/tcp (Security notes found)
   o ssh (22/tcp) (Security notes found)
   o http (80/tcp) (Security notes found)
   o general/udp (Security notes found)

 . Information found on port general/tcp


    The remote host is up

 . Information found on port general/tcp
```

TCP inject NIDS evasion function is enabled. Some tests might
run slowly and you may get some false negative results.

. Information found on port general/tcp

TCP fake RST NIDS evasion function is enabled. Some tests might
run slowly and you may get some false negative results.

. Information found on port general/tcp

HTTP NIDS evasion functions are enabled.
You may get some false negative results

. Information found on port ssh (22/tcp)

An ssh server is running on this port

. Information found on port ssh (22/tcp)

Remote SSH version : SSH-2.0-OpenSSH_3.7.1p2

. Information found on port ssh (22/tcp)

The remote SSH daemon supports the following versions of the
SSH protocol :

. 1.99
. 2.0

. Information found on port http (80/tcp)

A web server is running on this port

. Information found on port general/udp

For your information, here is the traceroute to 192.168.1.201 :
192.168.1.202
192.168.1.201

```
This file was generated by the Nessus Security Scanner
```

Discussion:
The scan required nearly two hours to complete. This test verified how important iptables is to the defense of the system. There was a lot of chatter in the log files from the Nessus scan. As the Nessus documentation warned, IDS systems lit up during the Nessus scan. The system was expected to survive the test. The system survived and performed rather well. Apache was serving Web pages during the scan, BB caught the scans, iptables protected the system against packet level attacks, and mod_security protected Apache. This test really stressed the system as a whole, the individual pieces combined to make a formable defense for the system.

4.4 Apache Verification

Although the business function of this server is enterprise monitoring, the user base will be using Apache to access the data. The goal has been to provide the securest Web server possible while still meeting the needs of the business. This will require a delicate balance, if the Apache server is secure but unusable, the business will not benefit from our work.

a) Verify Apache is not running as root

```
# ps -ef | grep htt[p]
root      2703     1  0 05:24 ?        00:00:00 /usr/local/apache2/bin/httpd
bbweb     2706  2703  0 05:24 ?        00:00:00 /usr/local/apache2/bin/httpd
bbweb     2707  2703  0 05:24 ?        00:00:01 /usr/local/apache2/bin/httpd
bbweb     2708  2703  0 05:24 ?        00:00:00 /usr/local/apache2/bin/httpd
bbweb     2709  2703  0 05:24 ?        00:00:01 /usr/local/apache2/bin/httpd
bbweb     2710  2703  0 05:24 ?        00:00:01 /usr/local/apache2/bin/httpd
bbweb     2717  2703  0 05:24 ?        00:00:00 /usr/local/apache2/bin/httpd
bbweb     5011  2703  0 05:29 ?        00:00:00 /usr/local/apache2/bin/httpd
```

Discussion:
The ps command displays Apache running as the user bbweb. This verifies one of our security concerns; we do not allow apache to run as the user root. The first Apache process needs to run as root to open port 80 (since it is a port lower than 1024 and a privileged port), once the port is opened the application is spawned running Apache as the bbweb user.

b) Verify Web server does not display too much information

```
# telnet kazoo 80
Trying kazoo...
Connected to kazoo.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 404 Not Found
Date: Sun, 29 Feb 2004 12:25:05 GMT
```

```
Server: Some Server
Connection: close
Content-Type: text/html; charset=iso-8859-1

Connection closed by foreign host.
```

Discussion:
We connected to the Apache server and query for information about the server; the server reported no usable information. Hopefully this will help deter an attacker who is attempting to attack our Apache server.

c) Verify apache obfuscation
Before Apache was compiled the server information was changed to help obfuscate the identification of the server. If an attacker scans the server we do not want to give away any more information than is necessary.

```
# /usr/local/apache2/bin/httpd -v
Server version: Some Server/99.99.99
Server built:   Jan 20 2004 21:10:25
```

Discussion:
The http server name and version is not displayed when the http server is queried. Although a local account with shell access can execute a ps command and see apache2 in the process table.

d) Verify mod_security works
Ivan Ristic (developer of mod_security) has made available many testing scripts to help verify mod_security is working. The scripts are included with the source in the "tests" directory. Getting the scripts to work was very easy, instructions as per the tests/README.test file.

```
# cd /usr/local/practical/mod_security-1.7.4/tests
# cp modsec-test.pl /usr/local/apache2/cgi-bin
# chmod 550 /usr/local/apache2/cgi-bin/modsec-test.pl
# chown root:bbgroup /usr/local/apache2/cgi-bin/modsec-test.pl
```

Edit /usr/local/apache2/conf/httpd.conf and add the contents of the file httpd.conf.regression-v2 for the mod_security source directory.

```
# cd /usr/local/practical/mod_security-1.7.4/tests
# ./run-test.pl 192.168.1.201:80 *.test
```

```
Test "01 Simple keyword filter": OK
Test "02 Self referencing directories": OK
Test "03 Evasion via path traversal": OK
Test "04 Evasion via a double slash in the path": OK
Test "05 Mixed case addresses": OK
Test "06 Evasion via URL encoding": OK
Test "07 Special characters in the path": OK
```

```
Test "08 Invalid URI encoding in parameters": OK
Test "09 Directory traversal in parameters": OK
Test "10 Keyword in POST": OK
Test "11 XSS attack": OK
Test "12 HTML forbidden": OK
Test "13 SQL injection": OK
Test "14 Redirect action (requires 302)": OK
Test "15 Not an attack (requires 200)": Failed (status = 500)
Test "16 Request without Host header": OK
Test "17 Request without User-Agent header": OK
Test "18 Keyword in POST only": OK
Test "19 Keyword in POST only, negative (requires 200)": Failed (status = 500)
Test "20 Keyword in QUERY_STRING only": OK
Test "21 Keyword in QUERY_STRING only, negative (requires 200)": Failed (status = 500)
Test "22 Keyword in ARGS, method GET": OK
Test "23 Keyword in ARGS, method POST": OK
Test "24 Keyword in single variable": OK
Test "25 Keyword in single variable, negative (requires 200)": Failed (status = 500)
Test "26 Keyword variable exclusion (requires 200)": Failed (status = 500)
Test "27 Keyword variable exclusion, negative": OK
Test "28 Simple keyword inverted pattern": OK
Test "29 Filter variable names": OK
Test "30 Filter variable values": OK
Test "31 Test for the URL encoding plus bug": OK
Test "32 SQL injection 2: SELECT test": OK
Test "33 XSS attack 2": OK
Test "34 Invalid byte range in parameters": OK
Test "35 Invalid byte range in the URL": OK
Test "36 Backslash conversion test (windows only)": OK
Test "37 URL decoding bug 2": OK
Test "38 Unicode test 1": OK
Test "39 Unicode test 2": OK
Test "40 Unicode test 3": OK
Test "41 post variable parsing bug test #1 (requires 200)": Failed (status = 500)
Test "42 post variable parsing bug test #2": OK
Test "43 post range check bug": OK
Test "44 normalisation bug": OK
Test "45 null byte attack": OK
Test "47 test action "allow" (requires 200)": Failed (status = 500)
Test "48 chained rules test #1": OK
Test "49 chained rules test #2 (requires 200)": Failed (status = 500)
Test "50 chained rules test #3 (requires 200)": Failed (status = 500)
Test "51 skipnext test 1, without a parameter (requires 200)": Failed (status = 500)
Test "52 skipnext test2 , with a parameter (requires 200)": Failed (status = 500)
Test "53 named cookie test": OK
Test "54 named cookie test, positive (requires 200)": Failed (status = 500)
Test "55 cookie names test": OK
Test "56 cookie values test": OK
```

Discussion:

There are 56 tests available for mod_security in the test directory. As Ivan was developing mod_security, he cobbled together a few scripts to verify the mod_security code worked. Several tests failed on our server due to the

restrictive nature of our configuration, which is very good. Our restrictive configuration blocks tests of a certain ilk from even running.

e) Verification of password protected pages

I did not screen-print the password dialog box, since it wasn't too interesting. I have included output from Apache and system log files.

First, the system is accessed through a valid account, nothing special happens on the system.

Now I will try to access the system as an illegal user baduser. This really makes the system light up. Here is the entry in /var/log/boot.log:

```
Feb 29 19:34:42 kazoo httpd[18103]: [error] [client 192.168.1.200] Digest: user `baduser' in realm `EM Monitoring' not found: /bb/bb.html
```

Additionally a message was sent to /var/log/kernel stating a Stealth Scan attempt on port 80 may have taken place. Since this is a notice in the /var/log/kernel file, a page will be sent to staff alerting them to a potential problem. The next section discuss in greater detail Big Brother notification and alerting.

```
Feb 29 19:34:42 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34598 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:42 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34599 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:43 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34600 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:44 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34601 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:45 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34602 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:49 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
```

```
TOS=0x00 PREC=0x00 TTL=64 ID=34603 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:34:55 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34604 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:35:09 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34605 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
Feb 29 19:35:36 kazoo kernel: NOTICE: Stealth scan attempt?IN=eth0 OUT=
MAC=00:0b:db:d0:bd:f0:00:10:5a:03:20:c2:08:00 SRC=192.168.1.200 DST=192.168.1.201 LEN=52
TOS=0x00 PREC=0x00 TTL=64 ID=34606 DF PROTO=TCP SPT=2629 DPT=80 WINDOW=6822
RES=0x00 ACK
FIN URGP=0
```

Discussion:
Password verification worked well, valid users are allowed Web access, and bogus users are prevented from accessing the system. Additionally, successful and unsuccessful login attempts are stored locally and centrally. One downside is the login incorrect message will cause BB to generate a major alarm, and thus a page. This behavior will need some modification; also the administration staff will need security and monitoring education.

4.5 Big Brother Verification

a) Prevent BB from running as root

```
# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
# cd /usr/local/bb/
sh runbb.sh start

*** NOTICE: BB has been configured not to run as root ! ***
```

Discussion:
One of our guidelines while building this machine was the principle of least privilege; only processes requiring root level access will get it. BB does not require root level access. If an exploit is found in the BB daemon, we won't have to worry about the root account being exposed through the exploit.

b) Prevent unauthorized machines from sending data to BB

BB does not run via xinetd/inetd, resulting in a lack of ability to leverage tcpwrappers for added protection. The BB development team added a tcpwrappers like facility into the product. The $BBHOME/etc/security file polices access to the BB daemon. If the IP

address of a node is not in the security file, the machine cannot send data to BB. I added a bogus node with the IP address 192.168.1.202 and started BB on that node. The BB server, kazoo refused the BB connection. This illustrates the defense in depth principle. The data made it through the firewall layer, through iptables on kazoo, however BB prevented access to the system.

From the node kazoo, file /usr/local/bb/BBOUT:
Sun Feb 29 20:09:42 2004 bbd Incoming message from 3389106368 - 192.168.1.202 refused
Sun Feb 29 20:09:43 2004 bbd Incoming message from 3389106368 - 192.168.1.202 refused
Sun Feb 29 20:09:43 2004 bbd Incoming message from 3389106368 - 192.168.1.202 refused
Sun Feb 29 20:09:43 2004 bbd Incoming message from 3389106368 - 192.168.1.202 refused
Sun Feb 29 20:09:43 2004 bbd Incoming message from 3389106368 - 192.168.1.202 refused

c) Verify disk space notification

The file systems on the BB server were created very large; this was done because there was a large amount of disk space available. A side effect of this was avoidance of disk resource problems by allowing for future growth. I manufactured a disk problem by lowering the BB threshold to 5% on the / (root) file system. I did this by adding this line into /usr/local/bb/etc/bb-dftab:
/:2:5
This was done to simulate a potential DOS or resource exhaustion attack, where an attacker's target is to exhaust disk space for an application possibly crashing the application.

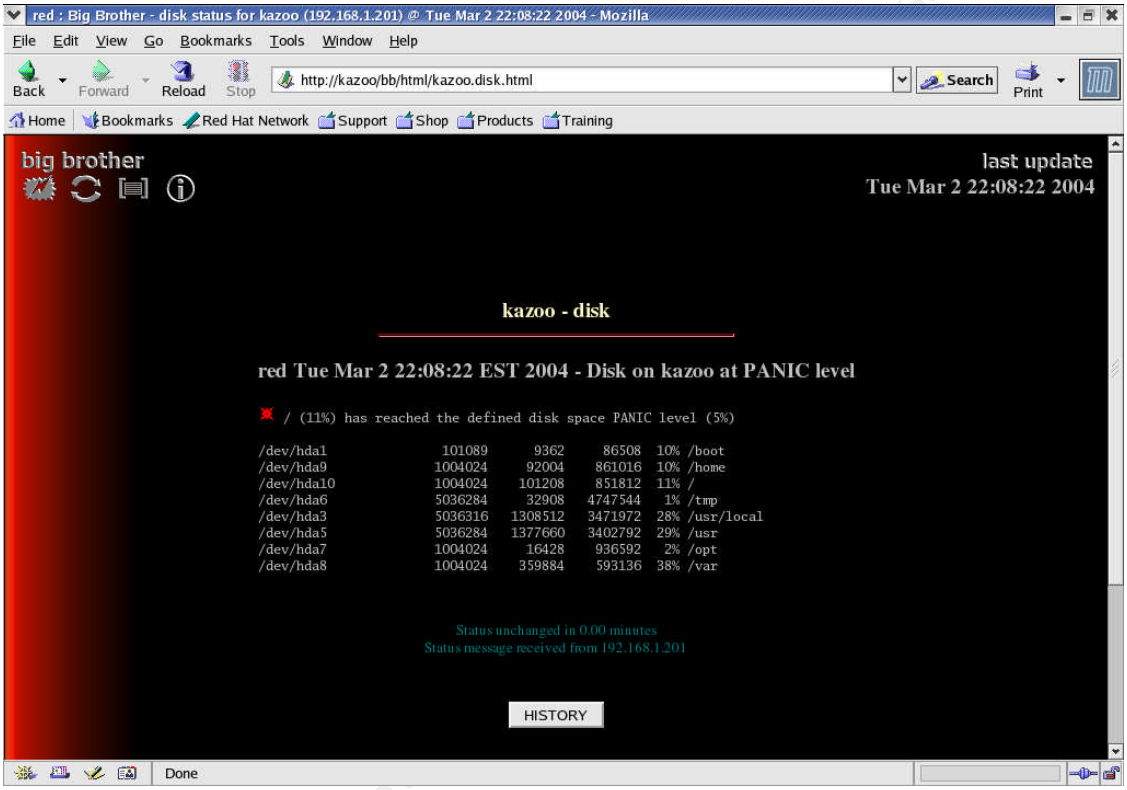
Page Notification

```
From bbuser@tFoobar.com Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@tFoobar.com
Subject: BB - 2089010  kazoo.disk Problem

[2030910 ] kazoo disk Problem 03/02/2004 22:08:22
```

E-mail notification

```
From bbuser@kazoo Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  kazoo.disk Problem

pg-page_unix [2030910 ] kazoo disk Problem [2030910]  kazoo.disk red Tue Mar  2 22:08:22 EST 2004 -
Disk on kazoo at PANIC level
&red / (11%) has reached the defined disk space PANIC level (5%)

/dev/hda1            101089     9362     86508  10% /boot
/dev/hda9           1004024    92004    861016  10% /home
/dev/hda10          1004024   101208    851812  11% /
/dev/hda6           5036284    32908   4747544   1% /tmp
/dev/hda3           5036316  1308512   3471972  28% /usr/local
```

```
/dev/hda5          5036284   1377660   3402792  29% /usr
/dev/hda7          1004024     16428    936592   2% /opt
/dev/hda8          1004024    359884    593136  38% /var

Please see: http://kazoo/bb/html/kazoo.disk.html
```



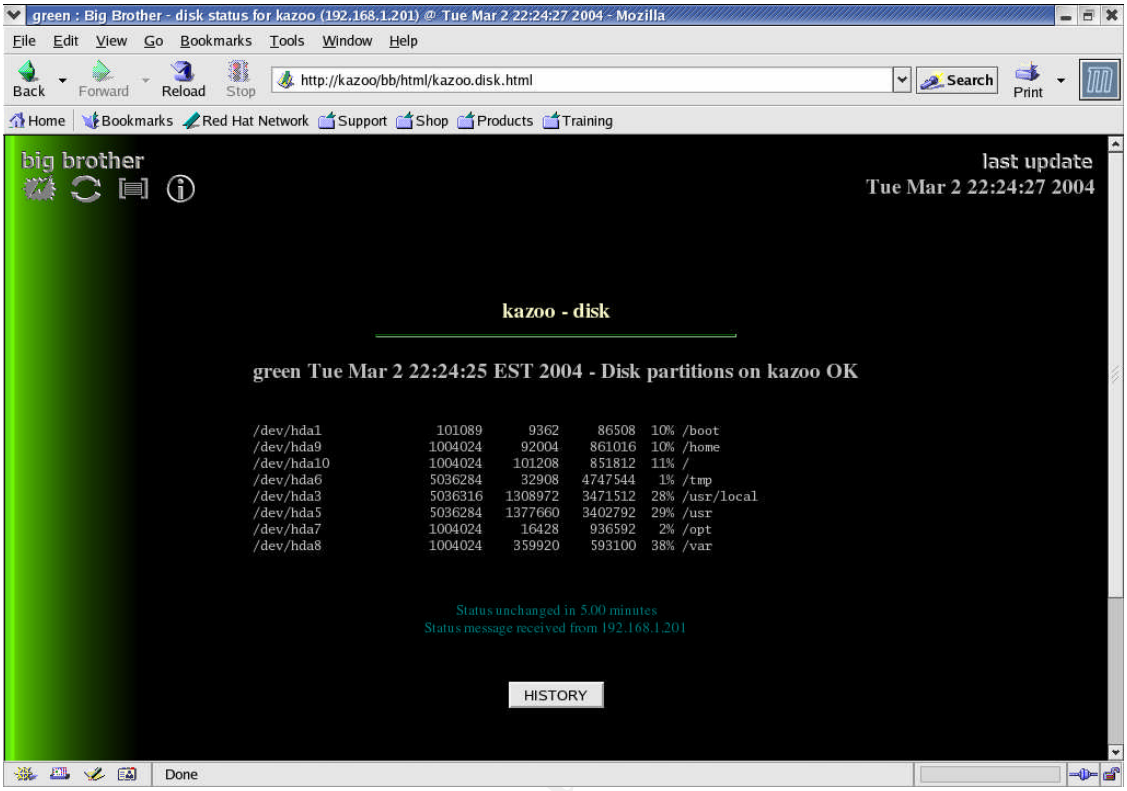
I removed the ridiculously low disk threshold and allowed BB to determine all is normal with the disk file systems on the system. A notification message is sent to staff notifying them of the change to a "green" state, and the Web page is updated with the clean status.

E-mail notification

```
From bbuser@kazoo Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  kazoo.disk Recovered

[0000000 ] kazoo disk recovered [0000000]  kazoo.disk recovered Tue Mar  2 22:19:27 2004 Problem has
been resolved after 665 seconds

Please see: http://kazoo/bb/html/kazoo.disk.html
```

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@Foobar.com
Subject: BB - 2089010  kazoo.disk Recovered

[0000000 ] kazoo disk recovered 03/02/2004 22:19:27
```



Discussion:
Disk problems are a common plaque for administrators. An attacker can attempt to crash our machine by exhausting file system space, or may try to disable an application by exhausting disk space needed by the application. Either could result in a loss of revenue and business.

d) Verify processes notification
An attacker may try to compromise our machine by stopping the logging facility, by doing this the attacker doesn't have to worry about their tracks being captured on the central logging machine. This will simulate an attack on our machine where the attacker is trying to disable a monitored daemon on the system. In this demonstration the syslog-ng daemon is stopped so the attacker can prey on our machine without being noticed.

```
# /etc/rc.d/init.d/syslog-ng stop
Stopping /usr/sbin/stunnel:                    [  OK  ]
# ps -ef | grep syslo[g]
#
```

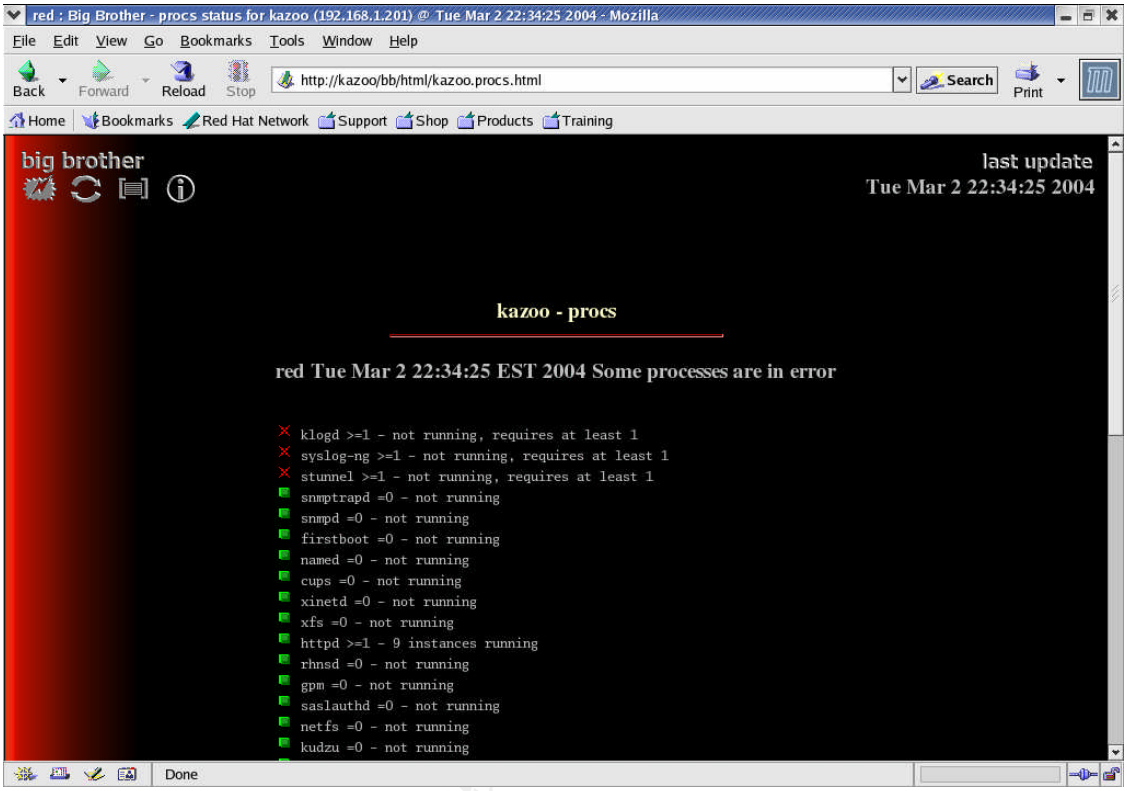
The logging facility is stopped on the machine, but not to worry BB is on the job, catching the script-kiddies in the act! BB detects a problem and sends an e-mail notification to staff. As in previous examples I have included the full notification send via e-mail, and the terse messages sent to a pager/cell phone. In addition to the notification messages, the BB Web page is updated with the critical status.

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@tFoobar.com
Subject: BB - 2089010  kazoo.procs Problem

[3210210 ] kazoo procs Problem 03/02/2004 22:34:25
```

E-mail notification

```
From bbuser@kazoo Tue Mar  2 16:53:59 2004
Date: Tue, 2 Mar 2004 16:53:58 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  kazoo.procs Problem

pg-page_unix [3210210 ] kazoo procs Problem [3210210]  kazoo.procs red Tue Mar  2 22:34:25 EST
2004
Some processes are in error

&red klogd >=1 - not running, requires at least 1
&red syslog-ng >=1 - not running, requires at least 1
&red stunnel >=1 - not running, requires at least 1
&green snmptrapd =0 - not running
&green snmpd =0 - not running
&green firstboot =0 - not running
&green named =0 - not running
&green cups =0 - not running
&green xinetd =0 - not running
&green xfs =0 - not running
&green httpd >=1 - 9 instances running
&green rhnsd =0 - not running
&green gpm =0 - not running
&green saslauthd =0 - not running
&green netfs =0 - not running
&green kudzu =0 - not running
&green crond >=1 - 1 instance running
&green ntpd >=1 - 1 instance running
&green sshd >=1 - 5 instances running
&green init >=1 - 1 instance running
&green sendmail =0 - not running
&green nfslock =0 - not running
&green nfs =0 - not running
&green portmap =0 - not running
&green isdn =0 - not running
&green irda =0 - not running
&green autofs =0 - not running

Please see: http://kazoo/bb/html/kazoo.procs.html
```



The syslog-ng daemon is restarted, BB notification is sent informing staff all is fine in the world. Finally, the BB Web page is updated with the normal status.

```
# /etc/rc.d/init.d/syslog-ng start
Starting /usr/sbin/stunnel:
Starting Kernel Logger:                    [  OK  ]
```

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2 22:40:12 2004
Date: Tue, 2 Mar 2004 22:40:12 -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@tFoobar.com
Subject: BB - 2089010  kazoo.procs Recovered

[0000000 ] kazoo procs recovered 03/02/2004 22:40:12
```

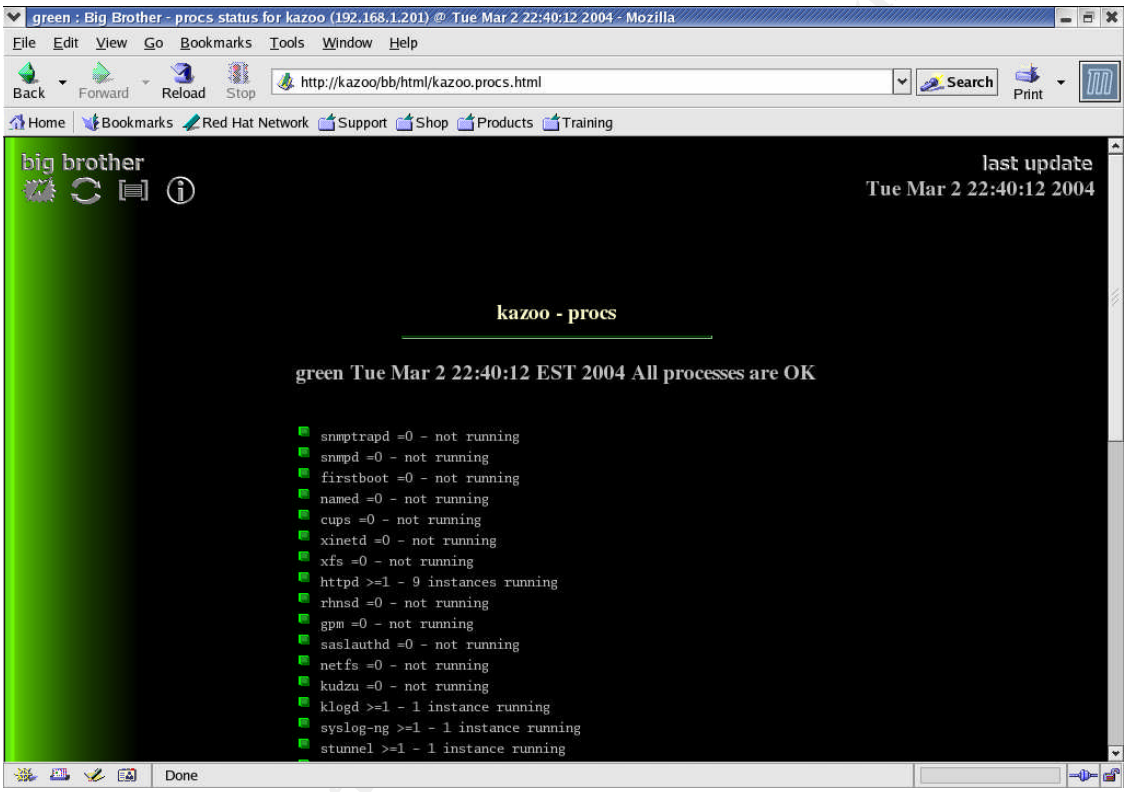
E-mail notification

```
From bbuser@kazoo Tue Mar  2 22:40:21 2004
```

Date: Tue, 2 Mar 2004 22:40:12 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010 kazoo.proc Problem

[0000000] kazoo procs recovered [0000000] kazoo.procs recovered Tue Mar 2 22:40:12 2004 Problem has been resolved after 347 seconds

Please see: http://kazoo/bb/html/kazoo.procs.html



Discussion:
Every computer runs mission critical daemons and applications that drive the business. When mission critical processes aren't running the business may loose revenue, the administrator has a tarnished image, and the result is unhappy customers. If through malice or stupidity a mission critical process is terminated, BB can help the business recover quickly without a major interruption of service. From a security perspective staff needs to be alerted if a mission critical process terminates. If a machine is being attacked precious time isn't lost while the machine is under attack.

e) Verify log file monitoring

Verify the log monitoring facility of BB. BB should detect if there has been alterations to our system log files. BB should detect a zero length system log file. To test the log file monitoring facility of BB we will remove/clear /var/adm/messages. This may have

happened either from a system compromise, or a mistake made by staff, in either situation we need to correct the situation immediately.

Stop and clear the /var/adm/messages log file.

```
# cd /var/log
# /etc/rc.d/init.d/syslog-ng stop
Stopping /usr/sbin/stunnel:                          [  OK  ]
# mv messages messages.good
# >messages
# /etc/rc.d/init.d/syslog-ng start
Starting /usr/sbin/stunnel:
Starting Kernel Logger:                              [  OK  ]
# chmod 440 messages
# ls -l messages
-r--r-----    1 root     root            0 Mar  2 22:58 messages
```

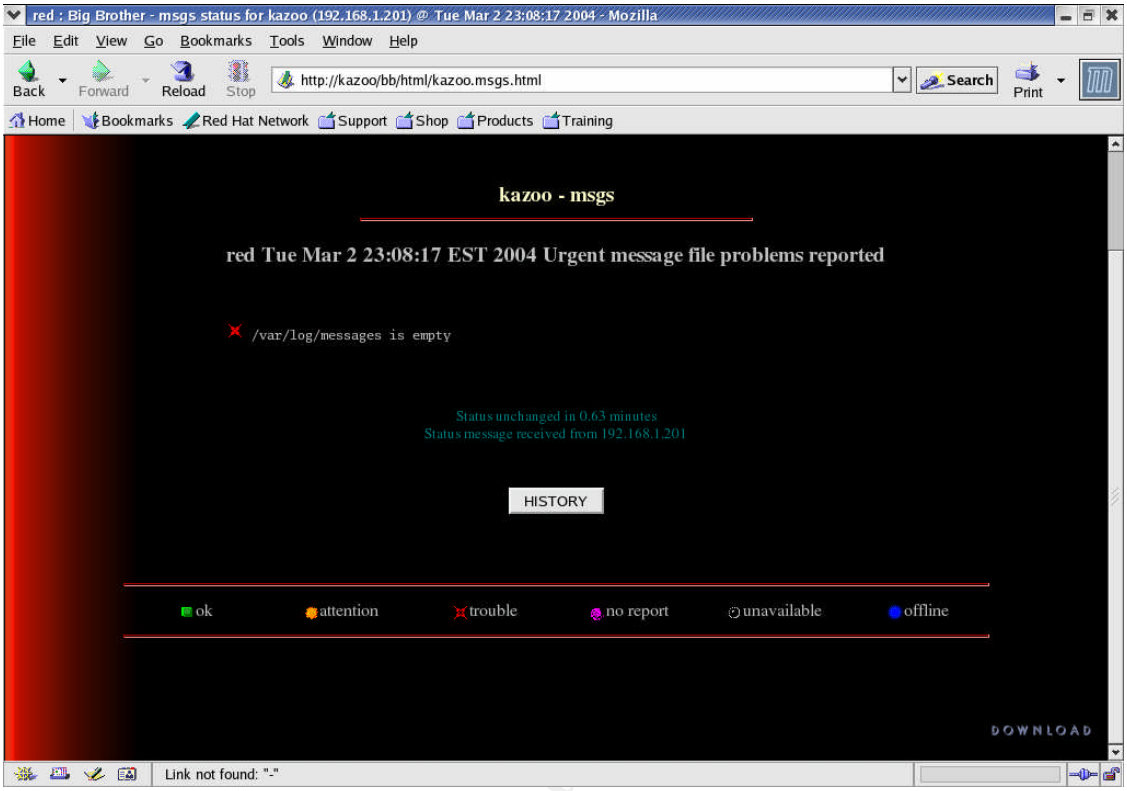
BB detects a problem and sends a notification message to staff, as in previous examples I have included the full notification send via e-mail, and the terse messages sent to a pager/cell phone.

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2 23:12:52 2004
Date: Tue, 2 Mar 2004 23:12:52 -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@Foobar.com
Subject: BB - 2089010  kazoo.msgs Problem

[[2335710 ] kazoo msgs Problem 03/02/2004 23:12:52
```

E-mail notification

```
From bbuser@kazoo Tue Mar  2 23:12:52 2004
Date: Tue, 2 Mar 2004 23:12:52 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  kazoo.msgs Problem

[2335710 ] kazoo msgs Problem [2335710]  kazoo.msgs red Tue Mar  2 23:12:52 EST 2004 Urgent
message file problems reported


&red /var/log/messages is empty

Please see: http://kazoo/bb/html/kazoo.msgs.html
```



Once the problem has been corrected, verification of a normal condition is tested. BB should detect the log files are normal, notifying staff to the normal condition.

```
# cd /var/log
# /etc/rc.d/init.d/syslog-ng stop
Stopping /usr/sbin/stunnel:                          [  OK  ]
# mv messages.good messages
mv: overwrite `messages'? y
# ls -l messages
-r--r-----    1 root     adm       129452 Mar  2 22:47 messages
# /etc/rc.d/init.d/syslog-ng start
Starting /usr/sbin/stunnel:
Starting Kernel Logger:                              [  OK  ]
```

Page Notification

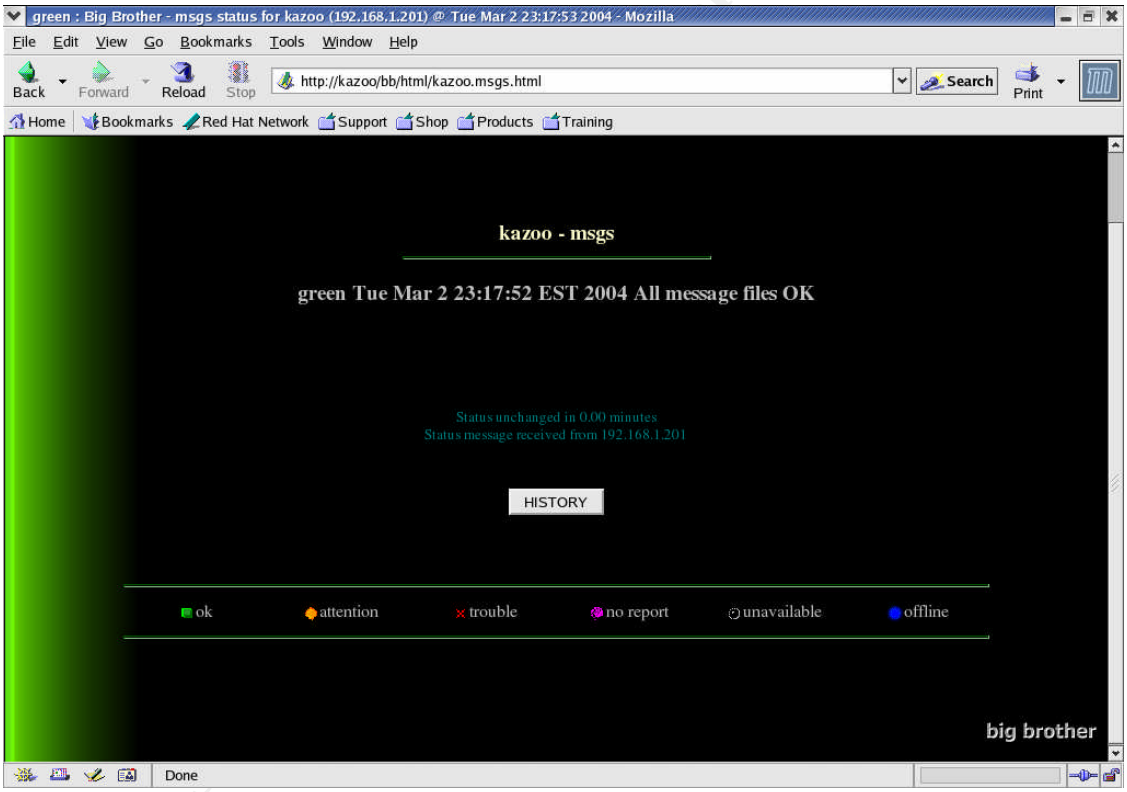
```
From bbuser@tFoobar.com Tue Mar  2  23:17:53  2004
Date: Tue, 2 Mar 2004  23:17:53  -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@Foobar.com
Subject: BB - 2089010  kazoo.msgs Problem

 [0000000 ] kazoo msgs recovered 03/02/2004 23:17:53
```

E-mail notification

From bbuser@kazoo Tue Mar 2 23:17:53 2004
Date: Tue, 2 Mar 2004 23:17:53 -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010 kazoo.msgs Problem

[0000000] kazoo msgs recovered [0000000] kazoo.msgs recovered Tue Mar 2 23:17:53 2004 Problem has been resolved after 614 seconds

Please see: http://kazoo/bb/html/kazoo.msgs.html



Discussion:

For this test, system log files were truncated, creating a zero length system log file. This was an attempt to simulate a brute force attack from a hacker with little skill. The attacker knew enough to remove log files, but the attacker wasn't smart enough to cover his/her tracks. BB detected the empty log files and created a major alert. The Web page turned red indicating a major condition; the result is

notification to staff. Log files are a mission critical component to any well run system. Not only do log files assist staff with problem analysis, log files are used to correct problems, and provide valuable information in event a system was compromised. BB has done an admirable job watching the log files.

f) Verify network services notification – shh node up/down

One of the most important functions of a monitoring system is detection of loss-of-connectivity. In this test the system "bud" will be halted to verify BB detects a loss-of-connectivity.

On Machine bud:

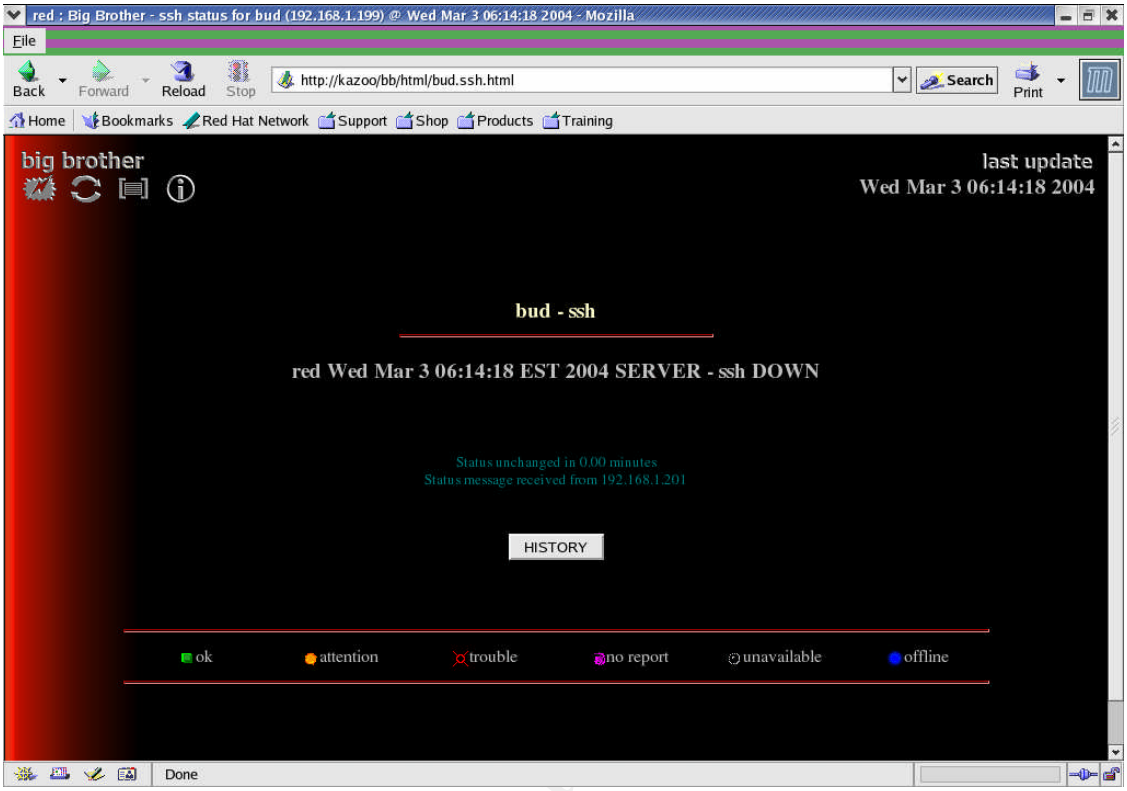
```
# halt
```

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2  23:17:53  2004
Date: Tue, 2 Mar 2004  23:17:53  -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@Foobar.com
Subject: BB - 2089010  bud.ssh Problem

  [1387410 ] bud ssh Problem 03/03/2004 06:14:18
```

E-mail notification

```
From bbuser@kazoo Tue Mar  2 23:17:53 2004
Date: Tue, 2 Mar 2004  23:17:53  -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  bud.ssh Problem

 [1387410 ] bud ssh Problem [1387410]  bud.ssh red Wed Mar  3 06:14:18 EST 2004 SERVER
- ssh DOWN

Please see: http://kazoo/bb/html/bud.ssh.html
```



Power the machine on.

Page Notification

```
From bbuser@tFoobar.com Tue Mar  2  23:17:53  2004
Date: Tue, 2 Mar 2004  23:17:53  -0500 (EST)
From: Big Brother User Account <bbuser@Foobar>
To: UnixPagers@Foobar.com
Subject: BB - 2089010  bud.ssh Problem

  [0000000 ] bud ssh recovered 03/03/2004 06:23:49
```

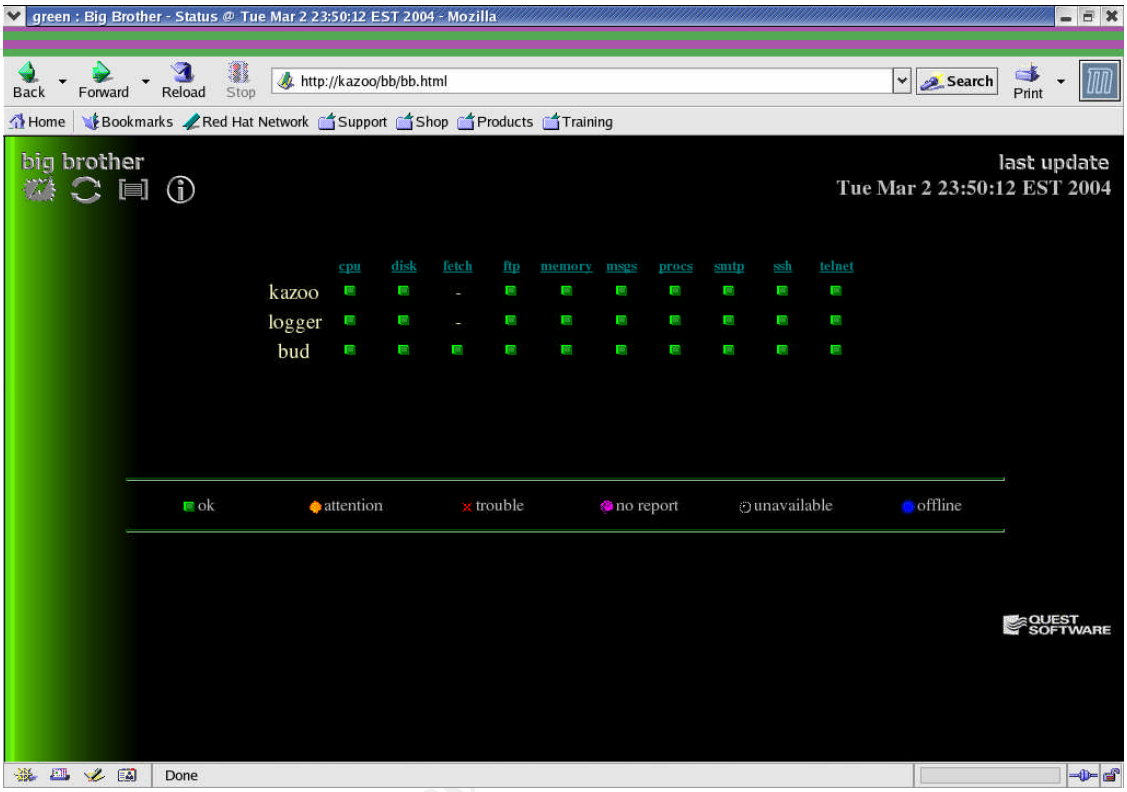
E-mail notification

```
From bbuser@kazoo Tue Mar  2 23:17:53 2004
Date: Tue, 2 Mar 2004  23:17:53  -0500 (EST)
From: Big Brother User Account <bbuser@kazoo>
To: UnixEmail@Foobar.com
Subject: !BB - 2089010  bud.ssh Problem

[0000000 ] bud ssh recovered [0000000]  bud.ssh recovered Wed Mar  3 06:19:16 2004 Problem has
been resolved after 298 seconds

Please see: http://kazoo/bb/html/bud.ssh.html
```

Discussion:
Testing network services is critical. BB allows an organization to verify a network service either is or isn't running, the condition is determined by business needs. In this case we tested for ssh connectivity. If the BB process bbnet can't connect to a server, there is a very good possibility something is wrong with the server, or there is a network issue. The administration team should be mobilized to restore service to the user community. One of the most frustrating things that can happen to an administrator is receiving a call from a user informing them a machine is "down." Every administrator has received the famous call from a user informing him or her the Internet is down. In most cases the administration team should know about connectivity issues before the user community. From a security perspective, staff needs to be informed of services running which are not business critical. There may be staff, or worse an attacker activating dangerous or non-business critical services. The running of non-business services must be dealt with immediately to reduce exposure to exploitation. Part of our design verified telnet, ftp and smtp are not running on machines. Although we tested ssh connectivity, the test could have verified an unsecured service was not running.

G) Big Brother Bliss

Take a long hard look at this picture; this is the main BB Web page with no problems: minor, major or anything else, all is working. In a medium to large environment this is not seen very often. ☺



Appendix

/etc/sysctl.conf

```
# Kernel sysctl configuration file for Red Hat Linux
#
# For binary values, 0 is disabled, 1 is enabled.  See sysctl(8) and
# sysctl.conf(5) for more details.

#######################################################
#
# Values were changed by Recommendation from
# SANS Institute
# Track 6 - Securing Unix
# 6.5 Unix Practicum
# Hal Pomeranz 2003
#
# Added by JBH 02/05/2004
#
#######################################################
```

```
#
# Server will not act as a gateway or router disable parameter
# which controls IP packet forwarding
#
net.ipv4.ip_forward = 0

#
# Controls source route verification
# will enable IP spoof protection
#
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1

# Controls the System Request debugging functionality of the kernel
kernel.sysrq = 0

#
# Controls whether core dumps will append the PID to the core filename.
# Useful for debugging multi-threaded applications.
#
kernel.core_uses_pid = 1

#
# Enables SYN flood protection
#
net.ipv4.tcp_max_syn_backlog = 4096

#
# Disable IP source routing
#
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0
```

```
#
# Disables ICMP redirect acceptance
#
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.default.accept_redirects = 0

#
#
#
net.ipv4.conf.all.secure_redirects = 0
net.ipv4.conf.default.secure_redirects = 0

#
# Disable the ability to send ICMP redirects
#
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0

#
# Log packets with impossible source addresses, any address
# not part of the same network as the local machine
#
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.default.log_martians = 1
```

/etc/rc.d/init.d/bigbrother

```
#! /bin/sh
###############################################################################
#
# Program: Big Brother init script
#
# Description:
#
#  This is an init script for Big Brother on the Linux platform.
#
#  It totally relies on the Redhat function library and works the same
#  way as other typical Redhat init scripts.
#
#
# Platforms (tested): Linux (Redhat 9.0)
#
# Author: Jim Horwath <jim.horwath@rcn.com>
#
# Last Changed: February 21, 2004
#
###############################################################################

###############################################################################
#
# Blurb needed to have chkconfig reconize the script
# chkconfig: 2345 12 88
# description: bigbrother
#
```

```
##############################################################################

# Source Redhat function library.
#
. /etc/rc.d/init.d/functions

umask 077
ulimit -c 0

# See how we were called.
case "$1" in
  start)
        echo "Starting Big Brother: "
        /bin/su bbuser -c "/usr/local/bb/runbb.sh start"
        ;;
  stop)
        echo -n "Stopping Big Brother: "
        /bin/su bbuser -c "/usr/local/bb/runbb.sh stop"
        ;;
  *)
        echo "Usage: $0 {start|stop}"
        exit 1
esac

exit 0
```

Sylog-ng.conf

```
# syslog-ng configuration file.
#
# This should behave pretty much like the original syslog on Red Hat. But
# it could be configured a lot smarter.
#
# See syslog-ng(8) and syslog-ng.conf(5) for more information.
#
# 20000925 gb@sysfive.com
#
# Updated by Frank Crawford (<Frank.Crawford@ac3.com.au>) - 10 Aug 2002
#       - for Red Hat 7.3
#       - totally do away with klogd
#       - add message "kernel:" as is done with klogd.
#
# Updated by Frank Crawford (<Frank.Crawford@ac3.com.au>) - 22 Aug 2002
#       - use the log_prefix option as per Balazs Scheidler's email
#
# Modified 01/05/2004 to meet JBH SANS practical needs
# -jbh-
#


#
# Global definition section.  These definitions will be in
# afect for the entire file, however each local section can
# override a configuration set here.
#
options {
```

```
        sync (0);
        time_reopen (10);
        log_fifo_size (1000);
        long_hostnames (off);
        use_dns (no);
        use_fqdn (yes);
        create_dirs (yes);
        keep_hostname (no);
        # File handling
        owner(root);
        group(adm);
        perm(0440);
      };

source s_sys { pipe ("/proc/kmsg" log_prefix("kernel: ")); unix-stream ("/dev/log"); internal(); };

#
# Setup the central syslog host
#
destination stunnel {tcp("127.0.0.1" port(514));};

#
# Destination log file locations on the local host,
# the log level is high to catch suspicious activity.
#
destination d_cons { file("/dev/console"); };
destination d_kern { file("/var/log/kernel"); };
destination d_mesg { file("/var/log/messages"); };
destination d_auth { file("/var/log/secure"); };
destination d_mail { file("/var/log/maillog"); };
destination d_spol { file("/var/log/spooler"); };
destination d_boot { file("/var/log/boot.log"); };
destination d_cron { file("/var/log/cron"); };
destination d_mlal { usertty("*"); };

#
# Filtered messages
#
filter f_filter1    { facility(kern); };
filter f_filter2    { level(info) and
                  not (facility(mail)
                    or facility(authpriv) or facility(cron)); };
filter f_filter3    { facility(authpriv); };
filter f_filter4    { facility(mail); };
filter f_filter5    { level(emerg); };
filter f_filter6    { facility(uucp) or
                  (facility(news) and level(crit)); };
filter f_filter7    { facility(local7); };
filter f_filter8    { facility(cron); };

#
# Log messages to their individual files locally and on
# the central log host
#
log { source(s_sys); filter(f_filter1); destination(d_kern); };
```

```
log { source(s_sys); filter(f_filter2); destination(stunnel); };

log { source(s_sys); filter(f_filter2); destination(d_mesg); };
log { source(s_sys); filter(f_filter2); destination(stunnel); };

log { source(s_sys); filter(f_filter3); destination(d_auth); };
log { source(s_sys); filter(f_filter3); destination(stunnel); };

log { source(s_sys); filter(f_filter4); destination(d_mail); };
log { source(s_sys); filter(f_filter4); destination(stunnel); };

log { source(s_sys); filter(f_filter5); destination(d_mlal); };
log { source(s_sys); filter(f_filter5); destination(stunnel); };

log { source(s_sys); filter(f_filter6); destination(d_spol); };
log { source(s_sys); filter(f_filter6); destination(stunnel); };

log { source(s_sys); filter(f_filter7); destination(d_boot); };
log { source(s_sys); filter(f_filter7); destination(stunnel); };

log { source(s_sys); filter(f_filter8); destination(d_cron); };
log { source(s_sys); filter(f_filter8); destination(stunnel); };
```

iptables start script

```
#! /bin/sh
# init.d/rc.d/iptables
#
# chkconfig: 2345 08 92
#
# Based on the example given by Michael D. Bauer
# pp 74-76
# Building Secure Servers with Linux
# O'Reilly Publishing Copyright 2003
#
# Munged by Jim Horwath
# Since this is running of the BB node there
# is more traffic being let out then I would normally
# let out.  BB needs to check if network services are
# running.  If I block the out going ftp traffic BB can't
# check if ftp is disabled on a node.
# last modified 02/16/2004
#

#
# Iptable executable location
#
IPTABLES=/sbin/iptables
test -x ${IPTABLE} || exit 5

#
# There is the standard start, stop and an
# additional wide-open option to disable
# iptables.  This is not recommended.
#
case "${1}" in
start)
echo -n "Loading Kazoo's Packet Filters"

#
# SETUP -- stuff necessary for any host
#
modprobe ip_tables

#
# Flush old rules, old custom tables
#
${IPTABLES} --flush
${IPTABLES} --delete-chain

# Set default-deny policies for all three default chains
${IPTABLES} -P INPUT DROP
${IPTABLES} -P FORWARD DROP
${IPTABLES} -P OUTPUT DROP

# Give free reign to loopback interfaces
${IPTABLES} -A INPUT -i lo -j ACCEPT
${IPTABLES} -A OUTPUT -o lo -j ACCEPT

# Do some rudimentry anti-IP-spoofinf drops
${IPTABLES} -A INPUT -s 255.0.0.0/8 -j LOG --log-prefix "Spoofed source IP!"
${IPTABLES} -A INPUT -s 255.0.0.0/8 -j DROP
${IPTABLES} -A INPUT -s 0.0.0.0/8 -j LOG --log-prefix "Spoofed source IP!"
${IPTABLES} -A INPUT -s 0.0.0.0/8 -j DROP
${IPTABLES} -A INPUT -s 127.0.0.0/8 -j LOG --log-prefix "Spoofed source IP!"
${IPTABLES} -A INPUT -s 127.0.0.0/8 -j DROP
# remove this comment when we move to production
#${IPTABLES} -A INPUT -s 192.168.0.0/16 -j LOG --log-prefix "Spoofed source IP!"#${IPTABLES} -A INPUT -s 192.168.0.0/16 -j
DROP
${IPTABLES} -A INPUT -s 172.16.0.0/16 -j LOG --log-prefix "Spoofed source IP!"
${IPTABLES} -A INPUT -s 172.16.0.0/16 -j DROP
${IPTABLES} -A INPUT -s 10.16.0.0/8 -j LOG --log-prefix "Spoofed source IP!"
```

```
${IPTABLES} -A INPUT -s 10.16.0.0/8 -j DROP
#${IPTABLES} -A INPUT -s 192.168.1.201 -j LOG --log-prefix "Spoofed Kazoo!"
#${IPTABLES} -A INPUT -s 192.168.1.201 -j DROP

# Tell netfilter that all TCP session do indeed begin with SYN
${IPTABLES} -A INPUT -p tcp ! --syn -m state --state NEW -j LOG --log-prefix "Stealth scan attempt?"
${IPTABLES} -A INPUT -p tcp ! --syn -m state --state NEW -j DROP

# Finally, the meat of our packet-filtering policy

# INBOUND POLICY

# Accept inbound packets that are part of previouly-OK'ed sessions
${IPTABLES} -A INPUT -j ACCEPT -m state --state ESTABLISHED,RELATED

# Accept inbound packets which initiate SSH sessions
${IPTABLES} -A INPUT -p tcp -j ACCEPT --dport 22 -m state --state NEW

# Accept inbound packets which initiate BB sessions
${IPTABLES} -A INPUT -p tcp -j ACCEPT --dport 1984 -m state --state NEW

# Accept inbound packets which initiate HTTP sessions
${IPTABLES} -A INPUT -p tcp -j ACCEPT --dport 80 -m state --state NEW

# Accept inbound packets which initiate TSM sessions
${IPTABLES} -A INPUT -p tcp -j ACCEPT --dport 4000 -m state --state NEW

# Log anything not accepted above
${IPTABLES} -A INPUT -j LOG --log-prefix "Dropped by default:"

# OUTBOUND POLICY

# If it's part of an approved connection, let it out
${IPTABLES} -I OUTPUT 1 -m state --state RELATED,ESTABLISHED -j ACCEPT

# Allow outbound FTP for BB to test ftp status on machines
${IPTABLES} -A OUTPUT -p tcp --dport 21 -m state --state NEW -j ACCEPT

# Allow outbound SSH for administration and BB
${IPTABLES} -A OUTPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT

# Allow outbound telnet traffic for BB to check daemons
${IPTABLES} -A OUTPUT -p tcp --dport 23 -m state --state NEW -j ACCEPT

# Allow outbound SMTP to allow BB to check service is down
${IPTABLES} -A OUTPUT -p tcp --dport 25 -m state --state NEW -j ACCEPT

# Allow outbound DNS queries to resolve IPs
${IPTABLES} -A OUTPUT -p udp --dport 53 -m state --state NEW -j ACCEPT

# Allow outbound NTP queries to keep clocks in sync
${IPTABLES} -A OUTPUT -p udp --dport 123 -m state --state NEW -j ACCEPT

# Allow outbound BB traffic
${IPTABLES} -A OUTPUT -p tcp  --dport 1984 -m state --state NEW -j ACCEPT

# Allow outbound TSM traffic
${IPTABLES} -A OUTPUT -p tcp  --dport 4000 -m state --state NEW -j ACCEPT

# Log anything not accepted above - if nothing else for trouble shooting
${IPTABLES} -A OUTPUT -j LOG --log-prefix "Dropped by default:"
;;

wide_open)
echo -n "DANGER!!! Unloading Kazoo's Packet Filter"
# Unload filters nad reset default policies to ACCEPT
# For emergency us only -- else use stop!!
${IPTABLES} --flush
${IPTABLES} -P INPUT ACCEPT
```

```
${IPTABLES} -P FORWARD ACCEPT
${IPTABLES} -P OUTPUT ACCEPT
;;

stop)
echo -n "Unloading all iptable rules, leaving default-drop policies"
${IPTABLES} --flush
;;

*)
echo "Usage: ${0} {start|stop|wide_open|status}"
exit 1
;;
esac

exit 0
```

/etc/rc/init.d/syslog-ng

```
#############################################################################
#
# Program: syslog-ng init script
#
# Description:
#
#  This is an init script for syslog-ng on the Linux platform.
#
#  It totally relies on the Redhat function library and works the same
#  way as other typical Redhat init scripts.
#
#
# Platforms (tested): Linux (Redhat 6.1)
#
#
# Author: Gregor Binder <gbinder@sysfive.com>
#
# Last Changed: October 10, 2000
#
#     Copyright (c) 2000 by sysfive.com GmbH, All rights reserved.
#
#############################################################################

#############################################################################
#
# Blurb needed to have chkconfig reconize the script
# chkconfig: 2345 12 88
# description: Syslog-ng
#
#############################################################################

#############################################################################
# configuration
#
INIT_PROG="/usr/local/sbin/syslog-ng"    # Full path to daemon
INIT_OPTS=""                    # options passed to daemon

STUNNEL_PGM="/usr/sbin/stunnel"
PATH=/bin:/sbin:/usr/bin:/usr/sbin

INIT_NAME=`basename "$INIT_PROG"`

# Source Redhat function library.
#
```

```
. /etc/rc.d/init.d/functions

# Uncomment this if you are on Redhat and think this is useful

. /etc/sysconfig/network

if [ ${NETWORKING} = "no" ]
then
        exit 0
fi

RETVAL=0

umask 077
ulimit -c 0

# See how we were called.
case "$1" in
  start)
        echo "Starting $STUNNEL_PGM: "
        $STUNNEL_PGM
        echo -n "Starting $INIT_NAME: "
        daemon --check $INIT_PROG "$INIT_PROG $INIT_OPTS"
        RETVAL=$?
        echo -n "Starting Kernel Logger: "
        [ -x "/sbin/klogd" ] && daemon klogd
        echo
        [ $RETVAL -eq 0 ] && touch "/var/lock/subsys/${INIT_NAME}"
        ;;
  stop)
        echo -n "Stopping $INIT_NAME: "
        killproc $INIT_PROG
        RETVAL=$?
        echo -n "Stopping Kernel Logger: "
        [ -x "/sbin/klogd" ] && killproc klogd
        echo -n "Stopping $STUNNEL_PGM: "
        killproc $STUNNEL_PGM
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f "/var/lock/subsys/${INIT_NAME}"
        ;;
  status)
        status $INIT_PROG
        RETVAL=$?
        ;;
  restart|reload)
        $0 stop
        $0 start
        RETVAL=$?
        ;;
  *)
        echo "Usage: $0 {start|stop|status|restart|reload}"
        exit 1
esac

exit $RETVAL
```

/usr/local/apache2/conf/httpd.conf

```
# HeaderName is the name of a file which should be prepended to
# directory indexes.
#jbh#ReadmeName README.html
#jbh#HeaderName HEADER.html

#
```

```
# IndexIgnore is a set of filenames which directory indexing should ignore
# and not include in the listing.  Shell-style wildcarding is permitted.
#
#jbh#IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t

#
# DefaultLanguage and AddLanguage allows you to specify the language of
# a document. You can then use content negotiation to give a browser a
# file in a language the user can understand.
#
# Specify a default language. This means that all data
# going out without a specific language tag (see below) will
# be marked with this one. You probably do NOT want to set
# this unless you are sure it is correct for all cases.
#
# * It is generally better to not mark a page as
# * being a certain language than marking it with the wrong
# * language!
#
# DefaultLanguage nl
#
# Note 1: The suffix does not have to be the same as the language
# keyword --- those with documents in Polish (whose net-standard
# language code is pl) may wish to use "AddLanguage pl .po" to
# avoid the ambiguity with the common suffix for perl scripts.
#
# Note 2: The example entries below illustrate that in some cases
# the two character 'Language' abbreviation is not identical to
# the two character 'Country' code for its country,
# E.g. 'Danmark/dk' versus 'Danish/da'.
#
# Note 3: In the case of 'ltz' we violate the RFC by using a three char
# specifier. There is 'work in progress' to fix this and get
# the reference data for rfc1766 cleaned up.
#
# Catalan (ca) - Croatian (hr) - Czech (cs) - Danish (da) - Dutch (nl)
# English (en) - Esperanto (eo) - Estonian (et) - French (fr) - German (de)
# Greek-Modern (el) - Hebrew (he) - Italian (it) - Japanese (ja)
# Korean (ko) - Luxembourgeois* (ltz) - Norwegian Nynorsk (nn)
# Norwegian (no) - Polish (pl) - Portugese (pt)
# Brazilian Portuguese (pt-BR) - Russian (ru) - Swedish (sv)
# Simplified Chinese (zh-CN) - Spanish (es) - Traditional Chinese (zh-TW)
#
AddLanguage ca .ca
AddLanguage cs .cz .cs
AddLanguage da .dk
AddLanguage de .de
AddLanguage el .el
AddLanguage en .en
AddLanguage eo .eo
AddLanguage es .es
AddLanguage et .et
AddLanguage fr .fr
AddLanguage he .he
AddLanguage hr .hr
AddLanguage it .it
AddLanguage ja .ja
AddLanguage ko .ko
AddLanguage ltz .ltz
AddLanguage nl .nl
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
AddLanguage pt .pt
AddLanguage pt-BR .pt-br
AddLanguage ru .ru
AddLanguage sv .sv
AddLanguage zh-CN .zh-cn
AddLanguage zh-TW .zh-tw
```

```
#
# LanguagePriority allows you to give precedence to some languages
# in case of a tie during content negotiation.
#
# Just list the languages in decreasing order of preference. We have
# more or less alphabetized them here. You probably want to change this.
#
LanguagePriority en ca cs da de el eo es et fr he hr it ja ko ltz nl nn no pl pt pt-BR ru sv zh-CN zh-TW

#
# ForceLanguagePriority allows you to serve a result page rather than
# MULTIPLE CHOICES (Prefer) [in case of a tie] or NOT ACCEPTABLE (Fallback)
# [in case no accepted languages matched the available variants]
#
ForceLanguagePriority Prefer Fallback

#
# Specify a default charset for all pages sent out. This is
# always a good idea and opens the door for future internationalisation
# of your web site, should you ever want it. Specifying it as
# a default does little harm; as the standard dictates that a page
# is in iso-8859-1 (latin1) unless specified otherwise i.e. you
# are merely stating the obvious. There are also some security
# reasons in browsers, related to javascript and URL parsing
# which encourage you to always set a default char set.
#
AddDefaultCharset ISO-8859-1

#
# Commonly used filename extensions to character sets. You probably
# want to avoid clashes with the language extensions, unless you
# are good at carefully testing your setup after each change.
# See http://www.iana.org/assignments/character-sets for the
# official list of charset names and their respective RFCs.
#
AddCharset ISO-8859-1  .iso8859-1  .latin1
AddCharset ISO-8859-2  .iso8859-2  .latin2 .cen
AddCharset ISO-8859-3  .iso8859-3  .latin3
AddCharset ISO-8859-4  .iso8859-4  .latin4
AddCharset ISO-8859-5  .iso8859-5  .latin5 .cyr .iso-ru
AddCharset ISO-8859-6  .iso8859-6  .latin6 .arb
AddCharset ISO-8859-7  .iso8859-7  .latin7 .grk
AddCharset ISO-8859-8  .iso8859-8  .latin8 .heb
AddCharset ISO-8859-9  .iso8859-9  .latin9 .trk
AddCharset ISO-2022-JP .iso2022-jp .jis
AddCharset ISO-2022-KR .iso2022-kr .kis
AddCharset ISO-2022-CN .iso2022-cn .cis
AddCharset Big5        .Big5       .big5
# For russian, more than one charset is used (depends on client, mostly):
AddCharset WINDOWS-1251 .cp-1251   .win-1251
AddCharset CP866       .cp866
AddCharset KOI8-r      .koi8-r .koi8-ru
AddCharset KOI8-ru     .koi8-uk .ua
AddCharset ISO-10646-UCS-2 .ucs2
AddCharset ISO-10646-UCS-4 .ucs4
AddCharset UTF-8       .utf8

# The set below does not map to a specific (iso) standard
# but works on a fairly wide range of browsers. Note that
# capitalization actually matters (it should not, but it
# does for some browsers).
#
# See http://www.iana.org/assignments/character-sets
# for a list of sorts. But browsers support few.
#
AddCharset GB2312      .gb2312 .gb
AddCharset utf-7       .utf7
AddCharset utf-8       .utf8
```

```
AddCharset big5        .big5 .b5
AddCharset EUC-TW      .euc-tw
AddCharset EUC-JP      .euc-jp
AddCharset EUC-KR      .euc-kr
AddCharset shift_jis   .sjis

#
# AddType allows you to add to or override the MIME configuration
# file mime.types for specific file types.
#
#AddType application/x-tar .tgz
#
# AddEncoding allows you to have certain browsers uncompress
# information on the fly. Note: Not all browsers support this.
# Despite the name similarity, the following Add* directives have nothing
# to do with the FancyIndexing customization directives above.
#
#AddEncoding x-compress .Z
#AddEncoding x-gzip .gz .tgz
#
# If the AddEncoding directives above are commented-out, then you
# probably should define those extensions to indicate media types:
#
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz

#
# AddHandler allows you to map certain file extensions to "handlers":
# actions unrelated to filetype. These can be either built into the server
# or added with the Action directive (see below)
#
# To use CGI scripts outside of ScriptAliased directories:
# (You will also need to add "ExecCGI" to the "Options" directive.)
#
#AddHandler cgi-script .cgi

#
# For files that include their own HTTP headers:
#
#AddHandler send-as-is asis

#
# For server-parsed imagemap files:
#
#AddHandler imap-file map

#
# For type maps (negotiated resources):
# (This is enabled by default to allow the Apache "It Worked" page
#  to be distributed in multiple languages.)
#
AddHandler type-map var

#
# Filters allow you to process content before it is sent to the client.
#
# To parse .shtml files for server-side includes (SSI):
# (You will also need to add "Includes" to the "Options" directive.)
#
#AddType text/html .shtml
#AddOutputFilter INCLUDES .shtml

#
# Action lets you define media types that will execute a script whenever
# a matching file is called. This eliminates the need for repeated URL
# pathnames for oft-used CGI file processors.
# Format: Action media/type /cgi-script/location
# Format: Action handler-name /cgi-script/location
#
```

```
#
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
#
# Some examples:
#ErrorDocument 500 "The server made a boo boo."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html
#

#
# Putting this all together, we can internationalize error responses.
#
# We use Alias to redirect any /error/HTTP_<error>.html.var response to
# our collection of by-error message multi-language collections.  We use
# includes to substitute the appropriate text.
#
# You can modify the messages' appearance without changing any of the
# default HTTP_<error>.html.var files by adding the line:
#
#   Alias /error/include/ "/your/include/path/"
#
# which allows you to create your own set of files by starting with the
# /usr/local/apache2/error/include/ files and copying them to /your/include/path/,
# even on a per-VirtualHost basis.  The default include files will display
# your Apache version number and your ServerAdmin email address regardless
# of the setting of ServerSignature.
#
# The internationalized error documents require mod_alias, mod_include
# and mod_negotiation.  To activate them, uncomment the following 30 lines.

#    Alias /error/ "/usr/local/apache2/error/"
#
#    <Directory "/usr/local/apache2/error">
#        AllowOverride None
#        Options IncludesNoExec
#        AddOutputFilter Includes html
#        AddHandler type-map var
#        Order allow,deny
#        Allow from all
#        LanguagePriority en cs de es fr it nl sv pt-br ro
#        ForceLanguagePriority Prefer Fallback
#    </Directory>
#
#    ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
#    ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
#    ErrorDocument 403 /error/HTTP_FORBIDDEN.html.var
#    ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
#    ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
#    ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
#    ErrorDocument 410 /error/HTTP_GONE.html.var
#    ErrorDocument 411 /error/HTTP_LENGTH_REQUIRED.html.var
#    ErrorDocument 412 /error/HTTP_PRECONDITION_FAILED.html.var
#    ErrorDocument 413 /error/HTTP_REQUEST_ENTITY_TOO_LARGE.html.var
#    ErrorDocument 414 /error/HTTP_REQUEST_URI_TOO_LARGE.html.var
#    ErrorDocument 415 /error/HTTP_UNSUPPORTED_MEDIA_TYPE.html.var
#    ErrorDocument 500 /error/HTTP_INTERNAL_SERVER_ERROR.html.var
#    ErrorDocument 501 /error/HTTP_NOT_IMPLEMENTED.html.var
#    ErrorDocument 502 /error/HTTP_BAD_GATEWAY.html.var
#    ErrorDocument 503 /error/HTTP_SERVICE_UNAVAILABLE.html.var
#    ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var


#
# The following directives modify normal HTTP response behavior to
# handle known problems with browser implementations.
#
```

```
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0

#
# The following directive disables redirects on non-GET requests for
# a directory that does not include the trailing slash.  This fixes a
# problem with Microsoft WebFolders which does not appropriately handle
# redirects for folders with DAV methods.
# Same deal with Apple's DAV filesystem and Gnome VFS support for DAV.
#
BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[012]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully

#
# Allow server status reports generated by mod_status,
# with the URL of http://servername/server-status
# Change the ".example.com" to match your domain to enable.
#
#<Location /server-status>
#    SetHandler server-status
#    Order deny,allow
#    Deny from all
#    Allow from .example.com
#</Location>

#
# Allow remote server configuration reports, with the URL of
#  http://servername/server-info (requires that mod_info.c be loaded).
# Change the ".example.com" to match your domain to enable.
#
#<Location /server-info>
#    SetHandler server-info
#    Order deny,allow
#    Deny from all
#    Allow from .example.com
#</Location>


#
# Bring in additional module-specific configurations
#
<IfModule mod_ssl.c>
    Include conf/ssl.conf
</IfModule>


### Section 3: Virtual Hosts
#
# VirtualHost: If you want to maintain multiple domains/hostnames on your
# machine you can setup VirtualHost containers for them. Most configurations
# use only name-based virtual hosts so the server doesn't need to worry about
# IP addresses. This is indicated by the asterisks in the directives below.
#
# Please see the documentation at
# <URL:http://httpd.apache.org/docs-2.0/vhosts/>
# for further details before you try to setup virtual hosts.
#
# You may use the command line option '-S' to verify your virtual host
# configuration.

#
# Use name-based virtual hosting.
#
#NameVirtualHost *:80
```

```
#
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost container.
# The first VirtualHost section is used for requests without a known
# server name.
#
#<VirtualHost *:80>
#    ServerAdmin webmaster@dummy-host.example.com
#    DocumentRoot /www/docs/dummy-host.example.com
#    ServerName dummy-host.example.com
#    ErrorLog logs/dummy-host.example.com-error_log
#    CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>

#
# Lines to help support mod_security module
# from document:
# http://cvs.sourceforge.net/viewcvs.py/mod-security/mod_security/httpd.conf.example-full
# -JBH-
# 01/13/2004
#
<IfModule mod_security.c>

    # Turn the filtering engine On or Off
    SecFilterEngine On

    # Make sure that URL encoding is valid
    SecFilterCheckURLEncoding On

    # Only allow bytes from this range
    SecFilterForceByteRange 32 126

    # The audit engine works independently and
    # can be turned On of Off on the per-server or
    # on the per-directory basis. "On" will log everything,
    # "DynamicOrRelevant" will log dynamic requests or violations,
    # and "RelevantOnly" will only log policy violations
    SecAuditEngine RelevantOnly

    # The name of the audit log file
    SecAuditLog logs/audit_log

    SecFilterDebugLog logs/modsec_debug_log
    SecFilterDebugLevel 0

    # Should mod_security inspect POST payloads
    SecFilterScanPOST On

    # Action to take by default
    SecFilterDefaultAction "deny,log,status:500"

    # Only accept request encodings we know how to handle
    SecFilterSelective HTTP_Content-Type "!^(|application/x-www-form-urlencoded|multipart/form-data)$"

    # Don't accept transfer encodings we know we don't handle
    # (and you don't need it anyway)
    SecFilterSelective HTTP_Transfer-Encoding "!^$"

    # Redirect user on filter match
    SecFilter xxx redirect:http://www.webkreator.com

    # Execute the external script on filter match
    SecFilter yyy log,exec:/home/users/ivanr/apache/bin/report-attack.pl

    # Simple filter
    SecFilter 111

    # Only check the QUERY_STRING variable
```

```
SecFilterSelective QUERY_STRING 222

# Only check the body of the POST request
SecFilterSelective POST_PAYLOAD 333

# Only check arguments (will work for GET and POST)
SecFilterSelective ARGS 444

# Test filter
SecFilter "/cgi-bin/modsec-test.pl/keyword"

# Another test filter, will be denied with 404 but not logged
# action supplied as a parameter overrides the default action
SecFilter 999 "deny,nolog,status:500"

# Prevent OS specific keywords
SecFilter /etc/passwd

# Prevent path traversal (..) attacks
SecFilter "\.\./"

# Weaker XSS protection but allows common HTML tags
SecFilter "<[[:space:]]*script"

# Prevent XSS atacks (HTML/Javascript injection)
SecFilter "<(.|\n)+>"

# Very crude filters to prevent SQL injection attacks
SecFilter "delete[[:space:]]+from"
SecFilter "insert[[:space:]]+into"
SecFilter "select.+from"

# Require HTTP_USER_AGENT and HTTP_HOST headers
SecFilterSelective "HTTP_USER_AGENT|HTTP_HOST" "^$"

# Forbid file upload
SecFilterSelective "HTTP_CONTENT_TYPE" multipart/form-data

# Only watch argument p1
SecFilterSelective "ARG_p1" 555

# Watch all arguments except p1
SecFilterSelective "ARGS|!ARG_p2" 666

# Only allow our own test utility to send requests (or Mozilla)
SecFilterSelective HTTP_USER_AGENT "!(mod_security|mozilla)"

# Do not allow variables with this name
SecFilterSelective ARGS_NAMES 777

# Do now allow this variable value (names are ok)
SecFilterSelective ARGS_VALUES 888

# Test for a POST variable parsing bug, see test #41
SecFilterSelective ARG_p2 AAA

# Stop spamming through FormMail
# note the exclamation mark at the beginning
# of the filter - only requests that match this regex will
# be allowed
<Location /cgi-bin/FormMail>
    SecFilterSelective "ARG_recipient" "!@webkreator.com$"
</Location>

# when allowing upload, only allow images
# note that this is not foolproof, a determined attacker
# could get around this
<Location /fileupload.php>
    SecFilterInheritance Off
```

```
      SecFilterSelective POST_PAYLOAD "!image/(jpeg|bmp|gif)"
   </Location>

   #
   # Chroot this baby
   #
   #jbh#SecChrootDir /chroot


</IfModule>
```

/usr/local/bb/ext/pg/Foobarpage

```
#!/bin/sh
#
#####################################################################
#                                                                   #
# FoobarPage    - This script will send the appropriate             #
#                 form of notification to staff.  If the            #
#                 receipient is a pager, a terse message is         #
#                 sent.  For Guardian e-mail addresses a            #
#                 longer, more informative page is sent.            #
#                                                                   #
# JBH 01/25/2001                                                    #
#    Version 1.0  Simple first pass, used BB templates for an       #
#                 example.                                          #
#                                                                   #
# Usage: FoobarPage                                                 #
#                                                                   #
#####################################################################
#
# The following variables are available thru the environment
#
# BBALPHAMSG - Default message defined
# ACKCODE - Notification associated ack code
# BBHOSTNAME - hostname in www.xxx.com format
# BBHOSTSVC - host in www.xxx.com.disk format
# BBHOSTSVCCOMMAS - host in www,xxx,com.disk format
# RCPT - recipient
# MACHIP - IP address of host in normalized 12 digits AAABBBCCCDDD
# BBSVCNAME - name of service
# BBSVCNUM - numeric equivalent of service (from svcerrlist in bbwarnsetup.cfg)
# BBNUMERIC - ${MACHIP}${BBSVCNUM}
# BBCOLORLEVEL - Color level
# DOWNSECSMSG - Default recovery string
# DOWNSECS - # of seconds to recovery
# RECOVERED - 1 in recovery mode, anything else non-recovery mode
#
#
# The RCPT value contains the real recipient value.  It was
# stripped of its ext-XXX- prefix (where XXX is what you called
# your notification script, in this example it would be ext-ex3-<recipient>)
#
#########################################################################

#
# BBTEMP variable needs to be set.
#
if [ "$BBTMP" = "" ]
then
        echo "BB environment not set !!!"
        exit 1
fi
```

```
#
# Validate the bbwarnsetup.cfg exists
#
if [ ! -f "${BBHOME}/etc/bbwarnsetup.cfg" ]
then

        echo "BB ${BBHOME}/etc/bbwarnsetup.cfg is missing !!!"
        exit 1
fi

#
# If the staff person exists as a paging group, get the members
# of the paging group from bbwarnsetup.cfg
#
STAFF=`grep "^${RCPT}" ${BBHOME}/etc/bbwarnsetup.cfg | cut -d":" -f2`

#
# Staff was not listed in the bbwarnsetup.cfg file. Use whatever
# contact list was passed as an argument
#
if [ -z "${STAFF}" ]
then
        STAFF="${RCPT}"
fi

#
# Get the list of reciepents who need to have brief messages
# sent, these are typically pagers.
#
BRIEF=""
export BRIEF
BRIEF=`${BBHOME}/ext/pg/ParseNames.pl -f ${STAFF} -o`

#
# Get the list of Guardian e-mail addresses.  Staff will
# receive a detailed BB message.
#
FULL=""
export FULL
FULL=`${BBHOME}/ext/pg/ParseNames.pl -f ${STAFF} -g`

#
# Is this a problem or a receovery?
#
if [ "$BBCOLORLEVEL}" -eq "red" ]
then
        BBCOLORLEVEL=Problem
fi

#
# All BRIEF receipents will receive a message like this:
#
# Subject: BB - 4869010  toolbox.disk
# [4869010 ] toolbox disk red 09/24/2003 15:37:25
#
if [ ! -z "${BRIEF}" ]
then
        echo "${BRIEF} [${ACKCODE}] ${BBHOSTNAME} ${BBSVCNAME} ${BBCOLORLEVEL} `/usr/bin/date '+%m/%d/%Y
%H:%M:%S'`" >> /usr/tmp/brief.bb
        echo "[${ACKCODE}] ${BBHOSTNAME} ${BBSVCNAME} ${BBCOLORLEVEL} `/usr/bin/date '+%m/%d/%Y %H:%M:%S'`" |
$MAIL "BB - ${ACKCODE} ${BBHOSTSVC} ${BBCOLORLEVEL}" "${BRIEF}"
fi

#
# Messages sent from this bit of code are more detailed in nature.  It is
# basically the HMTL snippet of the monitored entity.
#
if [ ! -z "${FULL}" ]
then
```

```
        echo "${FULL} [${ACKCODE}] ${BBHOSTNAME} ${BBSVCNAME} ${BBCOLORLEVEL} ${BBALPHAMSG} " >>
/usr/tmp/full.bb
        echo "[${ACKCODE}] ${BBHOSTNAME} ${BBSVCNAME} ${BBCOLORLEVEL} ${BBALPHAMSG} " | $MAIL "!BB -
${ACKCODE} ${BBHOSTSVC} ${BBCOLORLEVEL}" "${FULL}"
fi

#
# All done go home
#
exit 0

#####################################################################
#                                                                   #
# Copyright Foobar     All rights reserved 2004.                     #
#                                                                   #
#####################################################################
```

/usr/local/bin/VulnerableScan.ksh

```
#!/usr/bin/ksh

#############################################################################
#                                                 #
# VulnerableScan.ksh - Check the system for vunerablilites, save        #
#                  configuration information to a file in the root   #
#                  directory.  In the event of a failure this will   #
#                  help us recoever the system.                    #
#                                                 #
# NOTE: Many of the ideas used here were based on thoughts and          #
#       recommedations from the book:                               #
#       Linux Security Cookbook                                   #
#       Daniel J. Barnett, Richard E. Silverman, Robert G. Byrnes       #
#       O'Reilly and Associates, Inc                              #
#       Copyright 2003                                          #
#       pages 202-282                                           #
#                                                 #
# -JBH- 04/05/2004                                          #
# usage: VulnerableScan.ksh                                     #
#                                                 #
#############################################################################

#
# Path to script we need
#
PATH=/usr/bin:/usr/sbin:/sbin:/usr/local/bin:/opt/CIS
export PATH

#
# Name of the program
#
PROGNAME=`basename ${0}`
export PROGNAME

#
# Global definitions
#
CISPATH=/opt/CIS
export CISPATH

#
# Configuration file save to disk
#
CONFIGURATION=/configuration.`uname -n`
export CONFIGURATIOn

#
# CheckEnvironment - Verify the envrionment
#
```

```
CheckEnvironment()
{
	#
	# Check for accounts with no password
	#
	echo "Checking for accounts with no passoword
	awk -F: '$2 == "" { print $1, "had no passwd!" }' /etc/shadow

	#
	# List all accounts with superuser access
	#
	echo "List of all superuser accounts"
	awk -F: '$3 == 0 { print $1, "is a superuser!" }' /etc/passwd

	#
	# Print failed login attempts since last log rotation
	#
	lastb

	#
	# List all setuid and setgid files on the system
	#
	echo List all setuidf and setguid files
	find / -xdev -type f -perm +ug=s -print

	#
	# List any regular files in the /dev directory
	#
	echo List regular files in the /dev directory
	find /dev -type f  ! -name MAKEDEV -print

	#
	# Find world-writable files
	#
	echo Find world writable files
.
	find / -xdev -perm +o=w ! \( -type d -perm +o=t \) ! -type l -print

	#
	# Find any rootkits
	#
	echo "Checking for rootkits
."
	chkrootkit

	#
	# List open files with no files descriptors
	#
	echo "Listing open files"
	lsof -L 1

	#
	# List network connections
	#
	echo "List TCP network connections"
	lsof -i TCP
	echo "\nList UDP network connections"
	lsof -i UDP
}

#
# CIS_Scan - Use the cis-scan to find the system CIS Benchmark
#	rating.
#
CIS_Scan()
{
	echo "Running the CIS scan..."
	echo cis-scan
```

```
        RATING=`grep "Final rating =" ${CISPATH}/cis-most-recent-log`
        echo $RATING

}

#
# Configuration - Gather system configuration information and
#                 save said information to a file.
#
Configuration()
{
        #
        # Remove previous configuration copy and
        # set the modes to stringent.
        #
        echo "Removing old configuration information"
        > ${CONFIGURATION}
        chown root:root ${CONFIGURATION}
        chmod 400 ${CONFIGURATION}

        #
        # Obtain and save disk information
        #
        echo "Getting disk information..."
        df -k >> ${CONFIGURATION}

        #
        # Save password file
        #
        echo "Saving password file.."
        cat /etc/passwd >> ${CONFIGURATION}

        #
        # Save group file
        #
        echo "Saving password file.."
        cat /etc/group >> ${CONFIGURATION}

        #
        # Save GRUB file
        #
        echo "Saving GRUB file.."
        cat /etc/grub.conf >> ${CONFIGURATION}

        #
        # Save kernel parameters
        #
        echo "Saving kernel configuration.."
        cat /etc/sysctl.conf >> ${CONFIGURATION}

        #
        # List acitve network services
        #
        chkconfig -list | grep :on >> ${CONFIGURATION}

        #
        # Verify rpm's database
        #
        echo "Verify the RPM database.."
        rpm -Va
}


##########################################
#
# Main code: nothing fancy call routines
# below and exit
#
##########################################
```

```
#
# Check system security rating
#
CIS_Scan

#
# Save environment information
#
CheckEnvironment

#
# Gather system configuration
#
Configuration

#
# Close shop and go home
#
exit 0

#####################################################################
#                                                                   #
# Copyright Foobar all rights reserved 2004.                        #
#                                                                   #
#####################################################################
```

Appendix: References

Internet Sources

Apache Software Foundation: “Apache HTTPD Server Version 2.0 Documentation.” URL: http://httpd.apache.org/docs-2.0/

CERT Advisory CA-2003-24 Buffer Management Vulnerability in OpenSSH” URL: http://www.cert.org/advisories/CA-2003-24.html (January 2004)

Ristic, Ivan, “ModSecurity Documentation” URL: http://modsecurity.org/documentation/modsecurity-manual-1.7.4.pdf

Gleditsch, Arne, Georg, Gjermshus, per Kristian, “Cross-Referencing Linux Linux/Documentation/networking/ip-sysctl.txt”, URL: http://lxr.linux.no/source/Documentation/networking/ip-sysctl.txt

Barnett, Ryan. Barnett, Ryan, C. “Securing Apache step by step”. http://www.cgisecurity.com/lib/ryan_barnett_gcux_practical.html.

MacGuire, Sean, Croteua, Robert, “Frequently Asked Questions”, URL: http://demo.bb4.com/bb/help/bb-faq.html

MacGuire, Sean, Croteua, Robert, “Purple Problems”, URL: http://demo.bb4.com/bb/help/purple.html

MacGuire, Sean, Croteua, Robert, ”Forbidden Problems”, URL: http://demo.bb4.com/bb/help/forbidden.html

MacGuire, Sean, Croteua, Robert, “Installation and Configuration”, URL: http://demo.bb4.com/bb/help/bb-man.html

MacGuire, Sean, Croteua, Robert, “Changing the look of BB”, URL: http://demo.bb4.com/bb/help/bb-look.html

MacGuire, Sean, Croteua, Robert, “Big Brother - Files”, URL: http://demo.bb4.com/bb/help/bb-files.html

Books

Barrett, J. Daniel, Silverman , Richard E., and Byrnes G. Robert. Linux Security Cookbook. Sabastopol, O’Reilly & Associates, 2003

Bauer Michael D. Building Secure Servers with Linux. Sabastopol, O’Reilly & Associates, 2003

Chapman, D. Brent , Zwicky, D. Elizabeth, Building Internet Firewalls. Sabastopol, O’Reilly & Associates, 1995

Cyrus Peikari,Cyrus, & Chuvakin, Anton, Security Warrior. Sabastopol, O’Reilly & Associates, 2004

Koconis, David, Murray, Jim, Purvis, Jos, Wassom, Darrin. Securing Linux A survival Guide for Linux Security Version 1.0. Washington, The SANS Institute, 2003

Hudak, Typler, Sibley, Brad. OpenSSH A Survival Guide for Secure Shell Handling Version 1.0. Washington. The SANS Institute, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.1 Issues and Vulnerabilities in Unix. Oakland, Deer Run Associates, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.2 Unix Security Tools. Oakland, Deer Run Associates, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.3 Topics in Unix Security. Oakland, Deer Run Associates, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.4 Running Unix Application Securely. Oakland, Deer Run Associates, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.5 Unix Practicum. Oakland, Deer Run Associates, 2003

Pomeranz, Hal. Track 6 – Securing Unix 6.6 Unix Security Lab. Oakland, Deer Run Associates, 2003