Global Information Assurance Certification Paper

Web Host Security Audit
Prepared for
GIAC Enterprises

GCUX Practical Assignment v1.7 (August 2001)

By: Jamie Rees

Table of Contents

Executive Summary

As part of their E-Commerce operations GIAC Enterprises maintain a variety of Internet connected servers. At GIAC's request we undertook a full-scale security evaluation of two of these servers between August 13th and 19th. Excelsior and Enterprise are webservers housing part of the company's Internet presence.

Unfortunately we have discovered a number of issues with this pair of machines. These issues range from critical – must be addressed immediately, to important – best practices and recommendations. The specific problem levels and recommended recovery actions are contained in the body of this report.

To discover these vulnerabilities we used a number of tools. We believe that the best point of view to take when conducting these studies is that of an attacker. To that end we brought with us a number of common software packages available to and in use by what has been termed the "Black Hat" computer community. Though we stopped short of causing any damage or downtime the use of these tools gives us an important view of a systems security. If we can see the security holes, so can anyone with the tools and a little wherewithal. Our thoughts echo those of Farmer and Venema when they wrote in 1993:

> Instead of merely saying that something is a problem, we will look through the eyes of a potential intruder, and show *why* it is one. We will illustrate that even seemingly harmless network services can become valuable tools in the search for weakpoints of a system, even when these services are operating exactly as they were intended to.

Another kind of possible vulnerability was found when we reviewed the configuration files of the operating system and installed applications. For this process we employed a master set of known good files as templates. Using these, with consideration for the company's customization needs, we noted our change recommendations.

This report focuses on the two servers mentioned above and most of our suggestions pertain directly to these two machines. However some network infrastructure or policy adjustments that we believe would affect the servers in question were noted and possible fixes included in this report.

Critical Issues

- The servers do not have Tripwire or similar host based intrusion detection software installed. As they are connected to the Internet in this vulnerable state important files could be changed at anytime.

- Telnet is still being used to connect to the servers. Since this protocol passes all information including passwords in plain text it is a serious security risk.

- The webservers are configured to allow CGI scripts to be run from multiple insecure directories on the servers. When written improperly these scripts can be abused in ways to allow root access to undeserving visitors.

- Passwords are currently stored on a LAN server in a Microsoft Access database. Too many people have access to read this file and can get root passwords for servers they do not support.

Important Issues

- Both machines share the same or similar root passwords. This bad procedure is carried across a number of other servers in the organization. If one of the root accounts is compromised it could lead to multiple servers being vulnerable.

- Some of the scripts written by the local administrators need to be sanitized. One example of this is not using the full paths when calling commands within the script. This can lead to attackers path hijacking the script to get modified versions of the command to run.

- Too many services are running. We noted many that appeared to not have a purpose on these servers. Our belief is that this is a symptom of having installed the operating systems will full OEM support. Unless a valid reason exists for these services to be running they should all be turned off.

These issues and others are examined and our recommendations made in the different sections of this paper: Host based issues, Network based issues and Physical Security issues.

Host Based Issues

Excelsior and Enterprise both use the same hardware platform. They are SUN 220R's with 1GB of RAM and two processors. Each contain three 36 GB drives arranged as a mirror with a hotspare. All web data is contained on external storage bricks. Both servers have Solaris 8 installed.

Two major issues with the operating system (OS) installation and configuration stood out from the moment we logged on. The first was that they were installed with full OEM support. The second was that there was no host based intrusion detection software installed. Because of the large number of unneeded services that are installed and the inability to know if any files have been changed we decided to proceed under the impression that the boxes had already been compromised.

The first step we took was to transfer some known good commands to our home directory. We recommend that for each different operating system type and version a CD of common command binaries be kept at the ready. In the case of a suspected compromise, as was the case with Excelsior and Enterprise, the commands can be moved over and the system audited.

Though, after a thorough search with our trusted commands we found no signs of a system compromise we have two very urgent recommendations. First rebuild the back up machine for each of the servers using only the core OS install. For a comprehensive set of instructions we recommend The SANS Institute: Solaris Security Step by Step V2.0 edited by Hal Pomeranz be followed. Then install all needed applications making sure to implement our suggestions for running them securely, including getting the latest versions and patches. This will go a long way to hardening the system. The second step is to install Tripwire on the machines. This will monitor the system for any changes in important configuration files and binaries. This should be done before the machines are connected to the network. We realize that the backup machines have lower hardware specs than the main machines but they should suffice for the short time while the main servers are prepared in the same manner. We believe that this is the best course of action however most of our suggestions can be applied on Excelsior and Enterprise without a re-install.

Boot PROM Settings

The security of the server can start with the EEPROM security level. To protect against PROM level commands being run indiscriminately the command *eeprom security-mode=command* can be run from inside the OS. This will prompt for a password before any PROM level commands can be run.

The same command could be run with the *=boot* option. This will stop the machine from booting up unless a password is entered. We recommend against using this command as GIAC administrators do most of their work remotely and reboots would require someone at the console to proceed. Either of these commands should be used with care. If the password is forgotten and you get stuck at the OK prompt, the only Sun officially supported recourse is installing a new PROM chip.

A related command will set monitoring of failed EEPROM logins. Running *eeprom security-#badlogins* from the command line will enable the failed login counter. This doesn't affect the login, it will not lock the prompt after a number of bad logins or anything similar.

To change the password in the future the command *eeprom security-password=* can be used. It will prompt twice for a new password, it does not ask for the old one. Note that if an attacker gets control of a server they could change this password and reboot, locking the administrator out of the server (Noordergraaf and Watson, p 5).

OS Revision and Patch Level

We used three commands to determine the OS revision and patch level. When we typed *uname –a* the output was as follows: <OS Name> <Node Name> <Release> <Version> <Kernel Architecture> <CPU Type> <Hardware Platform>. There is a newer release version of the Solaris 8 operating system. We recommend using the newer version when rebuilding the servers.

We used two commands to determine the patch level: *showrev –p and patchadd –p*. The output to both these commands was simply *no patches installed*. This shows that no patches have been applied since the OS was originally installed. As of September 1st, 2001 there are 47.2MB of patches for Solaris 8 available at SunSolve online. To view the latest patch report and download the clusters visit

sunsolve.sun.som/pub-cgi//show.pl?target=patches/patch-access

Obviously the servers need to be patched to the latest releases.

We know that GIAC Enterprises has a support contract with SUN for these machines. We urge that the regular administrators use the SunSolve access that is part of this contract to download and use the patchcheck tool. It is similar to patchdiag which they may already be familiar with except it produces output in HTML. It is actually available to anyone at this website:
sunsolve.sun.som/pub-cgi//show.pl?target=patches/patchk . However to use it to automatically download the patches missing from the servers SunSolve access is required. Note PERL 5 is required for this tool to function. PERL 5 is already installed on the webservers so this should be of little concern.

OS Configuration

Many files and directories can have an effect on the security of the server. In this section we list important configuration files and recommend settings to lock down the operating system.

/etc/passwd

This file, combined with the shadow file controls which accounts can log into the server. We discuss passwords in more detail in a later section. We wanted to note here however that any unused accounts should be deleted. For example the the print process user lp. The webservers do no printing so this account is not used and should be removed.

/etc/notrouter

This file did exist on both Enterprise and Excelsior. It is important as its presence disabled IP forwarding between the network interfaces on the public and private networks. If this file was missing it could be possible for an attacker to find a path into the management side of the network. (Nemeth et al, p309)

/etc/ftpusers

Accounts with usernames included in this file will be denied FTP access to the server (Frisch, p 249). The root account should definitely be included along with uucp, nobody, sys, bin and others. FTP is only used on the webservers to move content into place. Only select developers and the administrators need to be able to FTP to the servers. Any other accounts can be placed into ftpusers.

/etc/cron.d

This is a directory rather than a file but it is important in that it holds two files that control which users can submit crontab files to the cron process, cron.allow and cron.deny. If neither of these files exist only root will be allowed to use cron. To allow more users to use cron make a cron.allow file. In it list any users that should be able to run cron jobs. Anyone not in the list will be denied. A more relaxed method is to use the cron.deny file to list users that aren't allowed to use cron. Everyone not in the list will be allowed to set cron jobs. We suggest the cron.allow method for Enterprise and Excelsior.

/etc/default/inetinit

By default Solaris uses a TCP sequence number that is easy for attackers to guess. If they accomplish this they can hijack users sessions and gain control of the server. Enterprise and Excelsior are currently using the default setting. Entering TCP_STRONG_ISS=2 on line in the inetinit file will make Solaris use a better generation system (Noordergraaf & Watson Network p17).

/etc/home/username/.rhosts

The .rhosts file can be a dangerous convenience. They can be found in any user's home directory, including root. The file's entries include machine names and optionally usernames. A machine name in a user's .rhost file will allow that user to log on from the listed machine without any authentication. Plus signs (+) are used as wildcards so an entry of + + in ,rhosts would let any user from any machine log in.

Another similar file is hosts.equiv. Found in the /etc directory it has similar effect to .rhosts except it will accept all accounts from the machine(s) listed without prompting for authentication.

Fortunately their were no .rhosts or hosts.equiv files on the servers when we checked. In order to keep users from making them we suggest creating empty .rhost files in each user's home directory owned by root. This will not stop anyone that has root access, including attackers that come by the privelge. However it may slow them enough for someone to notice and will stop casual creation of these files.

/etc/default/init

We checked this file for the value of the CMASK variable. It was set to the Solaris 8 default of 022. This sets the umask value of files created by system

processes. In older versions of Solaris the default umask for this kind of file creation was 000 which made world writable files.

/var/adm/loginlog

This file does not exist by default but is required to monitor failed login attempts. It was made on both Enterprise and Excelsior when we checked.

/etc/inetd.conf

Most of the services that are started by inetd can be disabled or removed. This is accomplished by adding a # to the beginning of lines corresponding to the services that need to be blocked from starting or deleting the line entirely. For example fingerd can be stopped as shown in the following example:

/etc/init.d/netconfig

This is not a default file and does not exist on Enterprise and Excelsior. There are a number of NDD commands that can be put into this file which can then linked to a startup script to run at every boot up. This should be run after inetinit has run (Pomeranz, Practicum, p 125).

Shortening the ARPcache timeouts can prevent ARP attacks. Using the line *ndd –set /dev/arp arp_cleanup_interval 60000* will bring the timeout for unsolicited ARP information down to 1 minute from the default of 5 minutes. This will help defend against ARP spoofing, where one machine trys to assume the identity of another.

IP forwarding can be turned off using another NDD command. We have addressed this by using the *notrouter* file, however to be double sure we recommend adding this line into our new script: *ndd –set /dev/ip ip_forwarding 0*

The line *ndd –set /dev/ip_strict_dst_multihoming 1* will stop the systems from accepting packets on the public interface that appears destined for the private network that GIAC uses for management.

Individual timestamp requests could be a normal part of network traffic but the ability to respond to broadcast time stamp requests is not necessary and could lead to DOS attacks using up network bandwidth. *Ndd –set /dev/ip ip_respond_to_timestamp_broadcast 0* will disable this ability. If administration decides that single timestamp requests are not needed then the *ndd –set /dev/ip ip_respond_to_timestamp 0* entry should be used to turn them off.

Routers, to inform sending hosts that they should go to another router, use redirect errors. This can cause trouble as if an attacker spoofs a redirect error a host can end up trying to send information to a bogus device. To turn off this ability use the line: *ndd –set /dev/ip ip_ignore_redirect 1.* Conversely only routers should be sending these redirect errors. Enterprise and Excelsior should use the following line to disable their ability to send redirect errors: *ndd –set /dev/ip ip_send_redirects 0*

Fix-Modes Script

Casper Dik has written a tool that "fixes" the default file permissions in a Solaris install. It does this in an effort to make the OS more secure. It uses /var/sadm/install/contents to figure out what packages have been installed on the system. Therefore anything that is not installed with the pkgadd command won't be repaired. We recommend that this script is run on any newly installed machines. Care should be taken on servers that are in production. Thoroughly test on the back up server before implementing it live. The tool is available at ftp.fwi.uva.nl/pub/solaris/fix-modes.tar.gz

TripWire Intrusion Detection

We have to start this section with the note that this software should be configured after all the required applications are installed and any files that are going to be monitored are in their final state but before the system is connected to a network. However we believe that this software is key to the security of the servers and should be addressed up front.

Tripwire is available as a commercial product from www.tripwire.com. There are versions available for numerous operating systems including Windows NT/2000. Though GIAC Enterprises' other servers are outside the scope of this report it would be of benefit to investigate the use of this package company wide. A version of Tripwire remains free to download. It doesn't have the full feature set of the commercial version but it will accomplish our goal of monitoring files and directories for change (Pomeranz 2001, 6.2 p124)

After the package is downloaded and installed the administrator must decide which files and directories to monitor. Files that are not supposed to change on a regular basis are the best candidates. Leave out files like /etc/passwd and other files that will change because of daily administrative duties. The /bin and /lib directories are good candidates to watch for intruder modifications.

It is tempting to put the HTML document directories for the website under Tripwire's wing however after looking at the almost daily change schedule of GIAC Enterprises' fortune cookie saying web based catalog we believe that it would be more trouble than good except for the usually static cgi-bin directories. This is because the database that Tripwire uses to monitor for change would have to be updated every time a page in the site was modified or false alarms would be reported. Just like the boy who cried wolf, too many false alarms and the administrators will begin to ignore the reports. This of course destroys any value the tool has.

As mentioned above Tripwire depends on a database to do its comparisons with. After the administrator chooses which files to monitor they must be entered into the tw.config file. This file allows the administrator to watch for things like user and group owner change, file size changes, permission alterations with chmod as well as other modifications. It also allows the admin to choose which checksums to run against the listed files. We suggest using at least the MD5 for most of the files being watched and to leave the default listing for the more important files.

After the original database is compiled Tripwire needs to be run on a regular basis. When this happens Tripwire runs the checksum configured and compares the result with the one stored in the database. Any change to the file will result in Tripwire sending an e-mail to the administrators. We believe that this is important enough to run everyday. Most of the time this will be done with tripwire -q from Cron to send e-mail only when a change is detected. The administrators may want to occasionally run Tripwire manually when they have the time to watch the output. This is just to assure them that Tripwire is working properly and is reporting things the way it should. (Garfinkel and Spafford p285)

Another point of contention for Tripwire users is the protection of the database. If an attacker gets access to the server with enough privilege to modify some of the files that Tripwire is watching they also have the power to alter the Tripwire database. Leaving the database on the server buried in some obtuse directory may work, if the attacker happens across the Cron entry that runs the program all is lost. Obscurity may be useful to slow intruders down but it really isn't enough in this case. It is best if the original database is kept on read only media. (Anonymous, 1997 p253)

The choices vary, however in this case we suggest burning the database onto a CD and altering Tripwire to look for the database on the required device. If in the future GIAC implements a plan in which the files being monitored change on a basis that makes the burning of CD's a hindrance the idea of using a read only mounted NFS share may need to be considered.

To reiterate this software is key to the detection of a compromise and should be implemented at once. Any server on an accessible network without this protection can not be trusted.

Passwords – John The Ripper

User access to the box is protected by password. The administrators have already been doing some of the things we always suggest. Such as using a /shadow/passwd file which is the default in Solaris. Also password aging is turned on and most users are made to change their passwords every 28 days, including the root user account. This setting is found in /etc/default/passwd. In this file should be entered the following four parameters followed by an equal sign and a number value.

MAXWEEKS – time period that password is valid (already set)
MINWEEKS – time before passwords can be changed.
PASSLENGTH – minimum character length of password (set to 8 currently)
WARNWEEKS – Time before expiry that the user will start to get warnings

As noted the passlength is set to 8, we recommend that this be set to 7. This would allow for a greater choice of passwords (Garfinkel & Spafford, 1996, p. 63). Of course passwords should be a combination of upper and lower case letters, numbers, and control characters. This advice is practically as old as computing, however users never seem to get it. A written policy that the employees have to read and acknowledge would at least give the company some recourse.

The worst thing that we noted about passwords is that all the root passwords for all the servers in the organization are stored on a networked machine in a Microsoft Access database. The passwords are stored in plain text and in their original form, no algorithm is used to scramble them in any manner. Also each password is listed alongside the real hostname of the server it belongs with. This is incredibly poor security. This file, though password protected, is far too accessible. For the case of Excelsior and Enterprise only the 6 people in the webserver administration group need to know the root password but more than 60 people have access to read and print the passwords in this database. We recommend removing this database file from the network share and breaking it down into more manageable pieces. This is considered the principle of least privilege.

Another bad point on the password front is that the administrators tend to use the same or very similar passwords for their accounts. This poor practice is also in place amongst the root accounts for related groups of servers. This is disturbing because if one account is compromised there is a very high likelihood that another machine will fall victim. It is unadvisable to use similar passwords or passwords that have some kind of common theme like bird names or car models, no matter how modified they are with control characters and the like. We recommend that this practice is stopped immediately and all passwords are changed. There are over 7 quadrillion passwords that can be encrypted by 4096 different salts. That is a staggering amount of combinations, unfortunately people are still using easy to crack passwords.

We used a password-cracking tool called John the Ripper to test these servers. It is available from www.openwall.com/john. Documents can be found at this same site that will help with the installation and configuration of the package. It works by encrypting strings and comparing them to the encrypted passwords in the password file of the server. It can test against DES, MD5 and Blowfish encryption. (Pomeranz, 2001, 6.2, p.59)

For GIAC Enterprises' Solaris servers it will be DES. The strings will come from dictionaries and/or permutation lists created by the administrators. We used a master list comprised of many online dictionaries with overlapping material removed. Also we set a number of permutations to check against.

The permutation rules are set in sections of the john.ini file. There is a section for both the –wordlist and –single command arguments. Permutations can be set to try each string in the dictionary in a number of different orders and with letter to number substitutions. For example password, drowssap, p@55w0rd, etc..

Single mode tries permutations of the user's full name and username. Wordlist mode allows for a specified file to be read and the permutations checked against the password file. John the Ripper also has a test mode, -stdout. We tested the permutation rules that we used, we are leaving a copy of these rules for your use. However in the future the GIAC administrators will want to update the wordlists and perhaps add permutations. When this happens the –stdout function should be used to double-check the behaviour of your entries.

Aside from the issues related above the passwords in use were generally well formulated. There were no plain non-permutated words and no dictionary words. We did crack a few after a sustained period but all in all the passwords were strong.

SUDO

We recommend that the general practice among the GIAC administrators to log in as themselves and immediately using su to become root be stopped. Most of the access to the server is through telnet and the root account is blocked from logging in anywhere except at the console using the CONSOLE=/dev/console entry in the /etc/default/login file. This provides some accountability. The su attempts, successful or not are logged in /var/log/remote/sulog. Also the command history for the root account is available to reconcile against.

However in the case of a compromise the command history is easily modified to cover the intruders tracks. The times when multiple administrators are logged onto a server working as root makes the history file a jumble of everyone's commands. This is unacceptable when there is a clear solution that falls in line with the principle of least privilege.

More extensive use of SUDO is called for. GIAC currently uses it for a few of the commands that need to be run as a user set up to handle html log files and reporting. We suggest that it becomes the norm for using root owned scripts and commands that need superuser privileges.

Sometimes it is necessary for a person to have extended privileges in order to complete their duties. When a user is granted root access to a box there is no way to stop them from doing whatever they want. That is not to say that GIAC's administrators are untrustworthy necessarily but once again the idea is to give users only the access they require to do their jobs. If someone only needs to run a subset of commands as root then SUDO is the answer. If too many people are logged on and acting as root at the same time, tracking who did what can be a nightmare. SUDO is the answer to this as well.

One major concern is that if a SUDO user's account is compromised the intruder gets access to any commands the user is allowed to SUDO. However the benefits of using SUDO make it a worthy addition to any administrator's tools. Besides the improved accountability and restricted access mentioned benefits include:

- Users don't need to know the root passwords. In GIAC's case that means the Access database can be removed from the network share.
- User privileges can be changed without changing the root passwords.
- A single configuration file can be used on all servers, this also serves as a list of users with varying degrees of superuser rights.

SUDO 1.6.3p7 is available from www.courtesean.com/sudo/ftp.html . Once installed the configuration is straightforward. The main configuration file is called sudoers and by default is found in /etc. The following is a sample of what a sudoers file might contain.

```
# Host alias specification
Host_Alias      Ent=enterprise.giac.com
Host_Alias      Exc=excelsior.giac.com

# User alias specification
User_Alias      ADM=bjenning,smacdoug
User_Alias      TTSBS=ttsbs,hsimpson
```

```
User_Alias      YUNK=adiddy,priddyboy,jmoney,superdave,elllveee
User_Alias      WEBDOC=superdave,elllveee,adiddy,priddyboy,jmoney


# Cmnd alias specification
Cmnd_Alias      STARTADMIN=/local/apps/defcon/start-admin:\
                STOPADMIN=/local/apps/defcon/stop-admin:\
Cmnd_Alias      SAND=/usr/bin/su - sand:\
          TTSBS=/usr/bin/su - ttsbs:\
          Daily=/local/apps/ttsbs/scripts/ttsbs_0202_daily:\
          ROLLUP=/usr/bin/su - rollup:\
          SAILFTP=/usr/bin/su - sailftp

# User privilege specification
ADM     Ent=STARTADMIN,STOPADMIN
TTSBSS  Exc=SAND
YUNK    ENT=SAND,TTSBS,ROLLUP,DAILY,SAILFTP
WEBDOC  Exc=SAND,TTSBS,STARTADMIN,STOPADMIN
```

The file is broken into three sections typically. The first section defines aliases for hosts. The second command aliases and the third, permissions. The full command and host names could be written out in each of the permissions lines. However it is much more legible when the aliases are used.(Nemeth et al, 2001)

In our example the first line of the user privilege section allows bjenning and smacdoug to run /local/apps/defcon/start-admin and /local/apps/defcon/stop-admin on Enterprise. The use of separate sections makes for ease of privilege change. It is easy to add or remove a username from the User_Alias lines on a central copy of this file which is then propagated onto other servers using rdist or sync.

After the sudoers file has been prepared it is consulted each time a user runs a command with sudo. For example if jmoney enters the following command on Enterprise: *sudo /local/apps/ttsbs/scripts/ttsbs_0202_daily* the sudo command prompts for a password. This is the users own password, if the password is given correctly sudo will refer to the sudoers file and see that jmoney, as part of YUNK, is allowed to run the command and then proceed to run the command. If the same user logged into Excelsior and tried the same command it would fail. According to the sudoers file members of YUNK are only allowed to run commands on Enterprise.

Security for Sudo is password based. A user's password is entered when sudo is first invoked. There is a time limit of five minutes within which a user can enter another command without having to supply the password. Each time a new command is entered the timestamp is updated and the five minute countdown begins again. This serves as protection against the user leaving their console and someone else getting sudo access. Also in newer versions the timeout is tied to the user and the session that originated the sudo request. This stops malicious users on other terminals from hijacking the sudo privileges, something that was possible under older releases.

We believe that Sudo's advantages far outweigh any issues that arise in its use. For this reason we reiterate it's importance and urge its implementation.

SUID/SGID Problems

Occasionally to help someone perform their duties a program will be set with special permissions that allow it to be run in the context of the user that owns the file. As shown by the lowercase s where the x usually is in the user's permissions as ls –l shows.

rw**s**r-xr-x 1 root sys 13462 Nov 8 /Max

For example the passwd program can be used by any user to change his or her password. It needs root level privilege to change the /etc/passwd file. It is set as a suid program so that when run it can edit the needed file and allow users to change their passwords.(Nemeth et al, p 38)

A similar permission is SGID which runs the program as a member of the group that owns the file. It also can be used on a directory to make all files created in that directory automatically belong to the specified group instead of the group that the creating user belongs to. It is denoted by a lowercase s replacing the x in the group portion of the permissions as shown below:

drwxrw**s**r-x 2 willy wonka 512 Nov 8 /Peanut

These are obviously needed functions but if mistreated and used haphazardly it can become a serious security problem. A specific example of this type file in use on GIAC's systems is the log administration tool, ladmin. To work properly ladmin needs to be run in the context of the user of the same name. An ls –al of the file shows the following:

-rwsr-xr-x 1 ladmin stats 451 Nov 22 2000 /data/stats/ladmin

This particular file isn't too worrisome. More concerning are files that are SUID root. To find which of these exist we ran /usr/bin/find / -type f -user root –perm -4000 –print which generatedthe following list:

/usr/openwin/lib/mkcookie
/usr/openwin/bin/xlock
/usr/openwin/bin/ff.core
/usr/openwin/bin/kcms_configure
/usr/openwin/bin/kcms_calibrate
/usr/openwin/bin/sys-suspend
/usr/lib/lp/bin/netpr
/usr/lib/fs/ufs/quota
/usr/lib/fs/ufs/ufsdump
/usr/lib/fs/ufs/ufsrestore
/usr/lib/pt_chmod
/usr/lib/sendmail
/usr/lib/utmp_update
/usr/lib/acct/accton
/usr/dt/bin/dtaction
/usr/dt/bin/dtappgather
/usr/dt/bin/sdtcm_convert
/usr/dt/bin/dtprintinfo
/usr/dt/bin/dtsession
/usr/bin/at
/usr/bin/atq
/usr/bin/atrm
/usr/bin/crontab
/usr/bin/eject
/usr/bin/fdformat
/usr/bin/login

/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/ps
/usr/bin/rcp
/usr/bin/rdist
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/su
/usr/bin/uptime
/usr/bin/w
/usr/bin/yppasswd
/usr/bin/admintool
/usr/bin/ct
/usr/bin/chkey
/usr/bin/nispasswd
/usr/bin/cancel
/usr/bin/lp
/usr/bin/lpset
/usr/bin/lpstat
/usr/bin/volcheck
/usr/bin/volrmmount
/usr/sbin/allocate
/usr/sbin/mkdevalloc
/usr/sbin/mkdevmaps
/usr/sbin/ping
/usr/sbin/sacadm
/usr/sbin/whodo
/usr/sbin/deallocate
/usr/sbin/list_devices
/usr/sbin/afbconfig
/usr/sbin/ffbconfig
/usr/sbin/m64config
/usr/sbin/lpmove
/usr/sbin/pmconfig
/usr/sbin/static/rcp
/usr/ucb/ps
/proc/295/object/a.out

These binaries are all default SUID files on Solaris. As these systems have few users and the administrators can have needed files added to the SUDO implementation, many, if not most, of them could be disabled by changing the permissions to 555. Unneeded files could be removed completely.

The good news is that there were no extra SUID files on the systems. Also there was no sign that any of the default ones had been compromised. However without Tripwire it will be hard to monitor all of these files for changes.

Logging

All the logging for Excelsior and Enterprise is done locally. If a box is compromised the logs are easily accessible to the attacker to modify. We suggest that the logging be done to a centralized server. This server should only be accessible by the logging machines and the administrator's network segment. This can be accomplished by the use of a firewall product such as Ipfilter or IPTable.

Currently the webservers are logging to /var/log as shown by the /etc/syslog.conf entries below. Entries can be change to point copies of the logs to a more secure location. The entry under "proposed settings" below shows an example of how each log should be set up. Having double entries and changing one to point to @logserver makes the machine log to its normal local spot and to a server named logserver out on the network.

```
Current settings:

*.err;kern.notice;auth.notice                   /dev/console
*.crit;kern.err;daemon.err                      root
*.emerg                                         *
```

```
kern.debug                                          /var/log/kernlog
user.info                                           /var/log/userlog
mail.info                                           /var/log/maillog
daemon.info                                         /var/log/daemonlog
auth.info                                           /var/log/authlog
news.info                                           /var/log/newslog
uucp.info                                           /var/log/uucplog
syslog.info                                         /var/log/syslog
```

Proposed Settings:

```
*.err;kern.notice;auth.notice                       /dev/console
*.err;kern.notice;auth.notice                       @logserver
```

In our example for proposed changes all error messages, notice level messages for the kernel and authentication would be displayed on the console as well as logged on the remote server. This should be repeated for each entry in the file. There is the only way to get two actions (a local and remote copy) for any given set of facilities and levels. (Nemeth et al, p213) Keeping a local copy is important as if the remote logging host is down the logs would be lost for ever. Notice also that we used @logserver in the file. This can be the name or IP of any server in the company. It could also say @loghost as long as the $LOGHOST variable was defined as the remote server in the /etc/hosts file.

Currently all the OS logs are kept for three months at a time and then just disposed of. We suggest that the logs be rotated out and stored on CDR media much like the webserver logging method describe below. The ability to trace through older logs would benefit the company in the event of a compromise.

The webserver logs to the directory /var/log/www/. There is an access log and error log maintained here. This setting comes from the Apache httpd.conf file's directives shown below.

```
TransferLog /var/log/www/access-www00_log
ErrorLog /var/log/wwwd/error-www00_log
```

These logs are rotated nightly and stored in /var/log/movewww/ as compressed files. For example 10162001access-www00_log.Z. These logs are collected by a dedicated log processing server where they can be used for trending and capacity planning. Six months of logs are stored on the processing server at any given time. Periodically all the older logs are copied to CDRW and then stored offsite. This makes it easier to find the logs if an issue arises.

Network Issues

The network supporting the two servers we investigated is rather straightforward. The webservers are connected to a network using Internet routable IP addresses. They are cordoned off from the Internet by a firewall called Guardian245. All traffic from the Internet passes through this chokepoint.

The servers are also connected to a private network using addresses in the 192.168.0.0 non-routable range. These interfaces are in use by the administrators for management work. This network is only accessible from the administrator's segment of the internal network and by the address pool assigned to administrators when they use the VPN to connect from outside. No other users can access the server except through the firewall on specific ports. See Figure 1 for a simplified view of this arrangement.

WU-FTP

Normally recommend that the FTP server be loaded on a separate staging machine than the servers running the webserver software. Because of budget constraints GIAC is unable to comply with this suggestion at this time. We believe that as hardware is upgraded or some machine becomes available in the future the current approach (running both on the same server) should be reconsidered and a staging server set up.

The latest version of WU-FTPD is 2.6.1, it is available from ftp.wu-ftpd.org During the process of rebuilding the servers should be sure to get the most current release, as there is a vulnerability in 2.6.0 and earlier. GIAC is currently using an older version on Enterprise and Excelsior. See www.cert.org/advisories/CA-2000-13.html for more information on the trouble.

The FTP access for the servers is for administrators and some content developers to load files into the html directory. We recommend that the FTP root be changed so that it no longer overlaps with the HTML root. This would mean that an administrator would either have to move the files manually or write an automated routine to move them. This protects against an attacker using an FTP exploit to get access to and changing HTML content.

Currently FTP is running all the time and the firewall is the only line of defense against would be attackers getting an FTP login prompt. As access to the FTP site is for admin and developers only and considering the fact that these groups work on certain network segments we believe that WU-FTPD should be run from inetd and use the TCP Wrappers program to allow connections only from the associated networks. This would add some depth to the security around the FTP server, both guarding against a firewall breach and locking out curious insiders that may be poking around. See the section TCPWrappers for more information and some examples of the inetd.conf entries.

Apache

Both Enterprise and Excelsior have Apache installed. Also both were recently upgraded to the newest versions of the software. Currently version 1.3.20 is available from httpd.apache.org/dist/httpd/. This was a good move as there won't be any more releases for the older 1.2 code that GIAC had been using.

As this is a public facing website, anonymous users have to be allowed access to at least some of the main pages. However configuration changes can be made to control what parts of the filesystem they have access to curtail their ability to cause trouble. For example the following entry stops access to the entire filesystem by the webserver.

<Directory />

```
        Order Deny, Allow
        Deny from all
    </Directory>
```

Now this will cause problems with people trying to visit the website as it blocks access to the directories where the content is stored. In GIAC's case that directory is /data/httpd/htdocs. To allow access to this directory another entry can be made like this:

```
    <Directory /data/httpd/htdocs>
        Order Deny, Allow
        Allow from all
    </Directory>
```

This can be repeated for any directory that visitors may require access to. In this manner a tighter control of who goes where is obtained. If a visitor tries to trick the webserver into displaying content outside the content directories they will be stopped.

Currently the AllowOverride directive is set to allow all overrides. This is bad both from a security and performance perspectives. Every page that is called on the webserver won't be displayed until an .htaccess files is searched for. It doesn't just search in the directory that contains the page being visited but all the way up to the root directory of the server. (Wainwright, 1999, p105)

We suggest setting AllowOverrides to none for the root directory and if absolutely necessary turning on just the required overrides for certain directories. Where GIAC is a closed development area and does not allow just anyone to put content onto the webservers, allowing some overrides should be acceptable. However the administrators still have to be aware of the risks involved. .htaccess file use can lead to unnoticeable changes to server security. Having all the settings that could be abused in an .htaccess file centralized in the httpd.conf where they can be changed by very few people is generally the best idea.

We noted some good settings such as having directory indexing turned off and the use of IncludesNOEXEC to stop scripts from being included in webpages. However until the ability to override these settings is turned off they can't be trusted to be in effect. In fact they might as well be turned on in httpd.conf for the amount of control that is being given away.

Another flaw we found in the setup of Apache on these servers was the ability to install and run CGI scripts from anywhere. GIAC's website has grown rapidly as business in fortune cookie sayings has taken off. Many scripts are required to keep the site functioning and offering the customer the best possible experience. A bad habit of allowing the reckless placements of scripts has evolved from this frantic pace.

Luckily it is a habit that can be broken. We recommend a policy be put in place that requires the developers pass their scripts to the administrator's for placement in a single directory on each server. This will help on a number of fronts.

Downtime spent searching for where a script is hidden in the maze of directories can be better spent working on problem solutions. Two sets of eyes seeing a script before it is put in place will cut down on the number of potential security vulnerabilities. The server will be less likely to be compromised by an attacker who places a script into any but the designated directory. Since this directory will be made outside the html content directories it will be quite unreachable in this manner.

Entries in httpd.conf can enforce the policy, as once set, no scripts outside the designated area will run, because of this much care has to be taken to avoid missing a script when turning these directives on. It will be easier to implement if our advice to rebuild the boxes is followed.

Currently the Option ExecCGI is turned on globally (and if it wasn't it could be overridden with .htaccess as discussed above). This needs to be turned off as it, combined with the set handler directive, allows all files ending in .cgi to be run by web visitors, no matter where on the server they are.

Turning ExecCGI setting off stops all scripts from running, so we have to have a alternative place where scripts will run from. We recommend this line be entered in httpd.conf :

```
ScriptAlias      /cgi-bin/  "/usr/local/scripts/cgi/"
```

This allows the developers to call a cgi by using /cgi-bin/scriptname. When the webserver is passed /cgi/bin it looks in the otherwise hidden /usr/local/scripts/cgi/ for the script and runs it if it is there. Access to this directory should be restricted, keeping in mind that the webserver user has to have the ability to run the scripts. The following entry will help by disabling the use .htaccess and turning of all options.

```
<Directory  "/usr/local/scripts/cgi/">
        AllowOverride None
        Options None
        Order allow,deny
        Allow from all
</Directory>
```

Another issue surrounding the scripts on the server is the need for tightening or sanitizing the code. For example one of the shell scripts we looked at was used to take files from /var/log/www/, move them to /var/log/movewww/ and compress them. The script runs the same commands as an administrator would when doing this task manually. It is SUID to the nobody user that owns the logs.

```
TODAY=`date +%Y%m%d`
mv /var/log/www/access-www00_log  /var/log/movewww/$TODAYaccess-www00_log
compress *access-www00_log
```

This seems harmless and does do the job it was meant to. However by just using the binary name and depending on the path variable to make sure the proper version is run is a bad habit. $PATH could be altered to include a directory containing compromised versions of mv or compress. By changing the line calling move and compress to include the full paths to the binaries to be used we can be sure that the script is running trusted files that we can monitor for change. An alternative method is to define $PATH in the script itself.

As sections of the GIAC websites are used for the selling of online fortune cookie sayings SSL is a requirement. We didn't see anything outstanding within the configuration of the SSL for GIAC's sites. However as SSL adds overhead to each page

served it may be advisable to segregate the pages requiring SSL from those that would be fine with regular http traffic. The SSLEngine only works at the server level so SSL can not be turned off and on for subdirectories. In the future, as the site grows, it may be necessary to run a separate instance just for the section of the site that requires SSL.

For now the use of SSL for different directories can be enforced, not turned off, using the SSLRequireSSL directive. If a non-SSL connection is attempted to a directory protected by this directive access will be denied. It can be used in Directory, Location or even .htaccess files, similar to the following.

```
<Directory /data/httpd/htdocs/secure>
        SSLRequireSSl
</Directory>
```

All credit card and other customer information is stored in an encrypted format on a separate database server. There is no file system level encryption on the webservers. This is of minimal concern as no sensitive data is kept on the machines.

Apache has proven to be a very secure and reliable webserver. However subscriptions to mailing lists and online newsletters are required. That and regular trips to www.apache.org will aid in keeping abreast of the latest developments.

SendMail

Some of the webpages served from Enterprise and Excelsior use CGI scripts to gather user input. These scripts also generate an e-mail that is sent to the user. Currently the sendmail daemon is running on Enterprise. It does not need to be.

Normally mail will be sent automatically when composed. For the times that it gets queued for later sending a cron entry will do the task hourly without running the security risk of having sendmail running all the time (Noordergraaf and Watson, p 28). The entry should look like this:

```
0 * * * *  /usr/lib/sendmail -q
```

SSH

Currently the administrators use telnet to remotely manage the servers. Though this is a very functional manner of doing things it is very insecure. With telnet all traffic, including username and password is sent in plain text.

Also, as mentioned in a later section of this paper, backups are currently done across the network in the open. This activity takes place on the private management plane however it is still vulnerable to eavesdropping. If any machine that has an interface on the private network is compromised, backup information from Enterprise and Excelsior is at risk.

The Secure Shell software package can resolve both of these issues. Even in its lowest secure form, using userids and passwords SSH is an improvement because of the encryption of all information. This would obscure the information supplied by both the administrators and the automated dump process that performs the backups.

It would be more secure to use private/public key arrangement. This takes a little more work to set up and then maintain as each client's public keys must be generated and

then copied to each server they need to access. Every time the client updates a key, the copy in each server's authorized_keys file needs to be updated. We suggest using the public authentication method as it is the more secure of the two.

OpenSSH for Solaris is available from http://www.openssh.com/portable.html It is free to download though we recommend that some donation be made to help further the development of the tools. The newest version as of September 25, 2001 is 2.9.9 which offers support for both the SSH 1 and SSH 2 protocols.

Two options in the configuration file for the building of OpenSSH to look for are:
--with-ssl-dir=/path/to/the/OpenSSL/library
--with-tcp_wrappers
The first tells the configure script where to find the required OpenSSL library. It has to be installed before OpenSSH is started. The second is to allow OpenSSH to link with TCP_Wrappers, which we recommend be installed, for the reasons found in the TCP_wrappers section of this work. It is recommended that a link be made in the directory where the configure script will run that points to tcpd.h. (Acheson, 2001, p30)

The program is configured using the /etc/openssh/sshd_config file. This file controls what type of authentication is used for SSH. It supports the use of .rhosts and host.equiv. We recommend that this support be removed by making sure the following lines are in the configuration file.

```
IgnoreRhosts    yes (this is default in OpenSSH)
RhostsAuthentication  no (this is a default setting)
RhostsRSAAuthentication     no
```

To allow for the most seamless transition to those used to using telnet access the next line should be added, it allows access by supplying a username/password combination similar to what is currently done.

```
PasswordAuthentication      yes
PermitEmptyPasswords        no
```

After the software is installed and configured users will gain access by calling the ssh command like this: ssh –l jmoney enterprise.giacookies.tld. Now instead of passing everything in the open, prone to network sniffing, it will be encrypted. If the public key model is used the authentication is a little more robust. When a user requests a logon the server will encrypt a challenge using the user's public key. Only a client with the correct private key will be able to decipher the challenge and answer it. This gives a satisfactory assurance that the users are who we think.

NTP

The Network Time Protocol is used to keep all server system clocks in sync with one another. This does not make the server any more secure by itself. The result of synchronized system clocks is easier correlation of log entries in the case of compromise. For example Enterprise and Excelsior's clocks were about 3.5 minutes different when we

checked. If we notice something suspicious in Enterprise's logs and wanted to check on Excelsior to see if the same thing was occurring by default we would look at the entries with around the same time stamp.

Most administrators would scan through more than just the entries for that exact time and would probably find the appropriate lines. However when it comes to building a criminal case against intruders, more exact time may be required. When involving many machines and tracing the path from one to another looking for how the attacker got in standardized time is a necessity. If you can't say without a doubt where an attacker was at any given time you may very well lose any chance of catching or prosecuting him or her.

Normally we would like to see at least three machines time synching with Internet time servers. GIAC does not have the resources at this time to conform to this request. GIAC does however have two DNS servers that connect to the Internet and that can be set as NTP peers within the network.

It is important to use both of these servers as having only one time source can allow attackers to skew the time on the network. Throwing off log timestamps or worse extending the privilege of time based security measures.

Each of the two DNS servers should point to three separate outside time sources. Getting time from six separate sources will result in the most reliable information. Each of the three sources for each server may report a slightly different time. The NTP software will calculate an error threshold and use the point where the most overlap occurs as the correct time. If one of the sources reports a time far out of the threshold it is ignored.(Pomeranz, 6.4, p73)

A list of available time server can be found at www.eecis.udel.edu/~mills/ntp/servers.htm. Each server's administration has set different access policies, make sure to follow any instructions provided when choosing time servers. The ntp.conf file is where the servers are configured to use outside sources and to peer with one another. Very simple edits to the file are required, in a format such as:

```
server 128.100.103.252      #tick.utoronto.ca
server  209.87.233.53       #clock.chu.nrc.ca
server  142.3.100.15        #timelord.uregina.ca

peer    xxx.xxx.xxx.xxx     #IP for the other DNS server
```

In our example the server will synch time with three Stratum 2 timeservers. The term stratum defines where a server is in relation to a reliable timesource. So GIAC's servers will be Stratum 3 timeservers in this scenario. This is fine until access to these servers is dropped because of an issue with the Gaurdian245 firewall.

If that were to occur than the servers would become Stratum 16 or disconnected. In other words their time signal would no longer be reliable. Fortunately there is a way to combat this. NTP allows for servers to be set up as pseudo-clocks. These pseudo-clocks will not drop to Stratum 16 when disconnected, instead they will continue to operate at the stratum that is defined in ntp.conf. An entry like the one below defines a pseudo-clock.

```
server 127.127.1.1          #the first three octets tell NTP to load the psuedo-
                            #clock driver, the last octet signifies that it is the
                            #first instance of the device.
fakeclock 127.127.1.1   stratum 5
```

The pseudo-clock level must not be the same level or higher than the stratum that the server normally operates at or it will interfere. It should be set a few levels lower than the normal setting so that there is some room for fluctuation.

After the DNS servers are setup the rest of GIAC's server should be configured to get the time. Each should point at both the DNS servers for redundancy and reliability purposes. The same type of entries as above are made in the ntp.conf file of these servers. The pseudo-clock settings can be left off at this level.

TCPWrappers

The systems depend on the firewall to keep unauthorized visitors away. This is fine until the firewall or some other internal server is compromised. We recommend that both Excelsior and Enterprise have TCPWrappers installed.

As mentioned previously the administrators and select developers need FTP access to the server and they do this from specific sets of IP addresses. Also the administrators use Telnet for remote management of the servers, again from a select set of IP addresses. TCPWrappers can be configured so that only certain hosts or network segments can connect on designated ports. This is exactly what is required in GIAC's case.

The software is free and can be downloaded from ftp.porcupine.prg/pub/security/ Because Enterprise and Excelsior are running Solaris 8 the ipv6 version of the software must be used. (Pomeranz, 2001, 6.2, p 95)

When installed it sits between inetd and the internet service daemons that run from it. The three main files involved with the configuration of TCPWrappers are /etc/inetd.conf, /etc/hosts.allow, and /etc/hosts.deny.

The inetd.conf is usually set up to start services when a request arrives from a remote host. The configuration is modified as shown below to first call TCPD, which in turn will start the service.

Without Wrappers:

```
ftp     stream  tcp  nowait  root  /usr/sbin/ftpd     ftpd
telnet  stream  tcp  nowait  root  /usr/sbin/telnetd  telnetd
```

With Wrappers

```
ftp     stream  tcp  nowait  root  /usr/sbin/tcpd  ftpd
telnet  stream  tcp  nowait  root  /usr/sbin/tcpd  telnetd
```

The above configuration makes TCP Wrappers run when an attempt to connect with FTP or Telnet happens at the server. Tcpd consults hosts.allow and hosts.deny, in that order, to determine if the connection is permitted. If a line in hosts.allow matches the request the service is started and the connection allowed, hosts.deny is never checked. As well only the first matching line in the file is used. (Frisch, 1995, p.625) Below are some sample lines from both files:

hosts.allow

```
fingerd     : bob dot enzo fong
telnetd     : 192.168.10
ftpd        :  192.168.10,  id9763 id4573 id6782
```

The first line allows fingerd to be run by the machines with the hostnames bob, dot, enzo, and fong. The second allows telnetd connections from the administrators network, the last octect doesn't need to be included, any machine with an IP that matches the first three octets will be allowed. The third line allows ftpd from both the administrators network and some hosts belonging to developers (as they use laptops the hostnames will be the same if they are working at home or their desk at work).

Service names can be put on the same line separated by a comma. In this case because we want only the administrators to use telnet we keep everything on separate lines. We suggest simple configuration the hosts.deny, as follows:

hosts.deny

```
ALL   :     ALL
```

This stops everything from connecting, except what is implicitly allowed in the hosts.allow. If someone tried to TFTP to this server the connection would not match anything in the hosts.allow and so would be passed to the hosts.deny. It would match the only line inside and get dropped.

Another feature of the hosts.deny that may be of use to an administrator is the ability to run a command when someone tries to access a certain service. An interesting use of this feature is to have the software mail the administrator details about the attempt. The author of the software, Wietse Venema, refers to this as "bugging" the service. It can be set up as follows to send the remotes host's name and the daemon they attempted to connect to:

```
telnetd  :  ALL  (/usr/bin/mail –s %d %h admin@giaccookies.com)
```

TCP Wrappers can be installed on the machines whether they are rebuilt or not. It is a very important part of an overall security plan. Working in conjunction with the firewall it will greatly improve the defense of the servers.

Nessus & NMAP Scan

We used the Nmap tool installed on a laptop hooked into the network outside of the Gaurdian245 firewall to scan Enterprise and Excelsior. We chose this viewpoint to closely mimic what an attacker would see. We found a small number of ports open/services running that should not be. Any service running is a target for attackers.

We advise that any extraneous services be turned off to give would be intruders less opportunity to strike.

Nmap is available from www.insecure.org/nmap. It is a free tool that is easy to setup and use. For example Linux users can use the RPM process to auto-install the software. The output of the completed scan is included at the end of this document in Appendix A. It provides a listing of the ports that are open.

We also ran a scan for vulnerabilities using the Nessus tool available from www.nessus.org/download.html. Nessus uses a list of plugins to determine which vulnerabilities to test for. We used the setting that turns off any of the tests that could bring the box down or interfere with clients accessing the site. To save time we also disabled any plugins that tested for vulnerabilities associated with Windows servers. To make sure we had the very latest set of exploits to test for we used the *nessus-update-plugins script*.

The servers faired quite well with only one hole discovered. GIAC is using an old version of WWWCount or count.cgi. This tool, used to track visits to some of GIAC's WebPages, has a vulnerability that can be used to run arbitrary commands on the server. This is limited to acting as the Apache user that runs the script, fortunately the nobody user has little other permissions. However there is a newer version of WWWCount available at www.muquit.com.muquit/software/Count/Count.html. We recommend that the software be upgraded.

Nessus did find some items it lists as security warnings and notes. They are contained in the complete Nessus report included at the end of this document under Appendix A. Each of them is prioritized and linked to solutions. Two we found disturbing were 21 and 23, FTP and TELNET respectively. Since there is no need to access these services from outside the firewall we were surprised to see responses from these ports. This could lead to an attacker brute forcing a password and gaining access to the server/directories where the site content is stored. None of the administrative staff we talked with had any knowledge of how and why the ports were opened in the firewall. TCP Wrappers configured as we suggested earlier and replacing the firewall rules to block access will plug this rather nasty hole.

Both of these tools are easily installed on a portable computer or desktop machine running Unix or Linux. We believe that any new machines being put on the network should be subjected to the scans of these tools set to their most intrusive levels. This would prove that the machines have been hardened against most known exploits. We recommend that Nessus be updated with the latest plugins and run against the servers on a regular basis, perhaps during planned maintenance windows.

Physical Security

Server Room

The GIAC Enterprises server room is large and contains many servers. It is impressive in size and scope. Unfortunately this is also a security downfall. Having all the servers in one large room is convenient but allows access to the machines to people that don't need it. Webservers sit right beside e-mail servers and fortune cookie saying generator machines, all in unlocked cabinets. Once again this bucks against the principle

of least privilege. These sets of servers all have separate groups of administrators, none of which work on all the servers.

At the very least the cabinets should be locked and keys only distributed to the specific groups of administrators that require access. More extreme but even more secure would be the subdivision of the server room into multiple smaller rooms or cages. This may not be feasible as the raised floor would lead under the cages and allow access into the different sections, though access to the main server area would be difficult.

The raised floor in the room as it is now is not an issue. Though the room is large, thick concrete walls with no windows surround it. The raised floor sits on a concrete slab that meets up with the outside walls so there is no way into the room except by door. The doors are electronically locked and require keypad access and a card swipe to enter.

This double authentication using a token and a secret is good security but occasionally somebody will draft in behind someone else that opens the door. This doesn't happen much but we noticed it happen while we were on premises and it would only take one entry for a miscreant to wreak havoc. A written and enforced policy stating that when a person opens the door they are responsible for anyone that enters with them would make administrators question the presence of people they don't recognize.

Fire Suppression

The fire suppression system in the server room is lacking. There are no individual fire extinguishers in the server room. Everything is dependant on water based sprinklers suspended from the ceiling. The system is not hooked up in a way that will cut the power before the water is released. This means that electricity could be on when the water falls which could destroy the machines in a fashion almost as final as fire.

An easy improvement would be the inclusion of a number of small fire extinguishers that can be used for electrical fires. Changing out the water based fire sprinklers will be much harder. GIAC does not own the building that houses the offices and servers. All major changes have to be done through the complex owners. They have shown resistance to changes of this nature in the past. Also financially, installing a new fire suppression system is hard. GIAC would be responsible for the cost of the new system and though there are benefits the cost ratio is high enough to deter the purchase.

Backups

Both Enterprise and Excelsior are fully backed up every night. The backups are done to a machine across the private network segment. Normally we would suggest a mixture of full backups and differential backups, however GIAC has a well established method for backing up these servers already in place and we feel changes would be unwarranted.

Something we do suggest be changed is the fact that these backups are done unencrypted. Even though the network that the backups run on is private and not

reachable from the Internet by conventional means the information is still vulnerable. If any server that has a connection to the private network is compromised it could be possible for the attacker to capture the backup information in transit. See the SSH section for more information.

Reccomended Fixes

Our main recommendation is the complete rebuilding of the passive standby machines. Since these machines are now just sitting, waiting to be connected to the network in the event of a problem with the live servers they will not be missed. When completed the backup servers could in turn take the place of the live servers so that the rebuild could be done on them as well. If this is can not be done because of monetary or time constraints the other recommendations can still be implemented. The list below gives our opinion on the order in which these changes should be implemented.

- Install and configure Tripwire to monitor important configuration files for signs of tampering.

- Install the Secure Shell software. Stop the administrators from using telnet to access the machines. Sending passwords in the clear is a bad idea.

- Enforce a policy to make administrators use different passwords on different machines. On some servers the same or very similar passwords are used multiple times.

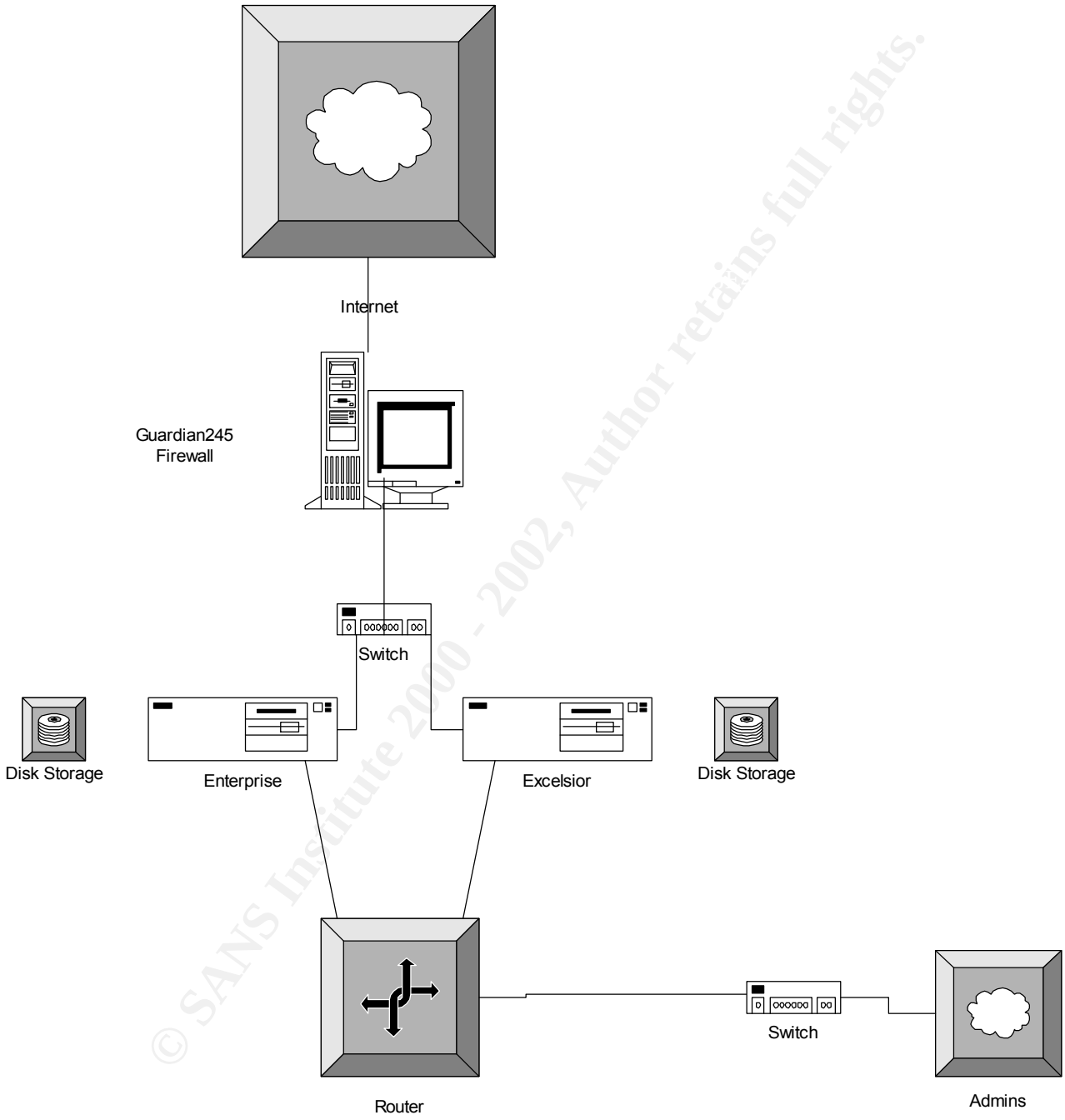
- Turn off the FTP and Telnet services. These can be used as targets by attackers looking for a way to compromise the servers. If FTP is required then TCPWrappers should be used to filter connections. Telnet should be replaced with SSH.

- As described in the Apache section of this document, CGI scripts should not be allowed to run from anywhere but the designated cgi-bin directory.

- Remove the passwords from the online Microsoft Access database. Too many people have access to the passwords for machines they don't administer. For example the firewall administrators can see the root passwords for the web servers.

- Audit scripts in use on the servers. Look for relative paths being used to call commands or pointing to files that will be modified. These can be used to point to trojaned files or unintended configuration files.

Figure 1 Network Overview



Appendix A

NMAP Results

Nmap (V. nmap) scan initiated 2.53 as: nmap -sS -O -v -T polite -oN /home/jtrees/nmapscan.txt 207.179.176.217

Interesting ports on www.giacookies.com (10.10.10.10):
(The 1517 ports scanned but not shown below are in state: closed)

Port	State	Service
21/tcp	open	ftp
23/tcp	open	telnet
80/tcp	open	http
443/tcp	open	https

TCP Sequence Prediction: Class=truly random

Difficulty=9999999 (Good luck!)

Sequence numbers: AD294CBB D6006D9E 86188F6C 756E2F5F 17657D2D 23CA8313

No OS matches for host (If you know what OS is running on it, see http://www.insecure.org/cgi-bin/nmap-submit.cgi).

TCP/IP fingerprint:

TSeq(Class=TR)

T1(Resp=Y%DF=Y%W=2297%ACK=S++%Flags=AS%Ops=NNTNWME)

T2(Resp=N)

T3(Resp=N)

T4(Resp=Y%DF=Y%W=0%ACK=O%Flags=R%Ops=)

T5(Resp=Y%DF=N%W=0%ACK=S++%Flags=AR%Ops=)

T6(Resp=Y%DF=N%W=0%ACK=O%Flags=R%Ops=)

T7(Resp=N)
PU(Resp=N)

Nmap run completed at Wed Nov 14 14:10:09 2001 -- 1 IP address (1 host up) scanned in 665 seconds

Nessus Results

List of open ports :

- *ftp (21/tcp) (Security notes found)*
- *telnet (23/tcp) (Security warnings found)*
- *http (80/tcp) (Security hole found)*
- *https (443/tcp)*
- *general/udp (Security notes found)*
- *general/tcp (Security notes found)*

[back to the list of ports]

Information found on port ftp (21/tcp)

Remote FTP server banner :
-##

220-

220- This system is restricted to authorized personnel. Only authorized

220- work is to be done. Use of this system is an agreement to monitoring.

220- Any unauthorized use is subject to disciplinary action.

220-##

Warning found on port telnet (23/tcp)

The Telnet service is running.
This service is dangerous in the sense that
it is not ciphered - that is, everyone can sniff
the data that passes between the telnet client
and the telnet server. This includes logins
and passwords.

You should disable this service and use OpenSSH instead.
(www.openssh.com)

Solution : Comment out the 'telnet' line in /etc/inetd.conf.

Risk factor : Low
CVE : CAN-1999-0619

Information found on port telnet (23/tcp)

Remote telnet banner :
##

This system is restricted to authorized personnel. Only authorized

work is to be done. Use of this system is an agreement to monitoring.

Any unauthorized use is subject to disciplinary action.

##

Vulnerability found on port http (80/tcp)

The 'Count.cgi' cgi is installed. This CGI has
a well known security flaw that lets anyone execute arbitrary
commands with the privileges of the http daemon (root or nobody).

Solution : remove it from /cgi-bin.

Risk factor : Serious
CVE : CVE-1999-0021

Information found on port http (80/tcp)

The remote web server type is :

–

Information found on port general/udp

For your information, here is the traceroute to 207.179.176.217 :
142.134.174.1
?

Information found on port general/tcp

QueSO has found out that the remote host OS is
* Solaris 2.x

CVE : CAN-1999-0454

References

Anonymous, Maximum Security: A Hacker's Guide to Protecting Your Internet Site and Network. Indianapolis, Sams.net, 1997

Acheson, Steve, SANS Securing Unix 6.3 – Topics in Unix Security, Fredericksburg, The SANS Institute, 2001

Farmer, D.; Wietse, V., Improving the Security of Your Site by Breaking Into it, 1993 URL:http://www.fish.com/security/admin-guide-to-cracking.html

Frisch, Æleen Essential System Administration, 2nd ed, Sebastopol, O'Reilly & Associates Inc., 1995

Garfinkel, S & Spafford, G., Practical UNIX & Internet Security. Sebastopol, O'Reilly & Associates Inc., 1996

Nemeth, E. et al, UNIX System Administration Handbook, 3rd ed, Upper Saddle River, 2001

Noordergraaf, A. and Watson K., Solaris Operating Environment Security, Palo Alto, 2000

Solaris Operating Environment Network Settings for Security, Palo Alto, 1999

Pomeranz, H. SANS Securing UNIX 6.2 - Topics in Unix Security. Fredericksburg The SANS Institute, 2001

SANS Securing Unix 6.4 – Running Unix Applications Securely, Fredericksburg, The SANS Institute, 2001

SANS Securing 6.5 - Solaris Practicum, Fredericksburg, the SANS Institute, 2001

Wainwright, Peter, Professional Apache, Birmingham, UK, Wrox Press, 1999