Global Information Assurance Certification Paper

Auditing FreeBSD

GSNA PRACTICAL
version 1.0, 2001
for SANS Parliament Square

Patrik Sternudd

1 TABLE OF CONTENTS

2 INTRODUCTION

2.1 Reading instructions

An important note is that even though this document mostly focus on the technical details of a specific Operating System, the underlying work of a security policy, proper staff training and good physical security must not be forgotten. Those controls are not covered here; the goal is to provide a checklist for an OS. It is an accepted truth that no system can be secure if a malicious person has physical access to it. And the need of a security policy should be obvious as well. What is considered secure at one site may be considered a gaping hole at another, all depending on policy and risk.

All sample output from programs come from real systems that are currently in production. To preserve the security of these systems, I have sanitised all output by replacing all identifying information (e.g. usernames and network addresses).

It is assumed the reader has certain knowledge about Unix systems. I will not explain fundamentals like file permissions. It is also to good advantage if the reader is familiar with the concepts of auditing information systems. The intended public is security professionals, although it is my hope that the information herein may prove useful for other persons as well.

I will use the following conventions for highlighting different types of text (Times New Roman is the default font unless otherwise stated):

Type	Style	Example
Document titles	Italics	*Intrusion Detection*
Program names (and options)	Arial + Bold. Prefixed with a hash sign when used in multi-line examples	**# ftp localhost** **ftp localhost**
Program output	Arial	Name: (localhost:sectest):
Program input	Arial + Bold	**anonymous**
Path names	Arial	/etc/resolv.conf

Quotes will be within quotation marks if shorter than 3 lines, and additionally indented if longer. All quotes will be in italics.

And, oh, I have tried to put at least a little humour in the text every now and then. We all know how fun it is to read texts like ISO/BS Standards (and aren't they dreadfully long, too!). Sure, they are interesting and useful, but entertaining? Hardly.
I do not guarantee that my humour is understandable, but at least I tried. If you find yourself chuckling somewhere, you may have found it.

If anyone is just interested in a FreeBSD checklist (without all extra information in this document), I will try to send one (which of course is the result of this work) to the SANS Institute

as well as the FreeBSD project. With luck, it will be available at any of the locations. Hopefully it will be available somewhere around the end of October 2001.

2.2 Selected target

For the assignments, I have chosen FreeBSD as my target. FreeBSD is interesting because there appears to be little information available about how to audit it. This may be because it is a relatively unknown OS compared to GNU/Linux, Sun Solaris, and various Microsoft OS:es. I feel this work is useful because FreeBSD seems to be commonly used (although the default installation often may be modified) in different devices. One example can be found in the Nokia IPSO solution (a Check Point Firewall-1 appliance). Of course, many people argue that FreeBSD is a dying distribution as it seems to fall in the shadow of GNU/Linux. However, I think the liberal BSD license has some advantages to the more restrictive GPL. Especially for companies wanting to create embedded systems without having to provide source code (I am not saying this is a good thing, but companies tend to protective of their assets and intellectual properties. Not as bad as the music and movie industry though). All data and tests will be done on FreeBSD 4.3-Release since it was the latest stable release available when I begun this work. In other words, your mileage may vary if you apply any methods herein on other versions.

3 A BRIEF INTRODUCTION TO FREEBSD

If you already are familiar with FreeBSD, or simply not interested, you probably can skip this chapter. It is not a requirement for understanding other chapters or sections.

Much (and more) of the facts in this chapter are available at the FreeBSD website which is located at http://www.freebsd.org. If you have not yet done so, I recommend a visit there.

FreeBSD is a Unix system, mostly for the Intel (and compatible) architecture, that is developed by different individuals. It is freely available for download (individuals who wish to support economically are welcome to purchase a set of CD-ROMS).

At a seminar I once attended, the question arose concerning the differences between FreeBSD and GNU/Linux. It got answered by an analogy (unfortunately, I cannot recall who told it): GNU/Linux may be like a brand new sport car. Cool, good looking and very nice to have when cruising in town. FreeBSD is more of an eighteen-wheeler truck. Not very attractive, but still the better choice when you are to transport stuff long distances.

Personally, I like this analogy. GNU/Linux make perfect workstations that are fast, and with all the themes available, very nice looking indeed. But I prefer FreeBSD when a server is to be deployed and I do not have the good fortune to be able to afford SunOS on Sparc architecture (the OS comes cheap, the hardware does not). The goals of FreeBSD are towards performance, stability and security.

FreeBSD is shipped under the BSD License. This is a very liberal license which gives anyone right to take the source, modify it to their own ends and sell it as a complete product without having to supply the modified source itself (as is the case with GPL). The only thing to remember is to keep any disclaimers and give due credit.

4 EXISTING GUIDELINES

I have searched over the Internet and tried to find existing information concerning FreeBSD audits. Unfortunately, I were unable to get any FreeBSD-specific guidelines.
I attempted to retrieve information from several websites.

4.1 The SANS Institute

Organisation/Company	**URL**
The SANS Institute	http://www.sans.org
Methodology	
Site searched the first of August 2001. The keyword "FreeBSD" yielded 79 results and checklist 112. None of these contained any FreeBSD audit checklist. Keyword "audit" gave too many matches to feasibly check so I had to settle with doing random checks on the different pages. The result, though, remained the same.	

4.2 Security Focus

Organisation/Company	**URL**
Security Focus	http://www.securityfocus.com
Methodology	
Site searched the first of August 2001. The keyword "FreeBSD" yielded 35 results, "checklist" 38 and "audit" 265. The search was made on the Library section of Security Focus.	

4.3 Packetstorm

Organisation/Company	**URL**
Packetstorm	http://packetstormsecurity.com
Methodology	
Site searched the first of August 2001. I browsed the "Evaluation", "Unix" and "General" subsections of the "papers" section without finding my goal.	

4.4 Naval Surface Warfare Center

Organisation/Company	**URL**
Naval Surface Warfare Center, Dahlgren Division	http://www.nswc.navy.mil
Methodology	
Site searched the first of July 2001. I browsed the Information Security subdirectory (http://www.nswc.navy.mil/ISSEC/) but failed to find any FreeBSD specific information.	

4.5 Center for Internet Security

Organisation/Company	URL
Center for Internet Security	http://www.cisecurity.org
Methodology	
Site searched the first of August 2001. The only thing I were able to retrieve was the latest Solaris Ruler. While the ruler probably contains useful information, it is copyrighted and have restricted use. As I do not want to mess with someone's copyright (and it is for a different OS), I will not use their Ruler in this document. However, I recommend the reader to grab a version for their own use (at least if they use Sun Solaris).	

4.6 Google

Organisation/Company	URL
Google Search Engine	http://www.google.com
Methodology	
Search executed the first of August, 2001. I did a multiple keyword search: "FreeBSD" "Audit" "Checklist". This gave 159 real matches (353 totally, but some were removed by Google since they seemed identical with others already displayed).	

4.7 The FreeBSD Security Officer

I also contacted the FreeBSD Security Officer[1] via electronic mail and inquired if he had knowledge of any guidelines. This message was sent 2001-07-04, but at the time of submission, no answer has been received. No blame, though. They are busy enough, especially considering they are volunteering and not doing it for money.

4.8 Existing Checklists - Conclusion

My conclusion is that there is no documented and/or easily available state of practise for auditing the selected Operating System. But during my research, I have found information about other Unix systems. As FreeBSD share a common ground with these systems, I will be able to use those guidelines although they may need to be improved to reflect the FreeBSD variant. Those documents are:

- *Risk Assessment/Countermeasure Analysis/Security Test and Evaluation (ST&E) for Unix Computer Systems*, available from the Naval Surface Warfare Center, Dahlgren Division.
- *Unix Computer Security Checklist*, available from UnixTools.com
- *VT IS Security Audit (Solaris 2.5.1 Test Procedure)*, available from Virginia Tech

[1] Information about the Security Officer can be found at http://www.freebsd.org/security/index.html

5 A SPECIFIC FREEBSD CHECKLIST

5.1 Why is it needed?

The existing lists are not adequate because they do not apply directly to FreeBSD but rather other Unix systems like HP-UX and SunOS. As there is no FreeBSD checklist available[2], there should be a certain need for one. That, at least, is my assumption. If I felt otherwise, this document would have ended here.

When it comes to differences, FreeBSD has its kernel running in various security levels, which may be used to ensure the kernel stays the way it was once loaded. This feature does not exist in SunOS, or to my knowledge, in HP-UX, and is therefore uncovered in the their checklists. In other parts, existing items may look the same, but in reality, the commands behaves differently, and affects security in other ways.

I will quote a couple of items from each of the three lists and discuss why they are not valid when working with FreeBSD. There are, of course, other items that can be applied to FreeBSD without further work, but I will not list those. As a rule, items that refer to standard Unix security and password policy tend to be the same, whereas those referring to more specific controls or services are incompliant. I should also mention that some practices have changed since the creation of the lists. I would also like to mention that those lists are very good in themselves, and contains a wealth of information, but unfortunately, they do not cover FreeBSD.

5.1.1 From the NSWC List

Section "Unauthorized Access"

"COPS (or its equivalent) HAS BEEN RUN ON THIS SYSTEM AND VULNERABILITIES WERE CORRECTED ON DATE:"

Well, this tool is getting a bit dated, but it is still useful. However, a system can be made very secure by other procedures. I am interested in creating a checklist that I can apply to any FreeBSD system, no matter in what organisation I am working. A side note; FreeBSD has its own script for doing security tests that in my opinion appears to do much the same as COPS (and the results are automatically mailed to root).

Section "Data Exports"

"IS does not allow information queries (e.g. World Wide Web (HTTP) or similar (archie, gopher, etc.)) from addresses external to the site."

This is extremely organisation specific. I am expecting that some hosts which are going to be audited may in fact be web servers.

[2] See chapter 4

5.1.2 From the Unix Security List

Section "Network Security"
"Create access control lists /var/adm/inetd.sec to say what hosts can connect"

This is OS specific. FreeBSD does not even have a /var/adm directory. Instead, a check should be done whether any host based firewall is being used.

Section "Password Security"

"HP can use trusted system package (via SAM) - if NOT using NIS or NIS+"

This one actually beats me. At this point in my career, I am not very familiar with HP-UX so the meaning is lost to me.

5.1.3 From the VT List

Section "Audit"

"Verify the audit_warn script is configured correctly"

This one is also OS specific. FreeBSD permits process accounting, but there is no auditing subsystem similar to the one offered by SunOS.

Section "Availability"

"Verify user lock out feature is enabled"

To my knowledge, this feature does not exist in FreeBSD. Also, user lockout is a risk in itself, permitting denial of service attacks. Of course, if it is desired, one can always create a script that runs as root and let it monitor the system logs. But I really would not recommend that.

5.2 The actual checklist

Now that I have some information to begin with, I will attempt to put together a security audit checklist specific to FreeBSD.

To better organise the work, the list is divided into several subsections to make it easier to read (and for that matter, execute, when the time comes to use it for audit). Furthermore, all checklist items will have one of three colours to signify their importance. The colours in the table below will be used:

Colour	Severity	Description
Red	Critical	Items that are required for the system security. None of them should fail in the audit.
Blue	Important	Items that are important, but not required. The number of failed items

		should under no circumstances not be more than 10.
Green	Optional	Optional items provide much for the system security when deployed, but they are not required and may also prove too inconvenient.

If the audit result is within these above specification regarding failed item count, the system can probably be said to be secure enough for all but the most classified or otherwise sensitive environments.

For each of the checklist items, I will provide the following information:

- The severity (see the table above).
- A brief information/description about the item.
- Whether the item is subjective, objective or both. An example on when I use "both" is if I can use a utility to get information, but this information is not enough to pass or reject the test. In these cases, the auditor must use other information to grade the item. Items that are both will be labelled "Objective/Subjective" whereas subjective and objective items will be "Subjective" and "Objective" respectively.
- Accepted results of the item. If the audit data is not within the parameters of what is accepted, the test is automatically failed. When other considerations may either pass or fail an item, I will so state. The fail/success status is not in regard to the item title itself. A failed item is always negative in the sense of security.

Throughout the text, I may use words like "unneeded services" or "securely". Unless otherwise stated, my meaning is that only services or programs required for the correct operation of the system should be installed and running.

5.2.1 Undetected compromises

This section may be skipped when a system is freshly installed from a known good media and has not yet been connected to the network or left without supervision in an insecure area.

The first thing that comes into my mind when I am going to audit a system that has been connected to any network (or if the hardware is placed in an insecure environment) is whether or not the system already is compromised. If the host has suffered from a compromise and a root kit has been installed, I can no longer trust any output from the system. So, to be able to trust the output from this first test, all binaries must be known to be good. The same applies to any used library. Now, this may be achieved by having a set of binaries and libraries on a CD (or even statically linked so the libraries can be omitted). Unfortunately, if the kernel is compromised, it may not make any difference if the binaries themselves are unmodified. So the best way to tackle this problem is to load up another kernel, preferably on CD or floppy. But before doing this, I would consider the issues regarding production systems. I may not be allowed to reboot the system. In this case, I will just have to trust the kernel (there may be mitigating factors, e.g. the kernel security level which will be discussed in a later section).

Item #1	**Are there any modified system binaries or otherwise suspicious files around?**
Severity	Critical
Information	See section introduction
Subjective	If you know of an utility that is capable of - by 100% accuracy - telling me if my

	system is compromised or not, please, let the community know. I certainly would need a copy! Short of that, the auditor must to the extent possible ascertain that the system is in good shape. There are several tools that may be used, but all of these are standard Unix tools which the auditor should be familiar with. A short but good paper on the subject is written by Chris Kuethe[3]; an application that may or may not be of interest is chkrootkit[4]
Accepted Result(s)	There should be no modified system binaries. If this is not the case, a very compelling reason why this is so should exist. For example, some organisations may replace or modify system binaries.

5.2.2 Kernel

The administrator is also responsible for compiling the kernel correctly. A precompiled version is loaded at system installation time, but chances are good that it must be modified to support some feature or another not included by default. There are some issues one should check for in a kernel configuration. For starters, unless the system is used as an IDS (Intrusion Detection System) or is otherwise in need of sniffer capabilities, the bpf (Berkeley Packet Filter) device is unnecessary (this is turned off by default and should probably not be turned on, but we will check for it nevertheless). The *FreeBSD Security How-To* states this as well.

When it comes to the kernel securelevel, the following can be read in the *System Administration section of the FAQ*:

> ***Warning:*** *Securelevel is not a silver bullet; it has many known deficiencies. More often than not, it provides a false sense of security. One of its biggest problems is that in order for it to be at all effective, all files used in the boot process up until the securelevel is set must be protected. If an attacker can get the system to execute their code prior to the securelevel being set (which happens quite late in the boot process since some things the system must do at start-up cannot be done at an elevated securelevel), its protections are invalidated. While this task of protecting all files used in the boot process is not technically impossible, if it is achieved, system maintenance will become a nightmare since one would have to take the system down, at least to single-user mode, to modify a configuration file.*

I, however, still feel the securelevel is an valuable addition to the overall security. First, it is very likely that the average script kiddie not will be aware of how it works (or even that it exists). And if they do, they must still reboot the system to defeat it. Chances are that the prudent system administrator/security officer will notice any unscheduled reboots. But the point is well taken. The administrator should do well to be aware of the limitations of the securelevel (and to the limitation of all other security controls deployed).

Item #2	**Is the kernel using an appropriate security level?**
Severity	Important

[3] Available at http://www.sans.org/newlook/resources/IDFAQ/unix_signs_compromised.htm
[4] Available at http://www.chkrootkit.org/

Information	The FreeBSD kernel may run more or less secure. There are four levels available, ranging from -1 to 3 where a higher number is more secure. For example, level 2 stops any kernel modules from being loaded or unloaded. Kernel modules could otherwise be utilised to defeat various security systems and/or to install backdoors. Only init may lower the security level. The usual trade-off between security and convenience does, of course, apply. A full listing of the respective levels exist in the init (8) manual page.
Objective/ Subjective	The following command can be used to get the security level: **# sysctl kern.securelevel** kern.securelevel: 1 The value is in the range -1 to 3. In this case, it was set to 1.
Accepted Result(s)	This depends on what the system is designed to do. However, -1 should never be used, and 0 should be used with care. The *FreeBSD Security How-To* states: *"For example, if all your system does is web serving, you can safely set your securelevel to 2. However, if you are running an X server, setting your securelevel to 1 or higher will give you problems because X server needs to open /dev/mem and /dev/kmem for writing, and securelevel of 1 prevents doing so."* So if the system being audited only provide one or a few services like DNS or SMTP, the securelevel should at least be 2; perhaps even 3, although as far as I understand the init(8) manual page, 3 is only effective when ipfw also is deployed. If the system instead is a workstation, then 0 probably is a better choice.

Item #3	**Is bpf turned off?**
Severity	Important
Information	See chapter introduction
Objective	A check can be done by compiling and running a small C program which is available in the appendices, or if available, by attempting to start tcpdump. Example output from the attached source code: **# ./bpftest** Trying '/dev/bpf0'...error code 6 This most likely means bpf is disabled. An attempt with tcpdump yields the following error: **# tcpdump** tcpdump: /dev/bpf0: Device not configured The two tests are in harmony. We can safely assume that bpf is disabled.
Accepted Result(s)	*"If someone does manage to get root on your system, having BPF in the kernel will make sniffing of your network much easier for them. Don't compile BPF into the*

	kernel if you won't have a need for it." -- The FreeBSD Security How-To I could not agree more; bpf should be turned off unless the system requires sniffer capabilities. For a good reason, I might add.

Item #4	**Is IP Firewalling support compiled in?**
Severity	Important
Information	IP Firewalling should be used to ensure that only approved network traffic enters or leaves the system in question. FreeBSD comes with support for both ipf[5] and IPFW[6]
Objective	The test for ipf is easily done by issuing the **ipfstat** command. If the following output appears, it is an indication that ipf is disabled: open: Device not configured. If some statistics over dropped and passed packets appear instead, ipf is enabled. Support for IPFW is tested by **ipfw -a l** (lowercase 'l' as in 'letter'). The below output reports that IPFW was, in fact, not enabled: ipfw: getsockopt(IP_FW_GET): Protocol not available Anything else probably mean it is enabled, but with kernels, one just never know.
Accepted Result(s)	ipf or ipfw should be supported.

Item #5	**Is IP Firewall logging support enabled?**
Severity	Optional
Information	This test is dependent on the success of the previous test. Firewall logging is of course of no use if there is no firewall that can take advantage of it.
Subjective	The auditor will have to find out what rules that are supposed to log, and then generate such traffic. The command to list the rules depends on the firewall being used. When a suitable rule is found, **nmap** or **fragrouter** can most likely be used to generate the required traffic. The logging output also depend on the firewall type.
Accepted Result(s)	Logging should occur for the firewall in use.

Item #6	**Is /boot.config created and flagged to disallow kernel change during boot up?**
Severity	Optional
Information	This check is useless if the kernel security level is less than 1.
Objective	**ls** is, as always, a good command to list files with: **# ls -lo /boot.config** -r-------- 1 root wheel schg 0 Aug 11 17:16 /boot.config

[5] Manual page: ipf(8)
[6] Manual page: ipfw(8)

	The fifth field of the output is the flag. As can be seen in the example, the schg flag is set to disallow any modifications.
Accepted Result(s)	The flag "schg" must be set. The test automatically fails if the kernel security level is below 1, because then this flag may be turned off without any problems.

5.2.3 File systems

Item #7	**Are there any world writable files or directories?**
Severity	Critical
Information	World writable files or directories should not exist (except if the sticky bit is set, and there should only be a few of these around).
Objective/ Subjective	There is one utility above all when it comes to finding files on the system: **find**. In this case **find / -perm -0002 -and \(-type f -or -type d \)** may be used to find all files and directories that allows world write access to them.
Accepted Result(s)	There must be a valid reason for all world writable files or directories to have that particular permission. It is not a problem at all if no such files exist.

Item #8	**Is the "sticky bit" set on all tmp directories?**
Severity	Critical
Information	The sticky bit is used to protect a file so that only its owner may write to it (or unlink it, for that matter). As everyone usually have write access to /tmp, this mode is a very good thing to have.
Objective	**find / -type d -name tmp** can be used to find all directories named tmp. **ls -l** may be used to get the current permission.
Accepted Result(s)	The last bit should be 't' (the sticky one) on all tmp directories that allow world or group write access to them.

Item #9	**Does anyone except root have write permission in roots path?**
Severity	Critical
Information	This check makes sure that no one may alter any command normally used by root. If this was possible, it would be easy to install a trojan that eventually would be executed by root.
Objective	For all directories (and files within them) in root's path, determine the effective permissions on them. For example, assume that /usr/sbin is in the path. Example output may look something like this: **# ls -l /usr** drwxr-xr-x 2 root wheel 4096 May 21 08:13 sbin/ At least the sbin directory is safe. Root is the only one with write permission.
Accepted Result(s)	Only root should able to write to any of the directories or files in the path.

Item #10	**Is any exported file system allowing root privileges?**
Severity	Critical
Information	If NFS must be deployed, it should be done as securely as possible. One very bad thing to do is allowing root to mount a file system remotely and retain privileges. Even if NFS not is running, any configuration files associated with it should be in good shape in the event that it is enabled
Subjective	I have not been able to determine a objective way to get information from the server itself so the auditor must resort to reading the /etc/exports file. The option **-r** or **maproot** indicates that root access is mapped to the ID written after. The ID should NOT be root.
Accepted Result(s)	Root access should be disabled (i.e. mapped to something else, -1 may be a good idea for example)

Item #11	**Is NFS running?**
Severity	Important
Information	NFS (Network File System) is used for exporting file systems to other machines. However, it comes with a bunch of security risks as well.
Objective	The following example indicates that NFS not is running: **# showmount -e** RPC: Port mapper failure showmount: can't do exports rpc
Accepted Result(s)	NFS should not be running.

Item #12	**Is any exported file system allowing read-write?**
Severity	Important
Information	If possible, all NFS file systems should be exported read only. This is of course not feasible when users' homedirs or mail folders are about to be exported. But if there is just some programs like emacs, then why not add some extra security on it? And, if NFS not is running, why have a potential hole waiting?
Subjective	I have found no way to get this information from the server. The information in /etc/exports must be used here as well, but now the **-ro** option should hold our interest.
Accepted Result(s)	Read-write access should be disabled unless the opposite is required (as in users' homedirs).

Item #13	**Are file systems that only contain user or application data mounted with nosuid and nodev to protect against rogue programs?**
Severity	Important
Information	Another layer of protection can be achieved by having a file system to ignore setuid files and devices. I have yet to encounter a good reason why an user would have a setuid file in his homedir. And devices belongs to /dev, nowhere else.
Objective	The command to use is **mount**. Example output may look like this: /dev/ad0s1a on / (ufs, local).

	The option field is within the parentheses.
Accepted Result(s)	If there is any file system dedicated to users or applications, nosuid and nodev should be listed in the options field.

Item #14	**Are important system binaries flagged to disallow modifications?**
Severity	Optional
Information	For the truly paranoid (provided that the kernel security level is greater than 0, which, I assume, it will be for the truly paranoid), there is the option of going through important system binaries and add the schg flag to them, thus making it much harder to replace them with trojan versions.
Objective/ Subjective	To decide whether the schg flag is set, simply use **ls -lo** and look for the string schg. However, it will be up to the auditor to determine which files that are important.
Accepted Result(s)	The auditor feels the most important system binaries (and, if applicable, static configuration files) has the schg flag set.

5.2.4 Programs & Applications

Programs with the setuid/setgid flag should be avoided. As they run as the owner/group of the file (usually root or bin) instead of the person using it, any potential security hole in them are very dangerous. Only programs that absolutely requires this flag should have it. Also, non-setuid/setgid programs can pose the same danger if they are run as daemons and root owns the process. They may even prove more dangerous because they seem more harmless, but are not. Care must be taken about which programs that are installed, and whom they are run by. Root should be running as few processes as possible.

Item #15	**Are there any files that have excessive suid/sgid flag?**
Severity	Critical
Information	See chapter introduction
Objective/ Subjective	All setuid/setgid files may be listed by using **find / -perm -4000 -or -perm -2000**. The auditor will have to go through the list and ensure that the files listed absolutely required the given permission.
Accepted Result(s)	Only files that absolutely need to have a setuid or setgid flag be equipped with it. Note that the file itself may be unneeded. Then it should also not have the flag.

Item #16	**Are any programs being unnecessarily run as root?**
Severity	Critical
Information	See chapter introduction
Objective/ Subjective	All processes that runs can be listed with **ps -aux** but the auditor must do an evaluation on all results.
Accepted Result(s)	All processes run by root must require that special privilege. It should be unfeasible to deploy sandboxes or run them as other, less powerful accounts.

Item #17	**Are sandboxes/chroot jails in use?**
Severity	Important

Information	An additional layer of defence can be gained by using sandboxes. They also are commonly referred as chroot jails. This type of defence is most common to be applied on network services like DNS and SMTP (not on the services themselves of course, but rather the applications providing them).
Subjective	This test is dependent on which applications that are installed , and whether or not they can handle sandboxes.
Accepted Result(s)	All applications that permits it must be run in chroot jails.

Item #18	**Are any file integrity testing tools being used?**
Severity	Important
Information	You most likely cannot say your system is secure if you do not keep track of file modification. Keeping this control is one of the best ways for detecting if an intrusion or other misuse has occurred. Several tools exist, most famous among them is probably Tripwire[7].
Objective	Most such tools will be run from a crontab (most likely, root's). **crontab -l** is as usual the right command for checking crontabs.
Accepted Result(s)	A file integrity tool should be run regularly. The check should be done at least once a day, but for important files, even more often may be appropriate.

Item #19	**Are there any unneeded applications or programs installed?**
Severity	Important
Information	My reasoning is simple. More applications means more lines of code. The more lines of code, the more likely a bug. Programs that are not needed should not be installed. Only stuff that is needed for the system to function properly should be installed. Nothing less, nothing more (sounds easy, does it not?).
Subjective	For each program or application installed, the auditor must consider if the program/application is required.
Accepted Result(s)	The auditor should feel that all installed programs and/or applications are needed.

Item #20	**Are required administration and debugging tools kept so that only administrators can use them?**
Severity	Optional
Information	Although I strongly discourage debugging tools on production systems, I realise situations may arise when they may be needed. And administration tools are always good to have around. But with the risk of sounding a bit draconian, maybe all users should not have access to them?
Objective/ Subjective	Whereas ping and traceroute are good examples on programs ordinary users seldom need access to, different applications comes with their own management/administration/debugging utilities. As always when we are going to determine permissions, **ls** and **find** are good utilities to use: **# ls -l /sbin/ping**

[7] By Tripwire Security, Inc - http://www.tripwiresecurity.com/

	-r-xr-x--- 1 root wheel 196376 Apr 21 11:08 /sbin/ping **# ls -l /usr/bin/rsh** ---------- 1 root wheel 7584 Apr 21 11:09 /usr/bin/rsh The first of the above examples indicate that only root and members of the wheel group may use the network testing tool ping. The second shows that the administrator clearly thought rsh should not be used at all.
Accepted Result(s)	The auditor must decide the outcome of this test. I cannot list all tools from every application ported or otherwise available to FreeBSD. The list would simply be too long and I would hardly be able to finish it within the next few years, if ever. It also depends on what the system is used for. A development box where people compile stuff all days long is something altogether different from a mail server.

5.2.5 Patches

From time to time, a vulnerability will be found in a program. It is very important that such vulnerabilities are corrected as soon as possible. This is true for all programs, but those which are setuid or run by root must be prioritised. Patching is nothing you do once and then are done with it; it is an on-going never-ending process (which most administrators are painfully aware of).

Item #21	**Are all security related patches applied?**
Severity	Critical
Information	See chapter introduction.
Objective	The auditor should read all FreeBSD Security Advisories[8] and check whether all patches and/or other countermeasures described within the advisory are applied to the system. There is of course no need to check for a patch if the current system not is vulnerable, or the application in question not is installed (e.g. the advisory may only apply for previous releases of the operating system).
Accepted Result(s)	No unpatched application should be found.

Item #22	**Are third party patches applied?**
Severity	Critical
Information	See chapter introduction
Subjective	I am not a seer, so I cannot foretell what third party applications that will be installed, nor can I predict where security information can be downloaded. For all third party applications that are installed, the auditor should check their respective websites to discern whether or not the installed version has any security patches that should be applied.
Accepted Result(s)	All applications should be up to the current patch level (which does not necessary have to mean the latest release of the software).

[8] The complete listing is available at http://www.freebsd.org/security/index.html#adv

5.2.6 Accounts & Passwords

Item #23	Has all accounts non-empty password fields?
Severity	Critical
Information	An empty password field means that the account associated with it has no password. This is, for obvious reasons, very bad for security.
Objective	All lines in the /etc/master.passwd file should be reviewed. An account with an empty password field will look something like this (the second colon-delimited field should contain a password but does in fact not): test::1000:1000::0:0:Test User:/home/test:/usr/local/bin/bash
Accepted Result(s)	No account may have an empty password field.

Item #24	Is it possible to login as system accounts?
Severity	Critical
Information	System accounts are just that. System, that is. They should not be used by users.
Objective	/etc/master.passwd contain all password hashes.
Accepted Result(s)	All system accounts (root not included) must have a password field consisting of a single star or something similar.

Item #25	Is the minimum password length too short?
Severity	Critical
Information	Unless one-time password technologies are used, brute force or dictionary attacks becomes possible. There is only one good defence for these attacks. Strong passwords of sufficient length.
Objective	All system wide password configuration exists in the /etc/login.conf [9] file. By default, the length of 6 is enforced although it may be overridden. If an entry named "minpasswordlen" exists in the file, the number after it should be greater than, or equal to 6.
Accepted Result(s)	For a normal system, the minimum limit should be 6 or more. For a system where security is more important (or convenience is less important), it should be 8 or more. This in turn requires a different algorithm than DES to be used, but on FreeBSD, the **passwd** command by default uses MD5. The auditor must decide which of the options that apply. If the length is less than 6, the test has failed.

Item #26	Are strong passwords enforced?
Severity	Critical
Information	Even if a password is long enough to be secure, it is even more important that it is chosen with security in mind. An easily guessable password is a bad one. The system should enforce the password quality.
Objective	The file /etc/login.conf should contains the following entries: passwd_format md5 mixpasswordcase true

[9] Manual page: login.conf(5)

	The first means that MD5 passwords are used instead of DES which allows password lengths greater than 7 characters. The second tells passwd to warn if a new password contains only lower case letters.
Accepted Result(s)	Strong passwords should be enforced. That means the possible password length is above seven so the algorithm must be md5 or better. It also means that new user passwords should only be accepted if they contain mixed case letters. Additionally, One-Time password solutions are considered to be strong (they are usually very hard to guess).

Item #27	**Does passwords expire after a certain time?**
Severity	Critical
Information	Passwords should be changed regularly and that is the end of it. No arguments accepted.
Objective	The seventh field in the /etc/master.passwd file should be checked to determine if the current passwords will expire or not. Also make sure the /etc/login.conf file contains the passwordtime statement.
Accepted Result(s)	The test is successful if any of the following conditions are met: 1) All user account passwords has an expiration date and the system is configured to set a new expiration date upon password renewal. 2) An One-Time-Password (OTP) solution is deployed. 3) Password changes is enforced by other means.

Item #28	**Are all user and group IDs unique?**
Severity	Critical
Information	As the system refers to accounts by numbers and not names, it is important that all accounts and groups have unique ID numbers.
Objective	Check /etc/master.passwd and /etc/group for duplicate numbers.
Accepted Result(s)	No duplicated ID's should exist.

Item #29	**Is the password database and master file securely kept?**
Severity	Critical
Information	An additional layer of security is gained by using shadow passwords. This layer is worthless if the shadow file is world readable.
Objective	The file permission can be viewed by **ls -l**.
Accepted Result(s)	The following files should be readable only by root: /etc/master.passwd, /etc/spwd.db. The files /etc/passwd and /etc/pwd.db should be readable by anyone. None of these files must under no circumstances be world or group writable.

Item #30	**Are all accounts on the system still active?**
Severity	Important
Information	Inactive user accounts are bad in two senses. First, persons who has left or no longer is in need of access should not be able to log in to the system. Second, the chance that misuse of such an account is detected is less than a normal one.
Subjective	The /etc/master.passwd file contains all user account names, but it does of

	course not tell anything about whether the person associated with the account still should have access or not.
Accepted Result(s)	All accounts should be active.

Item #31	**Are there any shared accounts?**
Severity	Important
Information	Shared accounts should never be used unless no other option can be used.
Subjective	There is no data that can tell if an account is shared (well, accounts named "guest" or similar could be a good indication, but the name in itself does not prove anything).
Accepted Result(s)	The auditor should be sure there are no accounts made specifically to be shared (what normal users does with their own accounts is another story, often sad).

Item #32	**Are secure umasks enforced?**
Severity	Important
Information	To prevent sloppy permissions on files or directories, the umask should be set to a sensible value in system login profiles.
Objective	For all system configuration files that are sourced at login time, at least one should contain an umask directive.
Accepted Result(s)	The daemon umask should be 022 or stricter; the user umask 022 or 027. Both values are according to the course material[10]. Well, they are sensible values so why not use them? If it is something else, it should be for a very good reason. If nothing of the above is true, the test is failed.

Item #33	**Is the usage of cron restricted?**
Severity	Important
Information	The cron daemon allows for periodical execution of programs. While it is a powerful tool, normal users should only have access to it when it is explicitly required.
Objective	/var/cron/allow or /var/cron/deny should contain users that are allowed or denied cron usage. Only one file should exist[11].
Accepted Result(s)	cron usage should be restricted. The restriction is preferably done by using /var/cron/allow (everyone not in the file are denied).

Item #34	**Are restricted shells used?**
Severity	Optional
Information	Some (or most, depending on the system), users only need access to one or a few applications. In these cases, a restricted shell that permits that application (and that only) can be a good idea.
Objective/ Subjective	The /etc/master.passwd file contains information about which user uses what shell. It does not contain information about what tasks an user usually does.
Accepted	All users that use a limited set of commands are assigned a restricted shell.

[10] Marchany, p.154.
[11] Manual page: crontab(1)

Result(s)	

5.2.7 Root Access

Item #35	**Is root the only account with id zero?**
Severity	Critical
Information	Any accounts with the id zero are root. If there is more than one, the security implications increases. It is bad enough with one almighty account. It should perhaps be mentioned that FreeBSD by default provides the zero id account toor which is to be used if the admin desires a root account with a different shell. However, I think the root access should be as little as possible, and it is always possible to start a new shell after login.
Objective	Read through /etc/passwd and check for accounts with id zero.
Accepted Result(s)	There can be only one...account with the id of zero.

Item #36	**Are remote root logins allowed?**
Severity	Critical
Information	Root should never be able to log in remotely.
Objective	For each listening service that provides login capabilities, try to log as root. Examples of such services are telnet, ftp and ssh.
Accepted Result(s)	Root must not be able to log in remotely.

Item #37	**Does root's PATH variable contain a single dot?**
Severity	Critical
Information	A "." in the path means that commands will be executed from the current directory. This is especially dangerous if the dot is first in the path. It opens up for trojan attacks. Assume, for example, that a trojanised ls exists in an user's homedir. It is very likely the first command issued by root when entering the directory will be just 'ls'.
Objective	Example: # **echo $PATH** /sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:/root/bin No single dot in that path.
Accepted Result(s)	The sub string **:.:** (colon, dot, colon) must not be found in the output from the echo command.

Item #38	**Are only trusted accounts members of the group 'wheel'?**
Severity	Critical
Information	Accounts that are members of the group wheel are those who can, provided they know the password, su to root.

Objective/ Subjective	Read through /etc/group and make sure that only trusted accounts are members of the wheel group.
Accepted Result(s)	Of course, the fewer members of 'wheel', the better. Those who indeed are members should be so for a very good reason (e.g. they are administrators).

5.2.8 Network Configuration

Item #39	**Are all unneeded services disabled?**
Severity	Critical
Information	Since all network services potentially contains vulnerabilities that may be exploited to gain access, only services that are used should be enabled. If it is possible, not even inetd should be run.
Objective/ Subjective	**netstat -a** can be used to determine which ports that are listening. All ports will have a process associated with it (the process opened the port). The more powerful tool lsof will display the processes as well, but it is not installed by default. If **inetd** is used, it will spawn a new process for each service request that drops in, so in this case, the auditor check /etc/inetd.conf to see which services that are enabled.
Accepted Result(s)	The amount of services that are expected to run are indeed run. If fewer than expected are run, this should be noted in the report as well, but it will not give this item a failing grade.

Item #40	**Are all system accounts (including root) added to the ftpusers file?**
Severity	Critical
Information	If FTP is enabled, the /etc/ftpusers file should exist to protect users who normally not is logged in (e.g. root and bin). If FTP is disabled, the file should exist in case it is turned on by someone in the future. Most (if not all) Unix FTP servers honour this file.
Objective	The test is done by reading through the ftpusers file and comparing user named with the password files. The syntax is one account name per line.
Accepted Result(s)	All system accounts should be listed in the file. By system accounts, I mean all accounts that are not belonging to normal users. If these conditions are true, the test is successful.

Item #41	**Is Anonymous FTP (anon-ftp) disabled?**
Severity	Critical
Information	Anonymous FTP should only be permitted if absolutely necessary.
Objective	Initiate a connection to localhost and give the user id of anonymous, e.g.: **# ftp localhost** Connected to localhost 220 labsys FTP server (version 6.00LS) ready. Name: (localhost:sectest): **anonymous** 530 user anonymous unknown

	This indicates that anonymous access is denied. Other FTP servers may respond differently, for example "anonymous access denied".
Accepted Result(s)	Anonymous FTP access should only be enabled if this functionality is required.

Item #42	**Is it possible to authenticate through .shost .rhost and hosts.equiv files?**
Severity	Critical
Information	Sometimes it may be tempting to skip the password requirement on the assumption that if a user already is authentication on one host, then he is allowed on another too. More often that not, this can endanger the security.
Objective	The auditor should check all configuration files of currently available authentication mechanism (e.g. rlogin, ssh, etc).
Accepted Result(s)	All passwordless authentication should be disabled.

Item #43	**Are source routed packets ignored?**
Severity	Critical
Information	At this point of the Internet evolution, source routed packets almost certainly mean that someone is up to something nasty. A system that accepts such traffic is subject to several attacks.
Objective	There are two commands that should be run (both will return 0 or 1): **# sysctl net.inet.ip.accept_sourceroute** sysctl net.inet.ip.accept_sourceroute: 0 **# sysctl net.inet.ip.sourceroute** sysctl net.inet.ip.sourceroute: 0 0 in both cases means that the system neither will accept an incoming source routed packet; nor will it agree to transmit one.
Accepted Result(s)	The system should ignore all source routed packets.

Item #44	**Are there unneeded points of access like extra network interface cards, or modems?**
Severity	Critical
Information	Sometimes a system is provided with several network interface cards (NIC), but if only one is used. Unused cards should not be configured, and more importantly, not connected to any network. The same thing applies to modems. They should only be attached to the system if they are in use.
Objective/ Subjective	**ifconfig** may for example be used to list which interfaces currently exists in the system. A physical inspection is also in place.
Accepted Result(s)	There should be no unneeded paths into the system.

Item #45	**Are any network interface cards in promiscuous mode?**

Severity	Critical
Information	A network interface card that is in promiscuous mode is listening to all traffic on the wire. In other words, a sniffer is most likely deployed in one form or another.
Objective	**ifconfig -a** lists the status of all network interface cards. If an interface is in promiscuous mode, the output may look something like this: xl0: flags=8843<UP,BROADCAST,RUNNING,PROMISC, SIMPLEX,MULTICAST> mtu 1500
Accepted Result(s)	Unless the system is a sniffer (and a legitimate one), no NIC should be in promiscuous mode.

Item #46	**Is ssh used instead of telnet?**
Severity	Critical
Information	Telnet should be replaced by encrypted applications whenever possible. If telnet is used, any typed password could be compromised (from a security standpoint, they probably should be considered compromised too; the only mitigating factor I can think of is if a one time password solution is deployed).
Objective	I prefer to use **netstat -a** to decide which ports that are listening: Active Internet connections (including servers) Proto Recv-Q Send-Q Local Address Foreign Address (state) tcp4 0 0 *.https *.* LISTEN tcp4 0 0 *.ssh *.* LISTEN The output tells us that port 22 (ssh) is open, but not telnet. If telnet were used, the output would say *.telnet instead of *.ssh.
Accepted Result(s)	telnet should not be running. SSH is not a requirement for this test to succeed.

Item #47	**Are any "r-services" used?**
Severity	Critical
Information	R-services (rsh, rlogin, etc) are bad for security since they more often than not employ passwordless logins and utilise trust relationships. They should be replaced by SSH whenever possible.
Objective	Check /etc/inetd.conf whether any r-services are used.
Accepted Result(s)	No r-services should be enabled.

Item #48	**Does the current IP configuration match with the configuration files?**
Severity	Important
Information	If the machine is rebooted, it is a good thing it has the same configuration when it comes up again.
Objective	The auditor should compare the current configuration (**ifconfig -a, netstat -rn**) with the one in the /etc/rc.conf file.
Accepted Result(s)	The IP configuration should match with what is specified in the file /etc/rc.conf

Item #49	**Are all user accounts not in need of FTP access added to the ftpusers file?**
Severity	Important

Information	In addition to blocking system accounts, all user accounts that does not explicitly require FTP access should be added to the ftpusers file.
Objective/ Subjective	The /etc/ftpusers file contains the blocked accounts, but the auditor must go through this file and compare it with the password files as well as have information about which users require FTP access.
Accepted Result(s)	All accounts not listed in the ftpusers file are those that are required to have FTP access.

Item #50	**Are there any idle root shells?**
Severity	Important
Information	Root should only be logged in when there is administrative work to be done. Idle root shells can mean everything from a compromise to sloppy procedures.
Type	Objective. The example shows how all logged in users may be listed: **# w** 8:25 PM up 4:54, 2 users, load averages: 0.00, 0.00, 0.00 USER TTY FROM LOGIN@ IDLE WHAT root v0 - 3:31PM 42 bash root v1 - 8:23PM - w We can see that one root not is idle at all (the dash character indicates this, or in fact, it indicates an idle time less than a minute) and that the other has been idle for 42 minutes.
Accepted Result(s)	The test has failed if root is logged in and idle for more than 2 hours.

Item #51	**Are directed icmp broadcasts ignored?**
Severity	Important
Information	Directed icmp broadcasts should be disabled to prevent the system from acting as a smurf amplifier or allowing a malicious person an easy way to map the network.
Type	Objective. **sysctl net.inet.icmp.bmcastecho** returns either 0 or 1
Accepted Result(s)	0 should be returned for success (indicates that the feature is disabled)

Item #52	**Are packets to closed ports being logged?**
Severity	Important
Information	This feature should be enabled to give early warning on network based attacks.
Type	Objective. Use **sysctl net.inet.tcp.log_in_vain** and **sysctl net.inet.udp.log_in_vain**. Possible return values are 0 or 1
Accepted Result(s)	If either command returns a value of 0, the test has failed.

5.2.9 System logs

Logging is vital. If no logging is done, the possibility to quickly get notified of suspicious activities decreases dramatically. This in turn greatly increases the detection time.

Item #53	Are the system logs mirrored to another system?
Severity	Critical
Information	In the event of an intrusion or hardware failure, it is important that the system logs are available at another location. A daily transfer is not enough since among the first thing an intruder will do is to erase logs in an attempt to hide his presence. Additionally, a central loghost eases up administration (combined with sound policies, of course).
Objective	It is assumed that the standard log daemon is in use. The auditor should control the syslog daemon configuration and ascertain that log events also are sent an external host. If possible, he/she should also try to verify that the external host is functioning properly. There are two files of interest: /etc/hosts (should contain a loghost entry) and /etc/syslog.conf (in which logging configuration exists)
Accepted Result(s)	External logging should be enabled and verified to be working correctly.

Item #54	Do all log files have restricted access?
Severity	Important
Information	Only privileged users should be able to read log files. There is no need for normal users to have this information.
Objective	**ls -l** can be used to list permissions.
Accepted Result(s)	Only root or members of a special group (e.g. wheel) should be able to read the systems logs.

Item #55	Is authentication information logged to a separate file?
Severity	Important
Information	Unless authentication messages are logged to a separate file, it is possible that this information may be lost in other, less important, messages.
Objective	If the normal syslog is used, the file /etc/syslog.conf will contain its configuration. The following excerpt from /etc/syslog.conf shows a system where all authentication information is logged to /var/log/authlog: auth.info /var/log/authlog
Accepted Result(s)	Authentication messages should be logged to a separate file.

Item #56	Is NTP used to ensure that the system time is correct?
Severity	Important
Information	Log events lose much of their value if they have an incorrect timestamp. This is especially true if a central loghost is deployed (as it should).
Objective	The **ps -aux** command may be used to list processes. If NTP is used, a process named ntpd or xntpd usually exists. The file /etc/ntp.conf contains the daemon's configuration.
Accepted Result(s)	NTP is deployed and uses at least four different servers/other sources as reference. The reason for the last condition is that at least three sources must exist in the event of one of the servers becoming corrupt. The additional fourth is to guarantee operation if one server becomes unavailable.

Item #57	Is syslogd configured to drop packets from other hosts?
Severity	Important
Information	A normal system should listen on as few ports as ever possible. This includes the syslog daemon.
Objective	The way syslog is running can be viewed by using **ps -aux**
Accepted Result(s)	syslogd should be running with the **-s** switch. The only exception is if the system in question is a central loghost. See also syslogd(8).

Item #58	Are log files rotated?
Severity	Important.
Information	Log file sizes should be kept manageable. The size may vary between systems, but usually somewhere between 100KB and one MB is okay. If the files becomes too large, it will get harder to review them. The **newsyslog**[12] command is an excellent way to handle this.
Objective	The configuration parameters in /etc/newsyslog.conf decides how log file rotation is to be done. A sample line may look like this: /var/log/cron 600 3 100 * Z This means that /var/log/cron will have 600 as permission, three archive files will be kept, it will be rotated when it reaches the size of 1K, and only then (it is possible to force a rotate at a given time or date), and finally, compress the file upon rotation. Also, the **newsyslog** command should be scheduled to run at least once an hour. This can be checked by running **crontab -l** as root. The following line indicates that newsyslog is run on hourly basis. 0 * * * * /usr/sbin/newsyslog
Accepted Result(s)	Log files are rotated periodically or when they reach a specified size.

Item #59	Does all applications that provide network services log connections?
Severity	Important
Information	Even though IDS' (Intrusion Detection Systems) or host based firewalls can log incoming sessions and in some case even list any sent data, they cannot determine how the receiving application will interpret it. Therefore, the built-in logging functionality (if existing) should be used whenever possible.
Objective/ Subjective	Any text viewer (e.g. **less**) can be used to monitor logs. **ps** and **netstat** can show which services are running. However, as different systems use different services, it is hard to say which ones that are running beforehand.
Accepted Result(s)	All applications that provide network services log their connections. As I can present no sample data, the auditor must decide if the condition is met.

5.2.10 Procedures & Policy

Item #60	Is the root password kept securely?

[12] Manual page: newsyslog(8)

Severity	Critical
Information	The fewer that know the root password the better.
Subjective	There is no method available to say exactly how many persons' brains contain a certain set of information (in this case, the string consisting the root password). Unfortunately, the auditor is up to reviewing policy documents (if they exist) and interviewing managers and administrators.
Accepted Result(s)	The auditor should be able to ascertain that only those who absolutely requires the root password are in possession of it. The password should under no circumstances exist in clear text on any system, nor should any sticky notes with the password be attached to the system hardware or similarly available. The best thing is if the password is kept in a safe and only is used in emergency situations.

Item #61	**Is the root password changed regularly?**
Severity	Critical
Information	Even if the root password is securely kept and seldom used, it should be changed periodically (as should all other passwords, of course).
Subjective	Most administrators I have met are reluctant to put password aging on their root accounts. Of course, the command **chpass** can be used to view information about the root account, but as root is able to change that file on whim, this item is more of procedural type in my opinion. A policy regarding the root account should be in place, and the operator's log should contain data when the password last was changed.
Accepted Result(s)	The auditor must decide whether a password changing procedure exist for the root account. It should be changed at least twice a year, but once every third month is better. The test should be considered failed if no changing scheme exist, or the period is greater than one year. If no procedures regarding the root account is in place, it is best to assume no regular password changing take place (i.e. test fails).

Item #62	**Is the screen locked or logged out at console when no one is using it?**
Severity	Critical
Information	If it is possible for an passers-by to simply sit down at the console and start typing away, the security is almost non-existent.
Subjective	The auditor must do an on-site check to verify whether the console is locked or not. He/she should also talk with the administrator in charge to get information about the normal procedure concerning this. It is hard to discern procedural compliance through system utilities.
Accepted Result(s)	The console screen should be locked whenever not in use.

Item #63	**Is a password tester run periodically?**
Severity	Important
Information	There is only one way to make sure that no weak passwords are around. Run a password tester such as crack or john to find passwords that does not comply with the policy.
Subjective	There is no good way to objectively determine if a password cracker is run regularly. It could be run as a cron job, it could be run manually, or (probably the

	best), the /etc/master.passwd file could be copied to another system as the notion of a password cracker installed on a production system may meet certain opposition (as it should). The approach to be taken here is probably to interview the system administrator in charge of the IS in addition to the review of policy documents.
Accepted Result(s)	Password testing is done periodically.

Item #64	**Is su / sudo used for elevating access?**
Severity	Important
Information	Whenever work is done with root privileges, the administrator should log in with his normal account (which he should have) first, and then become root. The reason for this is to be able to trace activities to a person through system accounting or log information.
Subjective	It is not feasible (or rather, in most cases very stupid) to not allow root access on the console, or have a root account with a blocked password (such as a star). The auditor must depend on reading policies or operational procedures.
Accepted Result(s)	Direct root access is avoided save at emergencies.

Item #65	**Are backups taken regularly?**
Severity	Important
Information	There are several occasions when a backup can be useful. Some examples are files deleted by mistake, hardware failures and security compromises.
Objective/ Subjective	The objective part pertains to the likelihood that if automated backups are taken, there will either be a daemon running (usually when backup is done over the network) or an entry in the crontab file (if backup is done locally, e.g. DLT or DAT). For this, **ps** and **crontab** may be utilised. But (here comes the subjective part) periodical backups may be done by hand, (e.g. the administrator typing the commands himself, or manually activating the network based system) and that is also a perfectly good solution (unless, of course, the administrator forgets, becomes ill, and so on). In this case there will be no way of knowing, except of interviewing the administrator in charge.
Accepted Result(s)	Backups should be taken regularly.

Item #66	**Are restore procedures verified?**
Severity	Important
Information	A backup is useless if it cannot be restored to the system or if it is applied in a wrong way.
Subjective	As no sane administrator usually perform backup restoration tests on production systems (he/she might, but then there usually are very good reasons for it), the auditor must depend on procedural documents, policies and reviews to make sure that the restore procedures are verified.
Accepted Result(s)	The restore procedures are verified and working. The test is considered failed if the procedures are verified not to be working ('Yes, the restore completed without

	errors. But the /etc directory looks like something copied from /dev/random').

Item #67	**Are log and accounting information monitored?**
Severity	Important
Information	Monitoring data is no good if no one reviews it. This is best done by humans due to the complexity of connecting several not-so-critical events and decide that they together make one very critical event (e.g. one failed login on one system may not be critical but if it occurs on all systems at the same time, alarm bells should start ringing).
Subjective	It is impossible to tell whether someone reads the logs except asking around or watching the team/administrator during a working day.
Accepted Result(s)	Log files are reviewed at least daily.

Item #68	**Are all changes approved and documented?**
Severity	Important
Information	Nothing should be done to a production system unless the action is planned, documented and approved before it takes place. This way, many embarrassing or potentially damaging mistakes may be avoided.
Subjective	I cannot think of any command that may enquiry whether or not a policy or procedure is followed. We are stuck with interviews. Hopefully, someone will point out the appropriate documents (for example, the logs telling who approved what and why). Well, if the system is found running, say, SETI@Home, and no documentation says it should, then we probably found a system with no such policy, or a system with a policy no one cares about.
Accepted Result(s)	Work is never done on the system without planning, approval and documentation prior to the action. Of course, there may be other procedures that overrides this in an emergency situation. Such procedures does not cause this test to fail.

Item #69	**Is there a contact list for all users on the system?**
Severity	Important
Information	If monitoring data indicate one user is misusing resources, it should be easy to call that user and ask what he/she is doing (or know where he is so that security may apprehend him/her).
Subjective	Even if we find such a document on the IS, how do we know it is up to date? Interview the manager or administrator is probably a good way!
Accepted Result(s)	A contact list for all users on the system is available outside the IS in question (a nice place may be in a safe).

Item #70	**Is a revision control system in use for important configuration files?**
Severity	Important
Information	Revision control systems such as RCS are very handy to have in place. Especially when someone has done changes and needs to revert to the last working version quickly.
Subjective	There are several possible systems that can be used. It is also hard to say which information files that should be controlled. For example, if sendmail is being used

	and the system is an organisational MTA, /etc/sendmail.cf becomes quite important. If the sendmail is used, but only for delivering email to another system, the file might not be so important. The auditor should first enquiry whether revision control systems are in place, and if affirmative, poke around a bit on the system to mark likely important files, and then check if they are controlled.
Accepted Result(s)	A revision control system is deployed and most of the important configuration files seems to be controlled.

Item #71	**Is all installed software trusted?**
Severity	Optional
Information	One can only trust a system if all components on it are verified to be secure.
Subjective	If there is a way to check if a particular software has been security screened/tested, I do not know it. And the word "trusted" has different meanings depending on whom you ask. In this case, I feel compliance with the test should be made according to the site security policy. There should be a section describing software criteria's and possibly required evaluations before they are applied to a production system. Some organisations may require all software to be open source so they may review the source code, others may state that only applications from a list of trusted vendors are accepted.
Accepted Result(s)	New software installation procedures conform with the security policy. If no policy exists or if this subject is omitted, the test is failed.

Item #72	**Are there login banners in place stating that the system is for authorised use only, together with a statement saying that all activities may be logged in the process of finding misuse?**
Severity	Optional
Information	This does not affect the security in the technical sense. But in some countries, these types of banners are required to be able to monitor personal data, or to seek legal action against an intruder or fraudster.
Objective	The easiest way is by logging in to the system and see if any banner is displayed.
Accepted Result(s)	A banner is displayed at login time.

6 A REAL WORLD SECURY AUDIT

6.1 Objectives

Now that I have a complete checklist, it is time to test it. As I think the checklist to contain at least some quality, my goal is also to know whether my target system is secure or not. In other words, it is time to do an audit. The target is running FreeBSD 4.3-Release and is in production use as a reverse proxy server for HTTP and HTTPS. The system itself is placed behind a firewall, filtering out some of the malicious traffic, but I will not count on its ability to block all intrusion attempt. As the server accepts connections from the Internet, I will rate its placement as medium risk (it is, after all, protected by a firewall).

Some bad news though: time did not allow for me to test all items.
Instead, I had to settle for only 28 items of totally 72.
When a task is performed as a normal user, the command line will be prefaced by $ (dollar) sign. If root access is required, the line will begin with a # (hash) sign. Unless otherwise stated, it is assumed that an user is logged on when the dollar sign begins the line; similarly, it is assumed that root privileges somehow has been achieved when the hash sign is used. The least privilege that allows the task to be performed will be used; I will not waste space by showing that a normal user may not access a certain file unless this is the actual test.

Also note that in order to avoid an excess of repeated information (not to mention that the page count increased dramatically), I have cut away all fields except the item description and the severity and added a "Testing & Evaluation" field instead.

I would like to point out that whereas screen shots in general are more nice looking than printed command line input/output, I favour the latter because it takes less space and usually is more convenient to read. In order to avoid a 10+ megabyte document, the screen shots would have to be scaled down until bordering to unreadable. The one occasion I will favour a screen dump is when I wish to show the reader how a certain application looks. The problem here (for all who prefers graphics, that is) is that most or all utilities will be command line ones. Unix is usually not nifty graphics when it comes to basic administration; this will be reflected here.

6.2 Undetected compromises

To my great annoyance; I never did this one early on and now, later, I have no time for it. This looked like one of the funniest items. Oh well.

6.3 Kernel

Item #1	**Is the kernel using an appropriate security level?**
Severity	Important
Testing & Evaluation	
I will use the built-in command **sysctl** to perform this test. **$ sysctl kern.securelevel** kern.securelevel: 2 The highest security level is indeed set. The test is successful.	

Item #2	**Is bpf turned off?**
Severity	Important
Testing & Evaluation	

First, I will see what tcpdump says:

 # tcpdump
tcpdump: /dev/bpf0: Device not configured

Them just to confirm it, I will use the source code in appendix 9.2.1. First I copy and paste the source code from this document into my text editor of choice (it is not pico, trust me). After that, I will compile and run it.

 # gcc -o bpftest bpftest.c
 # ./bpftest
Trying '/dev/bpf0'...error code 6
This most likely means bpf is disabled.

Both tcpdump and bpftest agree. My conclusion can only be that bpf support is turned off in the kernel. The test is in compliance with the accepted result.

Item #3	**Is IP Firewalling support compiled in?**
Severity	Important
Testing & Evaluation	
I will check for both ipf and ipfw. First out is ipf: **# ipfstat** open: Device not configured Next one, ipfw: **# ipfw -a l** ipfw: getsockopt(IP_FW_GET): Protocol not available. Unfortunately, none of the firewall types were supported. Test failed.	

6.4 File systems

Item #4	**Are there any world writable files or directories?**
Severity	Critical
Testing & Evaluation	
Although I may tell **find** to list both files directories at the same time, I prefer to do each type in turn as to better discern what I am currently dealing with (bad multitasking on my part). I begin with the files: **# find / -perm -0002 -type f** **#**	

find returned no match at all, meaning there were no world writable files around. Very well. Then I will proceed with the directories.

 # find / -perm -0002 -type d
/tmp
/usr/tmp
/var/spool/uucppublic
/var/tmp
/var/tmp/vi.recover

Oops. Now that I have some entries on my list of suspects, I need to know the exact permissions on those directories:

 # ls -dl /tmp /usr/tmp /var/spool/uucppublic /var/tmp /var/tmp/vi.recover
drwxrwxrwt 2 root wheel 512 Sep 9 03:03 /tmp/
drwxrwxrwt 2 root wheel 512 May 16 12:58 /usr/tmp/
drwxrwxrwx 2 uucp uucp 512 Apr 21 11:02 /var/spool/uucppublic/
drwxrwxrwt 3 root wheel 512 Sep 9 03:03 /var/tmp/
drwxrwxrwt 2 root wheel 512 Sep 5 18:45 /var/tmp/vi.recover/

Seems that everything is OK except the uucppublic directory. All others have the sticky bit set. What it boils down to is whether or not uucp is used. As this system is a web server at a medium risk location, it most likely should not be used in any case. A quick interview with the administrator tells us that uucp not is in use so I will have to consider the test as failed.

Item #5	**Is the "sticky bit" set on all tmp directories?**
Severity	Critical
Testing & Evaluation	

We already know that at least some tmp directories have the sticky bit (see the previous test). Nonetheless, I want to be sure that all such directories has the sticky bit set. One caveat here is that we cannot know if some application uses a non-standard directory; it will be assumed that all applications will conform to the standard or handle the security problem in another way (one way is to use the calling user's home directory, but then we do not have to deal with world writable directories so the problem disappears by itself). By "standard" I mean subdirectories named tmp, e.g. /var/tmp.

 # find / -type d -name tmp -exec ls -ld {} \;
drwxrwxrwt 2 root wheel 512 Sep 9 03:03 /tmp
drwxrwxrwt 2 root wheel 512 May 16 12:58 /usr/tmp
drwxrwxrwt 3 root wheel 512 Sep 9 03:03 /var/tmp

All found tmp directories had the sticky bit. Test succeeded.

Item #6	**Does anyone except root have write permission in roots path?**
Severity	Critical
Testing & Evaluation	

The first step is to determine what root's path looks like:

 # echo $PATH
/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin:/usr/X11R6/bin

The second is to feed this output to **find** (with the colon replaces to spaces as this is the delimiter must applications currently favour). What I tell find to do is the following: find all files or directories in the list (recursively) which either are not owned by root or have a permission set which will allow the group or world to modify them.

 # find /sbin /usr/sbin /bin /usr/bin /usr/local/sbin /usr/local/bin /usr/X11R6/bin
\! -user 0 -or -perm 0022 -exec ls -l {} \;
find: /usr/X11R6/bin: No such file or directory

/usr/X11R6/bin could be an issue if root does not own the /usr so we need to check this (it gives the possibility for another id to create a directory which already is in root's path):

 # ls -ld /usr
drwxr-xr-x 17 root wheel 512 May 15 21:16 /usr/

But no, root is the owner of /usr and the only one with write permission.
No issues found; test succeeded.

Item #7	**Is any exported file system allowing root privileges?**
Severity	Critical
Testing & Evaluation	
The information we want will be in the /etc/exports file. **$ more /etc/exports** /etc/exports: No such file or directory It is very hard to export a file system when /etc/exports does not exist. Test is successful.	

Item #8	**Is NFS running?**
Severity	Important
Testing & Evaluation	
A **showmount -e** will attempt to query localhost for all entries in the /etc/exports: **$ showmount -e** RPC: Port mapper failure showmount: can't do exports rpc The call failed, meaning the NFS server did not respond. Just to be sure, I will do an **ps** and check for NFS that way too (**mountd** is required in order to provide NFS functionality): **$ ps -aux | grep mountd**	

$ No contradicting result. Item successful.

Item #9	Is any exported file system allowing read-write?
Severity	Important
Testing & Evaluation	
So, let us take a look at the /etc/exports file. **$ more /etc/exports** /etc/exports: No such file or directory If no file systems are exported at all, I fail to see how any system could be exported read-write. That must mean that the test is successful.	

Item #10	Are file systems that only contain user or application data mounted with nosuid and nodev to protect against rogue programs?
Severity	Important
Testing & Evaluation	
The mount utility is all we need: **$ mount** /dev/ad0s1a on / (ufs, local) /dev/ad0s1e on /system2 (ufs, local) procfs on /proc (procfs, local) ad0s1e is a data partition. Nonetheless, it is not mounted nodev or suid. The test must unfortunately be considered failed.	

6.5 Programs & Applications

Item #11	Are there any files that have excessive suid/sgid flag?
Severity	Critical
Testing & Evaluation	
First, let us find any setuid files: **# find / -perm -4000** /bin/rcp /sbin/ping /sbin/ping6 /sbin/route /sbin/shutdown /usr/bin/cu /usr/bin/uucp	

```
/usr/bin/uuname
/usr/bin/uustat
/usr/bin/uux
/usr/bin/man
/usr/bin/at
/usr/bin/atq
/usr/bin/atrm
/usr/bin/batch
/usr/bin/chpass
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/ypchpass
/usr/bin/ypchfn
/usr/bin/ypchsh
/usr/bin/keyinfo
/usr/bin/keyinit
/usr/bin/lock
/usr/bin/login
/usr/bin/passwd
/usr/bin/yppasswd
/usr/bin/quota
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/su
/usr/bin/crontab
/usr/bin/lpq
/usr/bin/lpr
/usr/bin/lprm
/usr/libexec/sendmail/sendmail
/usr/libexec/uucp/uucico
/usr/libexec/uucp/uuxqt
/usr/sbin/mrinfo
/usr/sbin/mtrace
/usr/sbin/sliplogin
/usr/sbin/timedc
/usr/sbin/traceroute
/usr/sbin/traceroute6
/usr/sbin/ppp
/usr/sbin/pppd
```

The number of matches is way too high (46 to be exact). This issue has obviously been overlooked by the administrator. What about setgid files then?

```
# find / -perm -2000
/bin/df
/sbin/ccdconfig
/sbin/dmesg
```

/sbin/dump
/sbin/rdump
/sbin/restore
/sbin/rrestore
/usr/bin/cu
/usr/bin/uustat
/usr/bin/fstat
/usr/bin/ipcs
/usr/bin/netstat
/usr/bin/nfsstat
/usr/bin/systat
/usr/bin/top
/usr/bin/vmstat
/usr/bin/wall
/usr/bin/write
/usr/bin/lpq
/usr/bin/lpr
/usr/bin/lprm
/usr/libexec/uucp/uucico
/usr/libexec/uucp/uuxqt
/usr/sbin/ifmcstat
/usr/sbin/iostat
/usr/sbin/pstat
/usr/sbin/swapinfo
/usr/sbin/trpt
/usr/sbin/lpc

Numbering 29 setgid files, I do not think any better of this part. It is not necessary to go through each file and check it (remember, it is a web server). Rather, it should be enough to mention that yppasswd is setuid even though NIS is not in use. Item failed.

Item #12	**Are any programs being unnecessarily run as root?**
Severity	Critical

Testing & Evaluation

As always when it comes to finding processes, ps is my tool of choice.

$ ps -aux

```
USER    PID %CPU %MEM   VSZ  RSS  TT  STAT STARTED      TIME COMMAND
auditor 57085  0.0  0.1   420  240  p0  R+    7:59PM   0:00.00 ps -aux
root        1  0.0  0.1   528  300  ??  ILs  16Aug01   0:00.03 /sbin/init --
root        2  0.0  0.0     0    0  ??  DL   16Aug01   0:03.60 (pagedaemon)
root        3  0.0  0.0     0    0  ??  DL   16Aug01   0:00.00 (vmdaemon)
root        4  0.0  0.0     0    0  ??  DL   16Aug01   0:17.30 (bufdaemon)
root        5  0.0  0.0     0    0  ??  DL   16Aug01   4:15.87 (syncer)
root       26  0.0  0.0   208   92  ??  Is   16Aug01   0:00.00 adjkerntz -i
```

```
root    146  0.0  0.2   924  628  ??  Ss   16Aug01   0:07.99 syslogd -s
root    151  0.0  0.3  1248  856  ??  S<s  16Aug01   1:46.14 ntpd -p /var/run/ntpd.pid
root    170  0.0  0.3   968  712  ??  Ss   16Aug01   0:12.41 /usr/sbin/cron
root    173  0.0  0.6  2148 1504  ??  Is   16Aug01   0:04.17 /usr/sbin/sshd
root    216  0.0  0.2   936  636  v0  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv0
root    217  0.0  0.2   936  636  v1  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv1
root    218  0.0  0.2   936  636  v2  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv2
root    219  0.0  0.2   936  636  v3  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv3
root    220  0.0  0.2   936  636  v4  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv4
root    221  0.0  0.2   936  636  v5  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv5
root    222  0.0  0.2   936  636  v6  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv6
root    223  0.0  0.2   936  636  v7  Is+  16Aug01   0:00.01 /usr/libexec/getty Pc ttyv7
root    291  0.0  1.2  4564 3016  ??  Ss   16Aug01   1:18.34 /usr/local/sbin/httpd -DSSL
nobody  292  0.0  1.3  4948 3420  ??  I    16Aug01   0:18.82 /usr/local/sbin/httpd -DSSL
nobody  293  0.0  1.3  4976 3424  ??  I    16Aug01   0:19.16 /usr/local/sbin/httpd -DSSL
nobody  294  0.0  1.3  4968 3432  ??  I    16Aug01   0:19.21 /usr/local/sbin/httpd -DSSL
nobody  295  0.0  1.3  4940 3424  ??  I    16Aug01   0:19.18 /usr/local/sbin/httpd -DSSL
nobody  296  0.0  1.3  4980 3452  ??  I    16Aug01   0:19.37 /usr/local/sbin/httpd -DSSL
nobody  297  0.0  1.3  4972 3432  ??  I    16Aug01   0:19.01 /usr/local/sbin/httpd -DSSL
nobody  298  0.0  1.3  4940 3440  ??  I    16Aug01   0:18.98 /usr/local/sbin/httpd -DSSL
nobody  299  0.0  1.3  4956 3428  ??  I    16Aug01   0:19.07 /usr/local/sbin/httpd -DSSL
nobody  300  0.0  1.3  4988 3464  ??  I    16Aug01   0:18.99 /usr/local/sbin/httpd -DSSL
nobody  301  0.0  1.3  4948 3428  ??  I    16Aug01   0:19.15 /usr/local/sbin/httpd -DSSL
root  56939  0.0  0.7  2232 1804  ??  S     6:11PM   0:00.60 sshd: auditor@ttyp0 (sshd)
auditor  56940  0.0  0.4  1060  904  p0  Ss    6:11PM   0:00.07 -bash (bash)
root      0  0.0  0.0     0    0  ??  DLs  16Aug01   0:01.45 (swapper)
```

System processes like init must of course be owned by root. We are obviously running SSH, and that one requires root as well. NTP requires root to be able to do any kernel time adjustments. And httpd is run as nobody (except for the master daemon which respawn new child processes). I recall hearing someone on a mailing list saying that httpd should be run as a dedicated account, and that may be a good idea, but my main concern here is that it is not root.
Test successful.

6.6 Patches

Item #13	**Are all security related patches applied?**
Severity	Critical
Testing & Evaluation	
So, FreeBSD patches may be applied in three (and possible more) ways. Either as a source patch (in this case, the best way to check if patches are installed would be to make sure the MAC time for the file in question is later than the advisory release date), or as a binary patch applied with pkg_add (this feature seems to be experimental, though). In the latter case, all patches could easily be listed with the pkg_info utility. A third way is when an old package is revoked and a new is provided. Here the MAC trick would probably be the most useful.	

My first step is to make sure how patches are applied on this particular system. If we are using source patching, then it makes sense to have the sources around, right?

$ ls -l /usr/src
total 2
drwxr-xr-x 2 root wheel 512 May 16 15:34 local
drwxr-xr-x 46 root wheel 1024 May 15 20:17 sys

Well, we may be able to patch our kernel and the local distribution but not much else. I will assume the other method is in use although experimental (patches should be applied on non-production systems first anyway). Then to the real business. I will randomly take two advisories from the last three months and see the system still is vulnerable (were this not an "proof of the concept" audit or I had more time available, I would of course check all patches).

Adv. 1, 2001-09-04) FreeBSD-SA-01:59.rmuser.v1.1.asc

$ ls -lT /usr/sbin/rmuser
-r-xr-xr-x 1 root wheel 15258 Apr 21 11:09:53 2001 /usr/sbin/rmuser

Hmm...The modification date does not get my hopes running high; what about an package?

$ pkg_info
apache+mod_ssl-1.3.19+2.8.2 The Apache 1.3 web server with SSL/TLS functionality
bash-2.05   The GNU Bourne Again Shell
lynx-2.8.3.1   A non-graphical, text-based World-Wide Web client
mm-1.1.3   Shared memory library for applications with pre-forked proc

A few packages, but I do not see anything about "rmuser" in any of them. My conclusion is that this patch not is installed.

Let us try another:

Adv. 2, 2001-07-17) FreeBSD-SA-01:48.tcpdump.asc

$ ls -lT /usr/sbin/tcpdump
-r-xr-xr-x 1 root wheel 250568 Apr 21 11:10:20 2001 /usr/sbin/tcpdump

Why do I have a bad feeling about this one, too...

And from the previous output from pkg_info, no tcpdump package is installed either.

This test is failed. Patches are not applied.

6.7 Accounts & Passwords

The master password file (/etc/master.passwd) will be used by several items in this section. Rather than printing the same information multiple times, I will handle it once and for all here in the introduction. Note that I verified the file to be the same when doing every item. This was done by using **md5** as well as manual review. When an item is dependant on this information, I will point that out.

So, first I want to know the md5 hash of the /etc/master.passwd file. At all subsequent tests, **md5** will be run again and the hashes will be compared. This is just to be sure nothing strange occurs with the file (it should not, but one never knows).

```
# md5 /etc/master.passwd
MD5 (/etc/master.passwd) = 229f4a12f60b77b6dcddeb0d82f37e5d
```

To actually list the master password file, **more**, **less** or **cat** would all suffice. I will not explain the differences between the utilities since I considerer it basic Unix concepts.

```
# more /etc/master.passwd
# $FreeBSD: src/etc/master.passwd,v 1.25 1999/09/13 17:09:07 peter Exp $
#
root: $1$qeVAYHWL$XLIN7LiBdxdbwh9JnePyz0:0:0::0:0:Simon T:/bin/sh
toor:*:0:0::0:0:Bourne-again Superuser:/root:
daemon:*:1:1::0:0:Owner of many system processes:/root:/sbin/nologin
operator:*:2:5::0:0:System &:/:/sbin/nologin
bin:*:3:7::0:0:Binaries Commands and Source,,,:/:/sbin/nologin
tty:*:4:65533::0:0:Tty Sandbox:/:/sbin/nologin
kmem:*:5:65533::0:0:KMem Sandbox:/:/sbin/nologin
games:*:7:13::0:0:Games pseudo-user:/usr/games:/sbin/nologin
news:*:8:8::0:0:News Subsystem:/:/sbin/nologin
man:*:9:9::0:0:Mister Man Pages:/usr/share/man:/sbin/nologin
bind:*:53:53::0:0:Bind Sandbox:/:/sbin/nologin
uucp:*:66:66::0:0:UUCP pseudo-user:/var/spool/uucppublic:/usr/libexec/uucp/uucico
xten:*:67:67::0:0:X-10 daemon:/usr/local/xten:/sbin/nologin
pop:*:68:6::0:0:Post Office Owner:/nonexistent:/sbin/nologin
nobody:*:65534:65534::0:0:Unprivileged user:/nonexistent:/sbin/nologin
auditor:$1$GAaYndVH$zQ4DdVDUYpoMr.H7fvEw40:1000:1000::0:0:Audit User:/bin/sh
```

Item #14	Has all accounts non-empty password fields?
Severity	Critical
Testing & Evaluation	
The only good way to find out is to have a look at the /etc/master.passwd file (see section introduction). It is obvious that no account has an empty password field (if you want to brute force the root	

password, go ahead, but please note that the real hash is replaced by an entirely different one). Test successful.

Item #15	Is it possible to login as system accounts?
Severity	Critical
Testing & Evaluation	
Again we have to rely on /etc/master.passwd to tell us about the password fields. The contents of the file can be found in the section introduction. As can be seen, all system accounts have a star as password hash. Test successful.	

Item #16	Is the minimum password length too short?
Severity	Critical
Testing & Evaluation	
We will take a look at /etc/login.conf to see if the minimum password length is set. **$ grep minpasswordlen /etc/login.conf** $ The setting does not exist. This means the default[13] value of 6 is used. Test OK.	

Item #17	Are strong passwords enforced?
Severity	Critical
Testing & Evaluation	
Again we will have to check in /etc/login.conf, but this time we will be looking for the passwd_format and mixpasswordcase parameters: default:\ :passwd_format=md5:\ Well, it seems that the mixedpasswordcase directive is missing. This is not necessarily a bad thing since the manual page login.conf(5) says the default value is true. And the password format is md5. Test successful.	

Item #18	Are all user and group IDs unique?
Severity	Critical
Testing & Evaluation	
In addition to the information in /etc/master.passwd, we also need to have a look at what /etc/group contains: **$ cat /etc/group** wheel:*:0:root,auditor daemon:*:1:daemon kmem:*:2:root	

[13] Manual page: login.conf(5)

```
sys:*:3:root
tty:*:4:root
operator:*:5:root
mail:*:6:
bin:*:7:
news:*:8:
man:*:9:
games:*:13:
staff:*:20:root
guest:*:31:root
bind:*:53:
uucp:*:66:
xten:*:67:xten
dialer:*:68:
network:*:69:
auditor:*:1000:
nogroup:*:65533:
nobody:*:65534:
```

So, while all groups are unique, both root and toor have an userid of zero. Test failed.

Item #19	**Is the password database and master file securely kept?**
Severity	Critical
Testing & Evaluation	

The only utility we need for this is **ls**:

```
$ ls -l /etc/master.passwd /etc/spwd.db /etc/passwd /etc/pwd.db
-rw-------  1 root  wheel   1047 May 15 21:16 /etc/master.passwd
-rw-r--r--  1 root  wheel    922 May 15 21:16 /etc/passwd
-rw-r--r--  1 root  wheel  40960 May 15 21:16 /etc/pwd.db
-rw-------  1 root  wheel  40960 May 15 21:16 /etc/spwd.db
```

All files are sufficiently protected. Test is successful.

6.8 Root Access

Item #20	**Is root the only account with id zero?**
Severity	Critical
Testing & Evaluation	

I will use awk and grep to display only id zero accounts:

```
#  cat /etc/master.passwd | awk -F: '{ print $1,$3 }' | grep " 0"
# $FreeBSD 09
root 0
toor 0
```

Identical output will appear when run on /etc/passwd instead. Obviously, there are more than one account that has a zero id. Test failed.

Item #21	Does root's PATH variable contain a single dot?
Severity	Critical
Testing & Evaluation	
The PATH variable is a shell built-in and is best viewed with the **echo** command. **# echo $PATH** /sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin:/usr/X11R6/bin No single dot exists in the path. Test succeeded.	

6.9 Network Configuration

Item #22	Are source routed packets ignored?
Severity	Critical
Testing & Evaluation	
sysctl is used to get the current status concerning source routing. **$ sysctl net.inet.ip.accept_sourceroute** net.inet.ip.accept_sourceroute: 0 **$ sysctl net.inet.ip.sourceroute** net.inet.ip.sourceroute: 0 Both in and outgoing source routing is disabled. Test successful.	

Item #23	Are there unneeded points of access like extra network interface cards, or modems?
Severity	Critical
Testing & Evaluation	
When it comes to showing network interface information, ifconfig is our friend: **$ ifconfig -a** xl0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500 inet 10.0.1.12 netmask 0xffffff00 broadcast 10.0.1.255 ether 00:50:ad:6d:8c:da media: autoselect (100baseTX <full-duplex>) status: active supported media: autoselect 100baseTX <full-duplex> 100baseTX 10baseT/UTP <full-duplex> 10baseT/UTP none lp0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500 lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384 inet 127.0.0.1 netmask 0xff000000	

The only active interface (loopback not included for obvious reasons) is xl0 which is the system's normal network connection. The lp0 interface is used for parallel port communication[14]. This could be an issue, but two mitigating factors are that the interface is down and a physical check on the system shows that nothing is connected. Test successful.

Item #24	Are any network interface cards in promiscuous mode?
Severity	Critical

Testing & Evaluation

We will deploy ifconfig once more:

```
$ ifconfig -a
xl0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 10.0.1.12 netmask 0xffffff00 broadcast 10.0.1.255
        ether 00:50:ad:6d:8c:da
        media: autoselect (100baseTX <full-duplex>) status: active
        supported media: autoselect 100baseTX <full-duplex> 100baseTX 10baseT/UTP
          <full-duplex> 10baseT/UTP none
lp0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet 127.0.0.1 netmask 0xff000000
```

No interface is in promiscuous mode (assuming that the system not is compromised, but we have already checked for that). Test successful.

Item #25	Is ssh used instead of telnet?
Severity	Critical

Testing & Evaluation

I need to know whether inetd is running (notice that FreeBSD uses GNU style **ps**):

```
$ ps -aux | grep inetd
$
```

Ok, it is not. I do not need to take /etc/inetd.conf into account. Instead, I proceed by listing all active connections:

```
$ netstat –a
Active Internet connections (including servers)
Proto Recv-Q  Send-Q  Local Address       Foreign Address      (state)
tcp4       0      20  auditsys.ssh        10.0.1.100.1067      ESTABLISHED
tcp4       0       0  *.http              *.*                  LISTEN
tcp4       0       0  *.https             *.*                  LISTEN
tcp4       0       0  *.ssh               *.*                  LISTEN
udp4       0       0  localhost.ntp       *.*
udp4       0       0  auditsys.ntp        *.*
```

[14] The FreeBSD networking FAQ shows how lp0 can be used to connect two systems.

```
udp4       0      0  *.ntp                  *.*
udp4       0      0  *.syslog               *.*
Active UNIX domain sockets
Address  Type   Recv-Q Send-Q    Inode     Conn     Refs  Nextref  Addr
ccc4afc0 dgram       0      0        0  ccc45fc0  0     0
ccc45fc0 dgram       0      0           ccc44d40  0     ccc4afc0  0 /var/run/log
```

We can see that SSH is enabled, as well as HTTP and HTTPS (as previously mentioned, it is a web server). The established connection is me running the test remotely, over SSH. And there are no suspicious looking ports either (telnet may be moved to another port for some reason or another). The test is successful.

6.10 System logs

Item #26	Is syslogd configured to drop packets from other hosts?
Severity	Important
Testing & Evaluation	
ps, our long and trusted friend: **$ ps -aux | grep syslog** root 146 0.0 0.2 924 628 ?? Is 16Aug01 0:08.03 syslogd –s -s means syslog will refuse messages sent from other hosts. This is an webserver, so it is fine with me. Test successful.	

6.11 Procedures & Policy

Item #27	Is the screen locked or logged out at console when no one is using it?
Severity	Critical
Testing & Evaluation	
A physical inspection shows a system where no vty is logged on. During the interview, the administrator, Bob, tells us that this is common practice for this system. Test successful.	

Item #28	Is there a contact list for all users on the system?
Severity	Important
Testing & Evaluation	
An interview with Bob tells us that such an document should exist. A list with usernames are found in a safe in the office. Test successful.	

6.12 Audit Summary

As probably already noted, I omitted the optional items and focused on the critical and important ones instead. The results is most easily shown in a small table:

	Critical Items	Important Items
Passed Items	**15**	**6**
Failed Items	**5**	**2**

So we have 5 failed critical items. The limit was 0. The failures numbered from bad patch routines to en excess of setuid/setgid files. This system will not get a passing grade. Administrator Bob will have to spend some time fixing the system. And he should start taking patches seriously. And still, only 28 of 72 items were run. It probable more item would have failed if the full list had been used.

7 EVALUATION & FUTURE IMPROVEMENTS

Overall, I think the presented checklist is quite useful. An IS (Information System) which pass all tests should be pretty hard to compromise. That is of course not to say it is perfect. During the real world audit (see chapter 6), I found some weaknesses in it. One thing is that it is very hard for a system to pass, even though it may be secure enough according to the security policy of the organisation in charge of the system.

In this chapter I will present the weaknesses found, and as such they are also items that should be corrected in the future should anyone have the time and resources for it. As this is an initial version, there should be a lot of possible improvements.

7.1 Accounting

Any information concerning accounting is absent. As previously mentioned, FreeBSD allows all processes to be accounted. The problem is, short of writing a kernel module (which I am no good at), I am unable to find any method to check whether it is enabled or not. In the kernel accounting source code (available from the installation distribution), the syscall (acct) for enabling or disabling the accounting is defined. In the same file, a function to do just the check I want exist. But Alas! It is not a syscall so unless in kernel space, I am out of luck. At this point, I would also like to give credit to a fellow named Elias Norberg for helping me around in the kernel code (I usually have problem enough understanding other peoples code, and that is without being in kernelspace). I do not understand why it is implemented this way (and I could be wrong too), but it may explain why I were unable to find any application that would give me the answer I wanted. The program used for displaying the last accounted commands, **lastcomm**, will kindly report "lastcomm: /var/account/acct: No such file or directory". Unfortunately, this does not tell us that accounting is disabled. It just tells us accounting is not done to the default file. And the command **accton** which will enable accounting if a filename is given as an command line argument (the file must exist) does not use the default path if you omit the filename. No, in that case it disabled accounting, were it enabled, no questions asked (no, I do not favour the "Are you sure?" mentality). If there are any good reasoning behind this, it escapes me.

So, if anyone find out how check for accounting, I'm willing to hear. But I do not want to load anything in my kernel for doing this.

7.2 Severity & Categories

The balance between critical, important and optional items should most likely be changed. Some critical items should probably be important and vice versa. And perhaps additional severity levels should be added. And the criteria's for how many failed items of each severity type that may exist for a certain system must be changed.

Except for the severities that exist, I have been considering another type of categorising that would exist in parallel with the current. Each item, in addition to the severity level, would also have a field which could be protective (e.g. firewalls), detective (system logs) or reactive (backups). If this sounds something like the Time Based Security class taught by SANS, well, that is correct. It is. What I hope this would accomplish is increased "portability" between different organisations; some may focus on very quick detection and reaction while others may lay their effort in protection. Of course, some items would be required a pass by all standards. Excusing a failure to apply a certain security patch by saying detection is prioritised will not be tolerated unless the system in question is a honey pot.

These items will probably need to be changed many times; I do not think one audit is enough to set these parameters right.

7.3 Procedures

I still have not decided if I feel procedures should be covered at all in a technical checklist. In a full audit, all policies and procedures will be reviewed in any case. But some subjective items comes very close to pure policy items, so it may be that some procedural items should exist. If so, one such item that does not exist is whether there are any incident handling routines for the system. Does the administrator know what to do in case of an intrusion?

7.4 Devices

To my horror I find that device security is completely forgotten! This is one thing that must be added. Just to mention one check to be done: anything in /dev should either be a block or character device (except the MAKEDEV file). Anything else should be regarded with suspicion since it could mean everything from a backdoor (there are so many files in /dev, and people tend to be afraid to touch things in there they do not know what it is, so is a popular hiding place for various things) to a misconfigured system.

7.5 X-Windows

Another omitted section. Well, most FreeBSD systems are probably going to be used as servers anyway, but now and them one may find those rare persons who likes pain enough to use FreeBSD on workstations. It works, that is true, but it takes so much time (especially on laptops). As **X11R6** is used, I assume it uses the same configuration as most GNU/Linux systems; the security part should also be similar.

7.6 Application Security

Even though I do not think it should be included in the this checklist, I would like to have an additional checklist for some of the most common applications ported to FreeBSD. Among these (but of course not limited to) are **BIND** and **Apache**. These kind of services is certain to be found on live systems. The process of evaluating the security of these applications are not as simple as checking whether they are using the last patch level. A misconfiguration in BIND could have severe implications on an organisation (consider allowing zone transfers from anyone).

A common problem (it is even on the SANS Top Ten) that almost is worth a chapter (or even books) of its own is CGI scripts. Auditing CGI scripts is usually the same thing as doing source code review, often in Perl or C. For those who do not have time to review all scripts (they may be very poorly documented), there exists the possibility to throw **Nessus** at it.

Well, this section is strictly speaking not OS Auditing, but a badly written CGI-script executed by a misconfigured web server will root your box as easily as an old version of **rpc.mountd**. And both the OS and web server could still use the current patch level! So these kind of checks should be done, and it should be mentioned in the report, but this checklist is long enough as it is, so I still advocate a separate list for it.

7.7 inetd issues

Then there is this dirty business about **inetd**. This ancient legacy daemon from the past. Well, enough joking. Although I personally prefer to run without **inetd**, I understand that some systems will require it. I also understand that **tcp-wrappers** by Wietse Venema may be favoured over **ipfilter** or **ipfw**. And **xinetd** may be favoured over **tcp-wrapped inetd**. Unfortunately, the checklist does not understand this; it is something that needs fixing.

7.8 Firewall Issues

One shortcoming I also found was that I do not test if there are any sensible firewall rules. I only want to know if firewalls are supported. On the other hand, to be able to test a firewall rulebase, the auditor must be familiar with firewalls and the theory behind them (which also means a solid knowledge of TCP/IP). But it should be included in any case. A firewall that accept all packets is just a slow router (and routers usually have some kind of ACL so change that to a slow, wide-open router).

7.9 Miscellaneous

Early in the document I do mention that FreeBSD have its own security and health check monitoring script. But I never got around to explain it further.

And I never talked about the wtmp. There are also other logging specific issues with FreeBSD that one should do more discussion about.

8 CONCLUSION

Well, perhaps "Outro" would be a better title for this chapter. It just felt so wrong to cut it without some finishing words. Unfortunately, I do not have anything of real substance to say more than I learned a lot about FreeBSD as well as auditing by doing this paper. It has been fun at times, bloody at others. It is my hope that it will be useful for someone else and that the work for getting good auditing guidelines for FreeBSD continues.

9 APPENDENCIES

9.1 References

9.1.1 Security Sites

Security Focus - http://www.securityfocus.com
Packetstorm Security - http://www.packetstormsecurity.com
The SANS Institute – http://www.sans.org
Naval Surface Warfare Center, Dahlgren Division – http://www.nswc.navy.mil/ISSEC/

9.1.2 Security Papers

Koum, B. Jan. "FreeBSD Security How-To". URL: http://people.freebsd.org/~jkb/howto.html.

"System Administration". Frequently Asked Questions for FreeBSD 2.X, 3.X and 4.X. URL: http://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/admin.html.

"Networking". Frequently Asked Questions for FreeBSD 2.X, 3.X and 4.X. URL: http://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/networking.html

Based on work by Dillon, Matthew. "Security". FreeBSD Handbook. URL: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/security.html.

Marchany, Randy. "VT IS Security Audit Solaris 2.5.1 Test Procedures" URL: http://courseware.vt.edu/marchany/SunAudit/audit.html.

UnixTools.com. "Unix Computer Security Checklist". URL: http://www.unixtools.com/securecheck.html.

Naval Surface Warfare Center, Dahlgren Division Information Systems Security Office. "Risk Assessment/Countermeasure Analysis/Security Test and Evaluation (ST&E) for Unix Computer Systems" version 3.1. 1 July 1999. URL: http://www.nswc.navy.mil/ISSEC/Form/AccredForms/acc_part2_unix.html (Version 3.2 7 September 2001).

Kuethe, C. "ID FAQ: What are telltale signs that a Unix system has been compromised?" URL: http://www.sans.org/newlook/resources/IDFAQ/unix_signs_compromised.htm

9.1.3 Books

Marchany, Randy. Understanding and Auditing Information Systems.
The SANS Institute 2001

Kolde, Jennifer & Green, John Advanced Systems Audit and Forensics.
The SANS Institute 2000, 2001

9.2 Source Code

9.2.1 Sample source code for determining bpf support

```
/*
   bpftest.c:
   Test program to determine whether Berkeley Packet Filter
   is enabled or not. Written by Patrik Sternudd 2001.
   This program is provided as is in hope that it will
   be useful. No warranties are given, use at your own risk.

   Code adjusted to look good in winword, meaning you'll have a bad
   time if you copy and paste it. I recommend indent or vi to make
   it readable on Unix again. And you might wish to look over the
   printf's because it looked horrible when it broke the line in winword,
   so rather than have that, I split some printf's into two.
*/

#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <unistd.h>

int main(){

   int device, minor, rval;
   char path[255];

   if( geteuid() != 0){
      printf("You need to be root.\n");
      return -1;
   }

   for( minor = 0; minor <=255; minor++){

      sprintf(path,"/dev/bpf%d",minor);
      printf("Trying '%s'...",path);
      device = open(path,O_RDONLY);

      if( device == -1){

            if( errno == EBUSY)

               printf("file in use.\n");

               else if( errno == ENOENT){

                  printf("file does not exist. Breaking.\n");

                  printf("If all devices were busy, you probably have");
                  printf("something else to worry about ;)\n");

                  return -1;
               }else{
```

```
                printf("error code %d\n",errno);
                printf("This most likely means bpf is disabled.\n");

                return -1;
             }
          }else{

             printf("successful!\n");
             printf("It appears bpf is enabled.\n");
             close(device);
             return 1;
          }
   }

   printf("You should probably check your kernel at this point, because");
   printf(" you have more than 255 available packet filtering devices.\n");
   return 0;

}
```

Last Updated: November 25th, 2020

Upcoming Training

SANS Cyber Defense Initiative 2020	**,**	**Dec 14, 2020 - Dec 19, 2020**	**CyberCon**
SANS Cyber Security West: March 2021	**,**	**Mar 15, 2021 - Mar 20, 2021**	**CyberCon**
SANS London July 2021	**, United Kingdom**	**Jul 05, 2021 - Jul 10, 2021**	**CyberCon**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**