Global Information Assurance Certification Paper

Deploying a Secure SuSE Linux Enterprise Server

Case Study in Information Security
GSEC Certification Practical
Case Study Option 2
V1.4b

Author: Michael Delabar
9 March 2004

Abstract

As security professionals we are faced with many challenges on a daily basis. One such challenge is balancing usability with security. We are faced with creating robust systems which can be easily maintained, while providing maximum availability, scalability, and ease of use to many people across an enterprise computing environment. One of the most fundamental choices we are faced with is which operating system to deploy on our core systems. This paper will discuss, how Linux can lend itself to the enterprise when called upon, and will do so in a secure manner.

For this case study I chose a recent Linux deployment in my organization that I was the technical lead on. I was given the task of deploying a secure database server which would house business related data. For various reasons I chose SuSE Linux Enterprise Server version 8 as the operating system. One such reason is detailed in a press release by SuSE, "The SuSE Linux Enterprise Server 8 with Service Pack 3 on the IBM eServer family platforms has achieved the world's first Common Criteria CAPP/EAL3+ certificate for an Open Source operating system..." This distribution was chosen mostly because of its stability and scalability, but, also, because of its credentials as a secure platform and out of the box compatibility with the IBM dual-AMD64Opteron based solution that was designated for the deployment.

I will demonstrate through this case study that although Linux is a secure platform to deploy, there are steps that need to be taken to get a distribution ready for business in high-risk realms such as the internet.

1. Before

1.1 The problem

The problem/challenge that I faced was to build and deploy a database server that would be accessed over the public internet for external client interaction purposes. The company I work for is a large international media firm with offices in 110 countries and needed to aggressively enter the B2B market space in a global fashion. As with all things in the web space, time to market was critical. The business units driving the project had given me a deadline of 2 weeks to complete the entire process, therefore, the approach had to be streamlined and portable enough to be passed to other IT teams in downstream regions.

Aside from the business problem at hand, there were hidden IT agenda's that we wanted to achieve within the same process. The hardening procedure I was to develop had to seamlessly apply to any server providing web-related services. In particular, the next set of targets would be publicly facing web and ldap servers. While this documented procedure will apply to all of these types of servers, we will concentrate on this database server as our example.

1.2 The Risk

As a global corporation doing business over the internet, risk analysis is an inherent component of any deployment that IT performs. In this scenario, we are faced with a database server that will house client related data as well as internal data for a web based collaboration application. Since this database will contain client data, security is of the utmost concern. Some of our B2B partners require Service Level Agreements (SLA) that have stringent security measures built in. To comply with some of these we need to deploy our services in the most secure fashion possible. The overall risk of these deployments comes with a potential and direct impact on the business as a whole.

1.3 The Team

As with most companies, we were faced with many personnel changes in recent times due to a sluggish economy. Not all positions were able to be re-hired and thus our IT department was reconfigured for "maximum efficiency". That being the case I was tasked to see this project through in a multi-faceted role. I took the technical lead, implementation, and project lead roles and spearheaded them all into 1 joint effort, and upon completion passed the final results to my director, and other colleagues.

1.4 The Methodology

Since the entire framework of this project is accessible from both internal and external networks I will need to consider this in my vulnerability assessment, as it can impact the varying degree of security possessed therein. Based on previous experience as well as knowledge gained in my GSEC training I chose to attack the situation with multiple tools from the open-source community. The approach I am going to take is as follows:

1. Identify possible vulnerabilities.
2. Apply hardening procedures
3. Enhance internal server security.

2. During

2.1 Initial Assessments

SuSE distributes a fairly secure operating system right out of the box, however, in this case study we will attempt to explain that the layered approach to security is the best model to follow in order to ensure that the integrity of the environment can be maintained throughout various levels. A good starting point of where you stand to the outside world, is to port scan yourself to see what ports are available for socket connection on your system. Port scanning is considered intrusive and should only be done in your own network with coordination of

resources responsible for network security. Nmap[1] is a good tool to use for this task and the syntax for running the scan is as follows:

```
bash# nmap 192.168.1.1

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.1:
(The 1598 ports scanned but not shown below are in state:
closed)
Port       State       Service
22/tcp     open        ssh
25/tcp     open        smtp
111/tcp    open       sunrpc
```

This scan resulted in the above ports being open to the outside world. Port 22 is a necessary port for the ssh service. All traffic along this port will be encrypted, so leaving it open is generally ok. Port 25 is used for the SMTP protocol and is used for sending and receiving mail, on some systems this port can remain open but, we will see later on that a good security measure is running the SMTP service in daemon mode is much more secure (more on this later). Port 111 is used for rpc or remote procedure call. This is used primarily for the portmapper which can be considered a security risk due to its many buffer overflow vulnerabilities. It is sometimes necessary for port 111 to stay open if a need arises. Such a need could be file sharing with NFS, which registers itself with the portmapper. If the portmapper is turned off then NFS cannot communicate with the outside world and NFS sharing/mounting would fail. The general rule of thumb which holds true in this case is if you are not using a service turn it off.

Turning unwanted services off can be accomplished in different ways. Some flavors of linux have a service management program called chkconfig, which is a system level script that starts and stops services and can also change the various runlevels that it will start in. SuSE which is the operating system that we are using here uses chkconfig to manage these services and it is the easiest way to disable them. The first step is to get a list of all services and the runlevels that they start in. This can be accomplished by typing: `chkconfig -l` . After I analyzed the list of services, I determined what services were necessary and turned off the ones that were not. Some examples of unwanted services in this case would be ldap, nfs, and the portmapper. Another good measure to take here is to make sure that the server is booting into the correct runlevel. This can be done by checking the /etc/inittab file. In this file there is a setting that controls the default runlevel of the server, you should make sure that if your goal is to not have a “graphical login” (X11) that you are entering runlevel 3 upon startup. The line in the file should read :

`::3::default::` ← 3 is the default runlevel

[1] http://www.insecure.org/nmap

Another good practice is to disallow the root account the ability to login remotely through ssh. The reason that this makes the box more secure is two-fold. First, in order to remotely compromise the machine, the attacker would need access to a user account. After accomplishing this, then they could proceed to try to gain root access. The second reason for this is to force users and system administrators to login to the server with their own user account and then su to the root account. This allows the system log (/var/log/messages on linux) to keep track of who is logging in as root. I was able to accomplish this by making the following changes. First, navigate to the /etc/ssh directory, there you will find a file named sshd_config. Open the file for editing with your favorite editor such as vi or emacs, and change the following :

```
# Authentication:

#LoginGraceTime 600
#PermitRootLogin yes     ← change this to no and uncomment
#StrictModes yes
```

After the change is made, you must restart the ssh daemon to allow the configuration to reload.

2.2 Hardening

Once the box is ready from an OS and filesystem level, the next step is to begin the hardening process. For my core hardening procedure I chose to implement Bastille. The version that comes with the SuSE Enterprise Linux Server 8 edition is 1.2.0. Newer versions can be found at:http://www.bastille.org, however, for ease of management I chose to use the rpm from the distribution media. Bastille is a hardening utility that is written in perl and it is intended to be a guide through some basic steps that can make your system more secure end to end. When Bastille is run in script mode, it makes some changes to your system that it thinks would benefit you from a security standpoint, this is generally not recommended unless you have prior knowledge that the script will not do damage to your system and/or render it in a useless state. The recommended method of installing Bastille on your system is to use the interactive mode, this will ensure that you, the engineer, are in touch with all of the changes that are being made and also that you have a clear understanding of the change and how it works. Interactive mode can be more time-consuming than the script mode and rightfully so. It will go through a series of questions about your intended use of the system and will adjust your security settings accordingly. I will attempt an overview of the process and will highlight key changes to the system and explain the reasons for and against some of the reasons why they are or are not a good idea to use on this system.

In order to start the Bastille hardening process, I ran the following command:

```
Linux~# InteractiveBastille –c
```

The program starts with a disclaimer which tells you how badly you could destroy your system and generally relinquishes the authors of the program from any and all liability concerning the state of your system after its running I chose to run the program using the Tk interface module, but, there is also a curses based module for servers without X11 installed. On most servers X11 is not required so it is not installed. However, in the event that it is installed you can export your display and use the X11/Tk interface. Incidentally, the entire procedure is documented in this paper and is shown in its entirety in Appendix A.

The first section deals with file permissions and takes you through a series of questions designed to pinpoint how users will be using the machine. For instance if the box is intended to be a database server you would probably treat it differently than if it was to be used as a file server. In this case study we are hardening a machine for use as a database server so the answers given in the interactive mode will reflect as such.

At this point you are presented with questions related to file permissions, some of these questions deal with administration utilities and other permission profiles. It is usually good measure to restrict the access of utilities such as ping, nmap, ifconfig, and fsck, however, it does decrease the level of administrative tasks normal users can do. In our case, the server will never be used by end users so restricting access of these commands to root is a good idea.

Bastille will give you three different profiles to choose from and will describe them as easy , secure and paranoid. I have chosen to use the secure setting to ensure a balance of security and usability. You can edit your choices later by hand if you choose in /etc/permissions.local. After you give the script your answer it will then take you through a series of questions that are tailored to the setting that you chose. The secure setting allows you to do things like disable the SUID status for mount, this is good to do so that only root can mount and unmount drives. Another SUID setting that is disabled in this part of the program is ping. This is a great example of the multi-layered approach to securing a server. By disabling the SUID for ping it ensures that only root can use the ping utility; you can also accomplish a similar result using IPtables rules to disallow outbound ICMP traffic. The one advantage to only killing the SUID is that the root account can still use the command for troubleshooting, while using IPtables to accomplish this would disable the ICMP traffic at the kernel level for all users on the system. Truly it is a matter of what you are comfortable with but, in this case I chose the SUID route.

Next, it takes you through disabling the BSD style r* commands. This is a no-brainer. These commands all use cleartext authentication and also send data in the clear. Typically you could get laughed at in the security community if you leave these turned on. This paper will assume knowledge of why cleartext is inherently insecure in its nature and any further discussion of encryption v. cleartext is really outside the scope of this case study.

Also contained in this section are some other good practice security settings. These include restricting the use of cron to only root, and aging

passwords on the system. Many adminstrators do not understand the importance of password aging. Password strength and aging should go hand in hand in a secure system. Even a very strong password can be cracked with a brute-force attack. The knowledgeable systems administrator will age his passwords in such a way so that the brute force attack against the password will take longer than the aging scheme that is put into place, thus ensuring that the attacker would only be able to crack an old password. Restricting the use of the cron daemon is also fairly important. If any user has access to his or her own crontab, then it stands to reason that they could run programs at any time. If a generic user account is compromised on the system then the attacker could bring a system to its knees just by manipulating system resources with the owned account's crontab. This is why we limit the use of this to only the root account.

Password protecting the GRUB bootloader is next on the script's to do list and is also a definite good idea. Although it limits booting to only the systems administrator that has the root password, it can be a failsafe for when physical access to the system is compromised. For our environment it is considered standard so this option was chosen. On the topic of physical security is the ability to halt the system by pressing the CTRL-ALT-DELETE buttons simultaneously on the server. Bastille gives you the option to "trap" that sequence so that one cannot reboot the machine without the proper privileges to do so. Coupling these 2 measures can greatly enhance protection on the physical side. It is also highly recommended that you password protect "single – user mode" which can give would be attackers a root prompt just by booting into this runlevel. I chose to implement all of the previous mentioned settings in my configuration.

Bastille gives some more service oriented security options next. In the next couple of questions it allows for some great security to be implemented. It gives the option to turn off telnet and also to configure a default-deny list for xinetd and TCP wrappers, this is recommended practice and if you need to modify this later to allow certain services it is ok to do so then. In addition to turning off telnet you may want to turn off the FTP service, FTP is often replaced by SCP and SFTP which will encrypt the traffic instead of sending all of the data in the clear. Many of these options may be redundant since we already turned off services that we didn't need earlier by using chkconfig, however, if there were any services that were questionable earlier this section of the script will address them.

In order to have any legal recourse against a hacker, you have to display a warning message on your system which will let people know that unauthorized use is prohibited. Many companies have their own legal disclaimer and at this point in the process the program will create one for you, or let you enter your own based upon your preference.

Now we move into the part of the script which deals with the system and can allow for protection against various types of attacks. The first system level component that we need to protect is the c compiler. This can be an area of contention both for you, as a systems administrator, and for users of the machine. The fact of the matter is that many attacks can come from generic

accounts, not all attacks go after root. If the user account that is attacked has access to the gcc compiler then they would have the ability to compile arbitrary code on your system. What Bastille can do is to disallow user's access to gcc and allow the root account the ability to compile software. This can be a good measure to not only prevent attacks but, also to control what software is compiled on the machine.

Next, we come across another great example of the multi-layered security approach. It comes in the form of modifying the /etc/security/limits.conf file. By modifying this file you can stop would-be attackers from denial of service (DOS) style attacks. This is done in a few different ways, first, creation of core files or core dumps is turned off, and this stops the attacker from crashing an application and filling up a filesystem with log data. Also, it limits individual file size to 100MB, which is claimed to be large enough for normal system usage. In this case, we are not going to allow this option in our hardening process because the server we are building will require files that are much larger than 100MB in size. Entering values into limits.conf provides us with a good example of the multi-layered approach because we are protecting against DOS attacks through the kernel as well as through DMZ and local firewalls.

The script now asks if it should restrict access to the console to certain users. This is generally recommended unless your environment has specific needs where denying user's access wouldn't make any sense. Most datacenters have a set group of user's that have this privilege and in this case I have restricted access to only them.

Logging is covered next, and is really a very important aspect of system maintenance and security. One feature that I implemented here is the ability to log remotely to a centralized logging server. This is an option if you have many machines that have log data that is going to be processed. Also, you have the option of logging to the virtual terminals TTY7 and TTY8. This can be convenient if you are debugging something and it is ok to put into practice as long as you restrict the TTY access as descried in the earlier section.

This next section of the hardening script is best summed up by the explanation given by Bastille itself, "To make the operating system more secure, we try to deactivate all system daemons, especially those running at a high/unlimited level of privilege. Each active system daemon serves as a potential point of break-in, which might allow an attacker illegitimate access to your system. An attacker can use these system daemons to gain access if they are later found to have a bug or security vulnerability. "[2]

The first section of system daemons that will be analyzed deal with routing. This concept is pretty simple, if the server is going to be a dual-homed machine which will handle routing traffic across multiple networks, then you would want to carefully examine this section and turn on this feature. In our case we are using this server as a database so we will not be relying on it to do any routing for us. I chose to disable routing.

The sendmail daemon is important on many systems. It is what routes your email where it needs to go. If you are a systems administrator it can be

[2] Lasser, John, Beale, Jay et al. "Bastille Linux Interactive Mode"

pretty important to have an operational mail daemon right? Wrong. It is not necessary for sendmail to run in daemon mode. Daemon mode is intended for systems whose only function is to be a mail server or relay. In our case, this server is going to be used as a database server so although it is important for this system to have the ability to send mail and receive mail it is not a service that we want turned on all of the time. The Bastille script will ask you if you want to disable daemon mode for this server and in our case we chose to do so.

The final selection that we get to make in our Bastille hardening process has to do with the /TMP environment. Bastille asks if you want to create a "suitable" /tmp environment every time a user logs in. It accomplishes this by using /etc/profile.d scripts. This can be a good security measure however, for this implementation I chose not to use this feature because our system will not deal with many user accounts.

After all of the configurations are made with the interactive setup mode, your selections are stored in a temporary file which is referenced by the execution script. At this point no changes have been made to the system and you can still back out. A warning message will be displayed and it gives you the option to implement your changes. Once you have committed the changes, it is a good idea to reboot the machine. After the system comes back online, the first thing to do is to check the log file (/var/log/Bastille/error_log) for Bastille to make sure the hardening process occurred with no errors. After I verified this I proceeded to check on all of the changes to the system to make sure that they took effect.

Once I verified the process, I ran through a series of tests to make sure that all of my core services were still functioning to their expected capacity. In this case, mysql was the only core service that was expected, and was functioning in the correct manner.

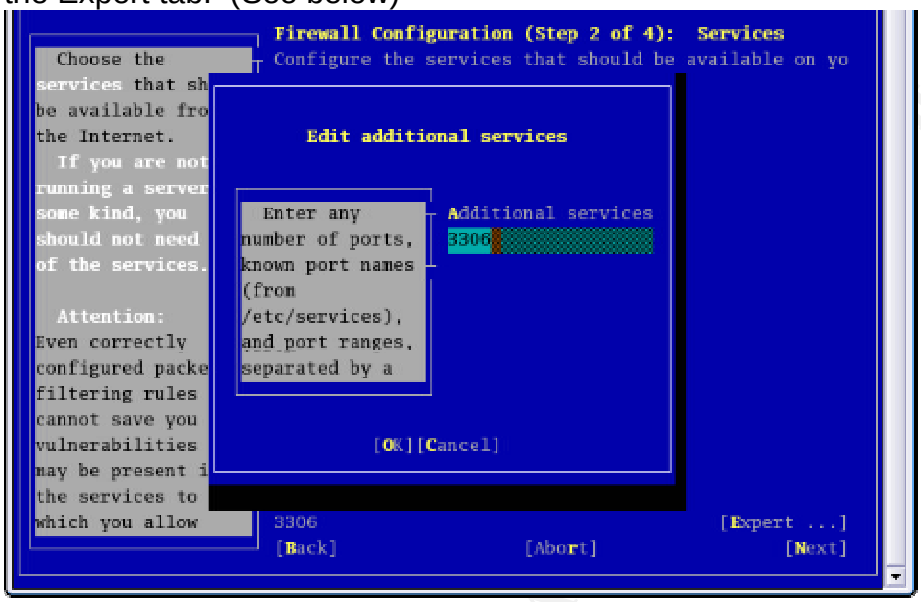
2.3 Internal Server Security

Now that the core of the system has been hardened, it can be considered moderately secure. However, in our production environment we also choose to implement local firewalls on all systems in our DMZ. For the local firewall we will use the standard issue IPtables which is built into the linux kernel. IPtables is a flexible rules based firewall that can get very granular in its configuration. In our case we will be using the SuSEFirewall2 rule set that is provided with the distribution and we will be configuring it using the Yast[3] module for ease of management.

The Yast module is very easy to use and configure. It outlines your configurations and makes all changes to IPtables for you, In this case we want to make sure that the only incoming traffic allowed on this server is on port 22(ssh) and 3306(mysql), this may change at a later date when other products get installed on the machine such as Oracle, at which time modifications will be made using the same utility.

[3] Yet Another System Tool

A full list of all of the screenshots of the firewall configuration is located in Appendix C of this paper. Step one in the firewall configuration is to name the interface that you wish to firewall. In this case I chose eth0 as this is the main interface that will handle incoming connections. Then, after I selected the interface, I choose the ports that I want to allow connections to, in this case again we choose 22 which is a listed port and 3306 which we have to enter manually in the Expert tab. (See below)



After you enter in the ports, you are presented with 2 final options, the first is a page that lets you select from a few features that you would like enabled or disabled. Some of these include traceroute, IPforwarding, masquerading and protecting running services. In this scenario we are going with the default options for this section as they meet our needs. The final screen that configures the local firewall is the logging options page. On this page you are presented with options on which packets you want to log, such as critical accepted and dropped packets or all accepted and dropped packets. In this case study I chose to log only the critical packets. Since the primary job of this server is to function as a database we will not want to log every packet coming at it. If you were configuring a device that was going to act only as a firewall or a firewall/router then you may choose to log all packets.

3.After

3.1 Internal Audit

Now that we have looked at the core hardening procedures and also added a local firewall, I performed a second integrity check on all core systems and services, basically checking all of the work that I had done and checking that all functionality was in tact. The best way to do this is to perform internal and

external audits. In the first scan I will emulate an internal scan from within our DMZ using the nmap utility. This time we will try to give nmap a few more options and make it a little more intrusive in an attempt to derive more information from the host. We will pass it the –T insane for "type : insane". It is best practice to perform this scan from a dedicated auditing host which is located on the same network in order to ensure that you are probing ports that are bound to the correct interface and ensure consistent results.

Results of nmap:

```
linux:/home/linuxuser # nmap – OsS –T insane 10.1.1.30
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on localhost (10.1.1.30):
(The 1599 ports scanned but not shown below are in state:
closed)
Port       State       Service
22/tcp     open        ssh
3306/tcp   open        mysql
No exact OS matches for host (If you know what OS is
running on it, see http://www.insecure.org/cgi-bin/nmap-
submit.cgi).
TCP/IP fingerprint:
SInfo(V=3.00%P=i586-suse-
linux%D=2/17%Time=4032439E%O=22%C=1)
TSeq(Class=RI%gcd=1%SI=438979%IPID=I%TS=100HZ)
TSeq(Class=RI%gcd=1%SI=43897A%IPID=I%TS=100HZ)
T1(Resp=Y%DF=Y%W=7FFF%ACK=S++%Flags=AS%Ops=MNNTNW)
T2(Resp=N)
T3(Resp=N)
T4(Resp=Y%DF=Y%W=0%ACK=O%Flags=R%Ops=)
T5(Resp=Y%DF=Y%W=0%ACK=S++%Flags=AR%Ops=)
T6(Resp=Y%DF=Y%W=0%ACK=O%Flags=R%Ops=)
T7(Resp=Y%DF=Y%W=0%ACK=S++%Flags=AR%Ops=)
PU(Resp=Y%DF=N%TOS=C0%IPLEN=164%RIPTL=148%RID=E%RIPCK=E%UCK
=E%ULEN=134%DAT=E)
```

This satisfied our 1st phase port scan and we achieved the results that we were after. As you can see in the results above, the only ports available for connection are port 22 for the ssh daemon and port 3306 for mysql. This can also be affirmed by performing the netstat command from the host itself to verify what it is listening on.

```
linux:/home/linuxuser # netstat -a |grep LISTEN
tcp        0      0 *:mysql                 *:*
LISTEN
```

```
tcp        0      0 *:32783                 *:*
LISTEN
tcp        0      0 *:ssh                   *:*
LISTEN
unix  2      [ ACC ]     STREAM     LISTENING     3383
/var/run/.nscd_socket
unix  2      [ ACC ]     STREAM     LISTENING     10893
/var/lib/mysql/mysql.sock
```

3.2 External Audit

After the initial nmap port scan, we now opt for something a little more intrusive. We will perform an external security scan using our existing nessus[4] auditing server. The full results of this scan are shown in Appendix B except for the initial vulnerabilities listing which was not included because it was quite lengthy. The nessus scan starts with an intrusive port scan, followed by a vulnerabilities scan which is smart in nature. The initial security scan yielded some interesting results which will now be analyzed. First, it detected ports 3306/tcp and 22/tcp for mysql and ssh respectively. Also, it detected port 32783/tcp to be open and gave no further information as to which service owns this port. Since nessus gave no information on that port I chose to dig a little deeper and check the Internet Storm Center from SANS to see if anyone else reported any vulnerabilities on this port. The check resulted in finding no vulnerabilities registered to this port [5] since I am not listening on this port and it did not turn up in any other scans I decided to move on with the audit and will probably research this further at a later date.

Since nessus detected that mysql was running on the server, it ran further penetration tests to see what known vulnerabilities and exploits were available against the versions I was running. It came back with 3 HIGH level warnings about the version of mysql that is currently installed. It recommends at least having version 3.23.56 installed to avoid being vulnerable. After investigating the warnings and checking with the vendor's website I was able to determine that the version that I was running was in fact patched in a sub-revision of the rpm package so that the result from nessus was a false positive.

Next on the list was another HIGH warning which was for the ssh version that I had installed. The explanation given by the nessus scan was that this version was exploitable through a flaw in buffer management which may allow for arbitrary code to be executed on the remote host. It turns out that this flaw was addressed in a recent sub-revision patch by SuSE as well so it too was a false positive. All other warnings from nessus were of the "Low" category so I read them and assessed that none of them required immediate attention save one. I

[4] http://nessus.org

[5] http://isc.incidents.org/port_details.html?port=32783&repax=1&tarax=2&srcax=2&percent=N&days=40

had not disabled ssh v1 in my hardening procedure. This is something that I missed in my initial assessment and was glad that nessus caught me on it. In order to change my ssh daemon to only accept connections using ssh2 I need to revisit the /etc/ssh/sshd_config file and change the following line:

`#Protocol 1,2` ← uncomment and should be changed to only show 2 as the value

(Just like the earlier change to ssh_config this will also require that the ssh daemon be restarted to take affect)

3.3 Impact

Now that the server has been hardened and tested we can revisit some of the general risk associated with this deployment. Given the verified level of hardening, we can comfortably move forward with the accepting of data into the database based on an accepted minimal level of risk. While no system can be 100% secure I feel that this process provides this organization a venue for moving forward with business over the public internet using a secured linux distribution. The overall impact that the process provided was that it addressed some serious security holes in an "out of the box' distribution and gave my organization a mitigated level of risk while maintaining the desired level of functionality as per the original goal. In simple technical terms, this means that clients can establish TCP sockets to this server on port 3306 and exchange data with this organization in a fashion that is as secure as is possible today.

In addition to meeting our immediate needs, we have also come up with a process that can be seamlessly applied to other core systems across the enterprise. This process was handed over to a team of engineers in Asia that followed the same procedures listed in this case study and their project yielded similar results. This added a second level of verification to my study and showed that the process was indeed streamlined and scalable.

Appendix A

```
linux:~ # InteractiveBastille
Using Tk user interface module.
Only displaying questions relevant to the current configuration.
Couldn't determine SuSE version! Setting to 7.2!
```

BASTILLE DISCLAIMER:

```
Use of Bastille can help efficiently optimize
system security, but does not guarantee system security.
Information about security obtained through use of
Bastille is provided on an AS-IS basis only
and is subject to change without notice.  Hewlett-Packard,
Jay Beale, and the Bastille developers DISCLAIM ALL
WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE.  Customer acknowledges that the customer
is responsible for their system's security.
You must accept the terms of this disclaimer to use
Bastille.  Type "accept" (without quotes) within 2
minutes to accept the terms of the above disclaimer
```

(Tk User Interface)

v1.2.0

Please answer all the questions to build a more secure system.

The Next and Back buttons navigate forward and backward in the questions database. Changes made in the Answer field are *only* saved when you push the Next button! The "modules" in the questions database are listed to the left. You can jump to the start of any module simply by clicking on its name.

Some questions have two levels of explanatory text, which you can adjust with the Explain Less/More button.

Please address bug reports and suggestions to jay@bastille-linux.org

Bugs in the Tk user interface are the fault of allenp@nwlink.com.

File Permissions:

Question:
Would you like to set more restrictive permissions on the administration utilities?

Explanation:
In general, the default file permissions set by most vendors are fairly secure. To make them more secure, though, you can remove non-root user access to some administrator functions.

If you choose this option, you'll be changing the permissions on some common system administration utilities so that they're not readable or executable by users other than root. These utilities (which include linuxconf, fsck, ifconfig, runlevel and portmap) are ones that most users should never have a need to access. This option will increase your system security, but there's a chance it will inconvenience your users.

Answer:
YES

Question:
Which permission profile should I use?

Explanation:
UnitedLinux supports different sets of file permissions depending on your security requirements. Different permission profiles are defined in /etc/permissions and related files; please select one of the following levels:

easy - this will leave all permissions as defined by the installed packages. This is the most convenient but also the most insecure configuration.

secure - this will take away the setuid bits from a large number of programs not needed for everyday operation by most operators.

paranoid - this will take away almost all setuid bits. Beware - the resulting system may not be fully usable. You can override some of the choices made by this setting by adding your preferred permissions to /etc/permissions.local.

Answer:
Secure

The following questions all pertain to disabling "SUID root" status for particular programs. This status allows non-root users to run these programs, increasing convenience but decreasing security. If a security weakness or vulnerability is found in these programs, it can be exploited to gain root-level access to your computer through any user account.

If you don't know what to choose for these questions, you might want to just answer "Yes" because you can always turn it back on later with chmod u+s <file name>.

Question:
Would you like to disable SUID status for mount/umount?

Explanation:
Mount and umount are used for mounting (activating) and unmounting (deactivating) drives that were not automatically mounted at boot time. This can include floppy and CD-ROM drives. Disabling SUID would still allow anyone with the root password to mount and unmount drives.

Answer:
YES

Question:
Would you like to disable SUID status for ping?

Explanation:
Ping is used for testing network connectivity--that is, for testing the ability of the network to get a packet from this machine to another and back. It should probably be used only by the person responsible for networking this host, who should have root access. Thus, we recommend disabling SUID status for it.

Answer:
YES

Question:
Would you like to disable SUID status for at?

Explanation:
"at" is used for scheduling an individual task to run at a single later time. There have historically been many exploits that take advantage of weaknesses in "at". Virtually all of the necessary functionality of "at " can be found in cron (and removing cron is not practical) so there is no need to retain privileged access for "at".

Answer:
YES

Question:
Would you like to disable SUID status for the r-tools?

Explanation:
The BSD r-tools rely on IP-based authentication, which means that you can allow anyone with (for instance) root access on 192.168.1.1 to have root access on 192.168.1.2. Administrators and other users have traditionally found this useful, as it lets them connect from one host to another without having to retype a password.

The problem with IP-based authentication, however, is that an intruder can craft "spoofed" or faked packets which claim to be from a trusted machine. Since the r-tools rely entirely on IP addresses for authentication, a spoofed packet will be accepted as real, and any hacker who claims to be from a trusted host will be trusted and given access to your machine.

These tools also transmit all of your data in cleartext, including passwords.

Tools are now available which allow you to spoof (fake) IP addresses as well as to monitor and/or hijack protocols which use cleartext. All of the same functionality can be found with the more secure replacement tools ssh and scp. Because of these insecurities, ordinary users should not be allowed to use the r-tools, and admins should use them only in cases where there are no other connection methods available.

Answer:
YES

Question:
Would you like to prohibit the clear-text r-protocols which trust IP addresses for authentication?

Explanation:
The BSD r-tools rely on IP-based authentication, which means that you can allow anyone with (for instance) root access on 192.168.1.1 have root access on 192.168.1.2. Administrators and other users have traditionally found this useful, as it lets them connect from one host to another without having to retype a password. The .rhosts file contains the names of the accounts and machines that are considered to be trusted.

The problem with IP-based authentication, however, is that an intruder can craft "spoofed" or faked packets which claim to be from a trusted machine. Since the r-tools rely entirely on IP addresses for authentication, a spoofed packet will be accepted as real.

Some of your users, or even possibly other administrators for this machine, might not be aware of the security problems with the BSD r-tools. If this is the case, they might create .rhosts files that would potentially allow crackers access to the machine. This option will disable the use of those r-tools using various methods.

Answer:
YES

Question:
Would you like to enforce password aging?

Explanation:
Your operating system's default behavior, which we would change here, is to disable an account when the password hasn't changed in 99,999 days. This interval is too long to be useful. We can set the default to 180 days. At some point before the 180 days have passed, the system will ask the user to change his or her password. At the end of the 180 days, if the password has not been changed, the account will be temporarily disabled. We would make this change in /etc/login.defs.

Answer:
YES

Question:

Would you like to enforce password aging?

Explanation:
Your operating system's default behavior, which we would change here, is to disable an account when the password hasn't changed in 99,999 days. This interval is too long to be useful. We can set the default to 180 days. At some point before the 180 days have passed, the system will ask the user to change his or her password. At the end of the 180 days, if the password has not been changed, the account will be temporarily disabled. We would make this change in /etc/login.defs.

Answer:
YES

Question:
Would you like to restrict the use of cron to administrative accounts?

Explanation:
Cron can be particularly useful for admins, giving them the ability to have the system check logs every night at midnight or confirm file integrity every hour. On the other hand, being able to execute jobs later or automatically represents an abusable privilege for users and also makes their actions slightly harder to track.

Many sites choose to restrict cron to administrative accounts. We suggest this action to new admins especially, until they understand more about how cron can be abused and know more about which users need access to cron. We would like to create the /etc/cron.allow file of users who may use cron. You can add to that later. If we don't create this file, all users will be allowed to use cron.

Answer:
YES

Question:
What umask would you like to set for users on the system?

Explanation:
The umask sets a default permission for files that you create. Bastille can set one of several umasks. Please select one of the following or create your own:

002 - Everyone can read your files & people in your group can alter them.

022 - Everyone can read your files, but no one can write to them.

027 - Only people in your group can read your files, no one can write to them.

077 - No one on the system can read or write your files.

Answer:
077

Question:
Should we disallow root login on tty's 1-6?

Explanation:
You can restrict which tty's root can login on. Some sites choose to restrict root logins, so that an admin must login with an ordinary user account and then use su to become root.

This can stop an attacker who has only been able to steal the root password from logging in directly. He has to steal a second account's password to make use of the root password via the ttys.

Answer:
YES

Question:
Would you like to password-protect the GRUB prompt?

Explanation:
If an attacker has physical access to this machine, and particularly to the keyboard, s/he could get superuser access through the Grand Unified Bootloader (GRUB) command line. We will look at other ways to prevent this later, but one easy way is to password-protect the GRUB prompt. If GRUB is password-protected, any user can reboot the machine normally, but only users with the password can pass arguments to the GRUB prompt.

Note that this option can interfere dual-booting with a second operating system, since dual booting often requires that type an O/S name to boot one of the two operating systems. If this machine sits in a general purpose lab and dual boots, you probably shouldn't choose this option.

Otherwise, this is strongly recommended for general use workstations and servers which are not locked away in their own room.

Answer:
NO

Question:
Would you like to disable CTRL-ALT-DELETE rebooting?

Explanation:

Disabling CTRL-ALT-DELETE rebooting is designed to prevent an attacker with access to the machine's keyboard from being able to reboot the machine. A reboot done in this manner should not damage the filesystem, as it shuts the machine down cleanly, writing out all pending data in the disk cache to disk first. Even with this functionality disabled, however, an attacker could just power cycle machine or pull the power cord.

Unless the power line, switch and case of the machine can be physically protected, this precaution is wholly unnecessary. Given the fact that

the attacker _can_ reboot the machine, would you prefer that s/he do it in a way potentially damages the filesystem? Think carefully here, as maintaining the integrity of the machine's filesystem may be secondary to the goal of keeping an attacker off, in which case it is better to answer yes here, since having to repair/ignore the damage and wait for filesystem checks may slow the attacker down.

Answer:
YES

Question:
Would you like to password protect single-user mode?

Explanation:
As we mentioned earlier, anyone who can get to the console on your machine can bring your machine up in "single user mode", where s/he is given root privileges and everyone else is locked out of the system. If you password protect single user mode, you won't have to remember yet another password--single user mode, or "root" mode, will require the root password.

We HIGHLY recommend that you password protect single user mode.

Answer:
YES

Question:
Would you like to set a default-deny on TCP Wrappers and xinetd?

Explanation:
Not recommended for most users:

Many network services can be configured to restrict access to certain network addresses (and in the case of 'xinetd' services in Linux-Mandrake 8.0 and Red Hat 7.x, other criteria as well). For services running under the older 'inetd' super-server (found in older versions of Linux-Mandrake and Red Hat, and current versions of some other distributions), some standalone services like OpenSSH, and --unless otherwise configured-- services running under Red Hat's xinetd super-server, you can configure restrictions based on network address in /etc/hosts.allow. The services using inetd or xinetd typically include telnet, ftp, pop, imap, finger, and a number of other services.

If you would like, Bastille can configure a default policy for all inetd, xinetd, and TCP Wrappers-aware services to deny all connection attempts. While you might have already chosen to install Bastille's firewall, setting a default deny policy for these services gives more defense in depth.

This will also configure xinetd so that the currently-installed xinetd services will use xinetd's more flexible access control and *not* /etc/hosts.allow. All other wrappers-based programs, like sshd, will obey the default-deny.

Answer:
YES

Question:
Should Bastille ensure the telnet service does not run on this system?

Explanation:
Telnet has severe design flaws.

telnet is a cleartext protocol, which means that when you type your password, it can be seen by a number of other machines on the network. Any machine on your LAN, along with the other machine's LAN, along with many computers/routers between the two LANS, can often eavesdrop on the telnet session and grab passwords. There's also another more active attack. Anyone who can eavesdrop can usually take over your telnet session, using a tool like Hunt or ettercap.

The standard practice among security-conscious sites is to migrate as rapidly as possible from telnet to Secure Shell (ssh). We'd advise you to make this move as soon as possible. ssh implementations are available from openssh.org and ssh.com. Some Operating System vendors also distribute a version of ssh.

Answer:
YES

Question:
Should Bastille ensure the FTP service does not run on this system?

Explanation:
Ftp is another problematic protocol. First, it is a cleartext protocol, like telnet -- this allows an attacker to eavesdrop on sessions and steal passwords. This also allows an attacker to take over an FTP session, using a cleartext-takeover tool like Hunt or Ettercap. Second, it can make effective firewalling difficult. Third, every major FTP daemon has had a long history of security vulnerability -- they represent one of the major successful attack vectors for remote root attacks.

FTP can often be replaced by Secure Shell's scp and sftp programs.

Answer:
YES

Question:
Would you like to display "Authorized Use" messages at log-in time?

Explanation:
At this point you can create "Authorized Use Only" messages for your site. These may be very helpful in prosecuting system crackers you may catch trying to break into your system. Bastille can make you default messages which you may then later edit. This is sort of like an "anti-welcome mat" for your computer.

Answer:
YES

Further Explanation to previous YES answer:
A default login/telnet/ftp "Authorized Use Only" banner will be created, and will be found in /etc/issue. You should modify this banner to apply more specifically to your organization (for instance, adding any site- specific information to the default warnings). If this is a corporate site, check with your corporate counsel to determine the most appropriate warning for the banner. These banners, according to CIAC's bulletin

(http://ciac.llnl.gov/ciac/bulletins/j-043.shtml)

may make it much easier to prosecute intruders. By including this default banner, neither the Bastille development team nor Hewlett-Packard Company take any responsibility for your ability to prosecute system crackers. Please, especially if you run a corporate site, review/replace this with more specific language.

Question:
Please type in the name of the company, person, or other organization who owns or is responsible for this machine.

Explanation:
Bastille will start to make the banner more specific by telling the user who is responsible for this machine. This will state explicitly from whom the user needs to obtain authorization to use this machine.

Answer:
YourCompanyName

Question:
Would you like to disable the gcc compiler?

Explanation:
The most common modus operandi for the bulk of the system crackers out there is to gain access to your system, often through a regular user account, and then use that access to compile exploits against your system or other systems. Disabling the gcc compiler on your system will slow these crackers down, and may even prevent some attacks entirely.

If this machine is a dedicated server/firewall, which does not have users who need to compile programs, this action is strongly recommended. Otherwise, you should very carefully consider whether you will be inconveniencing your users by disabling the compiler. If you do choose to disable it, we'll do so by only allowing root access to the compiler.

Answer:
NO (but, in most cases YES)

Question:

Would you like to put limits on system resource usage?

Explanation:
Denial of Service attacks are often very difficult to defend against, since they don't require access of any kind to the target machine. Since several major daemons, including the web, name, and FTP servers, may run as a particular user, you can limit the effectiveness of many Denial of Service attacks by modifying /etc/security/limits.conf. If you restrict the resources available in this manner, you can effectively cripple most Denial of Service attacks.

If you choose this option, you'll be setting the following initial limits on resource usage:

- Creation of core files is turned off. Core files (also known as core dumps) are created when an application crashes. They can be useful for diagnosing system problems, but they are very large files and can be exploited by an attacker to fill up your filesystem.

- Individual users are limited to 150 processes each. This should be more than enough for normal system usage, and is not enough to bring down your machine.

- Individual files are limited to a size of 100MB. Again, this should be more than enough for normal system usage.

All of these values can be edited later.

Answer:
NO

Question:
Should we restrict console access to a small group of user accounts?

Explanation:
Under some distributions, users logged in at the console have some special access rights (like the ability to mount the CD-ROM drive). You can disable this special access entirely, but a more flexible option is to restrict console access to a small group of trusted user accounts.

Answer:
YES

Question in Response to previous answer:
Which accounts should be able to login at console?

Explanation:
Please enter in the account names that should be able to login via the console, placing a space between each name.

Answer:
Root user1 user2 …

Question:
Would you like to add additional logging?

Explanation:
We would like to configure additional logging for your system. We will give you the option to log to a remote host, if your site already has one. We will add two additional logging files to the default setup and will also log some status messages to the 7th and 8th virtual terminals (the ones you'll see when you hit ALT-F7 and ALT-F8). This additional logging will not change the existing log files at all, so this is by no means a "risky" move.

Answer:
YES

Further explanation on previous YES response:
This script is adding additional logging files:

/var/log/kernel -- kernel messages

/var/log/syslog -- messages of severity "warning" and "error"

Also, if you check the 7th and 8th TTY's, by hitting ALT-F7 or ALT-F8, you'll find that we are now logging to virtual TTY's as well. If you try this, remember that you can use ALT-F1 to get back to the first virtual TTY.

Question:
Do you have a remote logging host?

Explanation:
If you already have a remote logging host, we can set this machine to log to it

Answer:
YES

Question:
What is the IP address of the machine you want to log to?

Explanation:
What is the IP address of the machine you normally log to? Remember, this should be a machine already configured to accept logging. If you have no such machine, select <Back> and change your answer.

Note: we ask for an IP address because this is safer -- it avoids DNS cache poisoning attacks on logging. You may use a hostname, but it should be added to your /etc/hosts file...

Answer:
127.0.0.1

Misc. Daemons Explanation:

To make the operating system more secure, we try to deactivate all system daemons, especially those running at a high/unlimited level of privilege. Each active system daemon serves as a potential point of break-in, which might allow an attacker illegitimate access to your system. An attacker can use these system daemons to gain access if they are later found to have a bug or security vulnerability.

We practice a minimalism principle here: minimize the number of privileged system daemons and you can decrease your chances of being a victim should one of the standard daemons be found later to have vulnerability. This section will require careful attention, but if you have doubts, you should be able to safely select the default value in most cases.

Question:
Would you like to deactivate the routing daemons?

Explanation:
Very few machines need to be running routing daemons. If your machine is only connected to the internet through one method, you can disable routing protocols. If this machine is at an ISP or major networking center, you can leave this on, but please prepare to configure your routing daemon.

Answer:
YES

Question:
Do you want to stop sendmail from running in daemon mode?

Explanation:
You do not need to have sendmail running in daemon mode to send and receive email, and unless you have a constant network connection, you probably cannot run sendmail in daemon mode. Daemon mode means that sendmail is constantly listening on a network connection waiting to receive mail.

If you disable daemon mode, Bastille will ask you if you would like to run sendmail every few minutes to process the queue of outgoing mail. Most programs which send mail will still do so immediately, and processing the queue will take care of transient errors.

If you receive all of your email via a POP/IMAP mailbox provided by your ISP, you may have no need of daemon-mode sendmail, unless you're running a special fetchmail-style POP/IMAP based retrieval program. For instance, you can turn daemon mode off if you read your mail via Netscape's common POP/IMAP read functionality. The only reason to run sendmail in daemon mode is if you are running a mail server.

Answer:
YES

Question:
Would you like to install TMPDIR/TMP scripts?

Explanation:
Many programs use the /tmp directory in ways that are dangerous on multi-user systems. Many of those programs will use an alternate directory if one is specified with the TMPDIR or TMP environment variables. We can install scripts that will be run when users log in that safely create suitable temporary directories and set the TMPDIR and TMP environment variables. This depends on your system supporting /etc/profile.d scripts.

Answer:
NO

Question:
Do you want to implement the choices now or continue making choices?

Explanation:
We will now implement the choices you have made here.

Answer NO if you want to go back and make changes!

Answer:
YES

Configuration file has been saved. What would you like to do now?

Appendix B

Preferences settings for this scan

email	root
max_checks	100
log_whole_attack	yes
cgi_path	/cgi-bin
optimize_test	yes
checks_read_timeout	15
delay_between_tests	1
test_file	/etc/passwd
port_range	0-65356
ping_hosts	yes
reverse_lookup	no
host_expansion	dns;ip
subnet_class	C
max_hosts	16
scan_level	normal
outside_firewall	no
language	english
track_iothreads	yes
cookie_logpipe_suptmo	2
ntp_save_sessions	yes
ntp_detached_sessions	yes
server_info_nessusd_version	2.0.7
server_info_libnasl_version	2.0.7
server_info_libnessus_version	2.0.9
server_info_thread_manager	fork
server_info_os	Linux
server_info_os_version	2.4.21-4-686
auto_enable_dependencies	yes
safe_checks	no
ntp_keep_communication_alive	yes
ntp_opt_show_end	yes

save_session	yes
save_knowledge_base	no
detached_scan	no
continuous_scan	no

Summary of scanned hosts

Host	Holes	Warnings	Open ports	State
10.1.1.30	4	10	3	Finished

10.1.1.30

Service	Severity	Description
mysql (3306/tcp)	**Info**	Port is open
unknown (32783/tcp)	**Info**	Port is open
ssh (22/tcp)	**Info**	Port is open
mysql (3306/tcp)	**High**	You are running a version of MySQL which is older than version 3.23.56. It is vulnerable to a vulnerability that may allow the mysqld service to start with elevated privileges. An attacker can exploit this vulnerability by creating a DATADIR/my.cnf that includes the line 'user=root' under the '[mysqld]' option section. When the mysqld service is executed, it will run as the root user instead of the default user. Risk factor : High Solution : Upgrade to at least version 3.23.56 CVE : CAN-2003-0150 BID : 7052
mysql (3306/tcp)	**High**	You are running a version of MySQL which is older than version 3.23.54 or 4.0.6 If you have not patched this version, then

		any attacker may crash this service remotely. See also : http://security.e-matters.de/advisories/042002.html Solution : Upgrade to the latest version of MySQL Risk factor : Medium CVE : CAN-2002-1373, CAN-2002-1374, CAN-2002-1375, CAN-2002-1376 BID : 6368, 6370, 6373, 6374, 6375, 8796
ssh (22/tcp)	**High**	You are running a version of OpenSSH which is older than 3.7.1 Versions older than 3.7.1 are vulnerable to a flaw in the buffer management functions which might allow an attacker to execute arbitrary commands on this host. An exploit for this issue is rumored to exist. Note that several distribution patched this hole without changing the version number of OpenSSH. Since Nessus solely relied on the banner of the remote SSH server to perform this check, this might be a false positive. If you are running a RedHat host, make sure that the command : rpm -q openssh-server Returns : openssh-server-3.1p1-13 (RedHat 7.x) openssh-server-3.4p1-7 (RedHat 8.0) openssh-server-3.5p1-11 (RedHat 9) Solution : Upgrade to OpenSSH 3.7.1 See also : http://marc.theaimsgroup.com/?l=openbsd-misc&m=106375452423794&w=2 http://marc.theaimsgroup.com/?l=openbsd-misc&m=106375456923804&w=2 Risk factor : High

		CVE : CAN-2003-0682, CAN-2003-0693, CAN-2003-0695 BID : 8628
mysql (3306/tcp)	**High**	You are running a version of MySQL which is older than version 4.0.15. If you have not patched this version, then any attacker who has the credentials to connect to this server may execute arbitrary code on this host with the privileges of the mysql database by changing his password with a too long one containing a shell code. Solution : Upgrade to MySQL 3.0.58 or 4.0.15 Risk factor : Medium CVE : CAN-2003-0780 BID : 8590
ssh (22/tcp)	**Low**	An ssh server is running on this port
ssh (22/tcp)	**Low**	Remote SSH version : SSH-1.99-OpenSSH_3.4p1
ssh (22/tcp)	**Low**	The remote SSH daemon supports the following versions of the SSH protocol : . 1.33 . 1.5 . 1.99 . 2.0
mysql (3306/tcp)	**Low**	Remote MySQL version : 3.23.52-log
mysql (3306/tcp)	**Low**	An unknown service is running on this port. It is usually reserved for MySQL
ssh (22/tcp)	**Low**	You are running OpenSSH-portable 3.6.1p1 or older. If PAM support is enabled, an attacker may use a flaw in this version to determine the existence or a given login name by comparing the times the remote sshd daemon takes to refuse a bad password for a non-existent login compared to the time it takes to refuse a bad password for a valid login.

		An attacker may use this flaw to set up a brute force attack against the remote host. *** Nessus did not check whether the remote SSH daemon is actually *** using PAM or not, so this might be a false positive Solution : Upgrade to OpenSSH-portable 3.6.1p2 or newer Risk Factor : Low CVE : CAN-2003-0190 BID : 7482, 7467, 7342
general/udp	**Low**	For your information, here is the traceroute to 10.1.1.30 : 10.1.1.22 ? 10.1.1.30
ssh (22/tcp)	**Low**	The remote SSH daemon supports connections made using the version 1.33 and/or 1.5 of the SSH protocol. These protocols are not completely cryptographically safe so they should not be used. Solution : If you use OpenSSH, set the option 'Protocol' to '2' If you use SSH.com's set the option 'Ssh1Compatibility' to 'no' Risk factor : Low
ssh (22/tcp)	**Low**	You are running OpenSSH-portable 3.6.1 or older. There is a flaw in this version which may allow an attacker to bypass the access controls set by the administrator of this server. OpenSSH features a mechanism which can restrict the list of hosts a given user can log from by specifying a pattern in the user key file (ie: *.mynetwork.com would let a user connect only from the local network). However there is a flaw in the way OpenSSH does reverse DNS lookups. If an attacker configures his DNS server to send a numeric IP

		address when a reverse lookup is performed, he may be able to circumvent this mechanism. Solution : Upgrade to OpenSSH 3.6.2 when it comes out Risk Factor : Low CVE : CAN-2003-0386 BID : 7831
general/tcp	**Low**	Remote OS guess : Linux Kernel 2.4.20 CVE : CAN-1999-0454

Appendix C

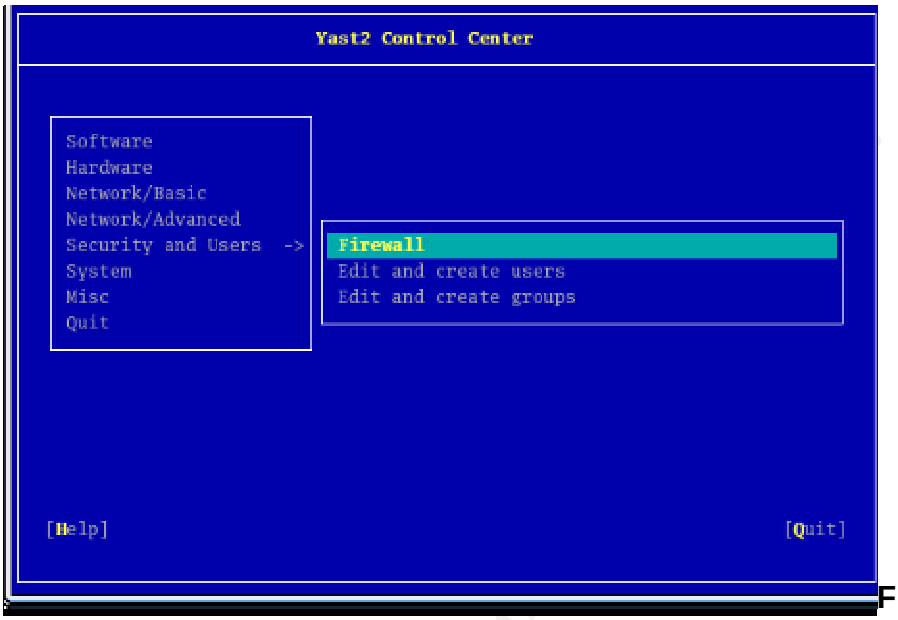


Figure1

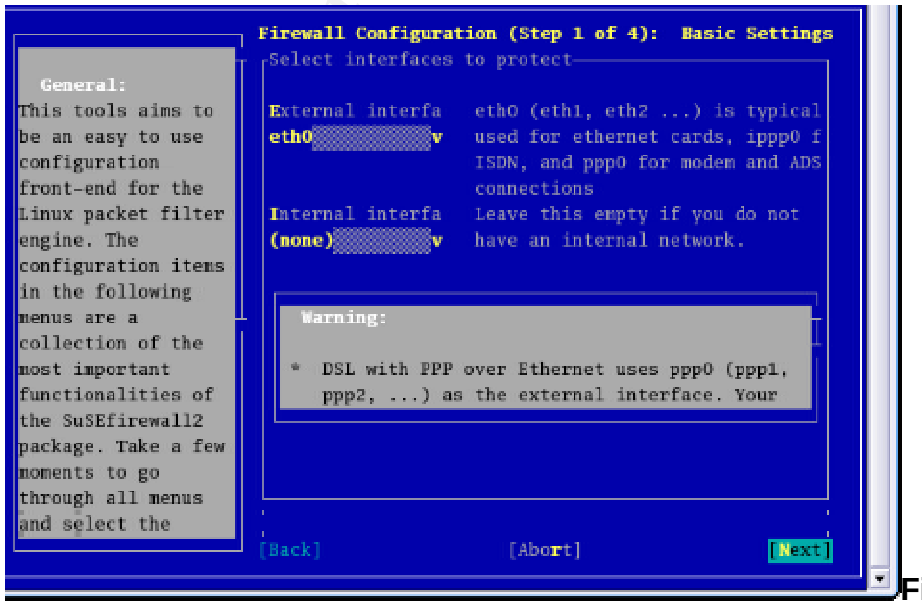


Figure2

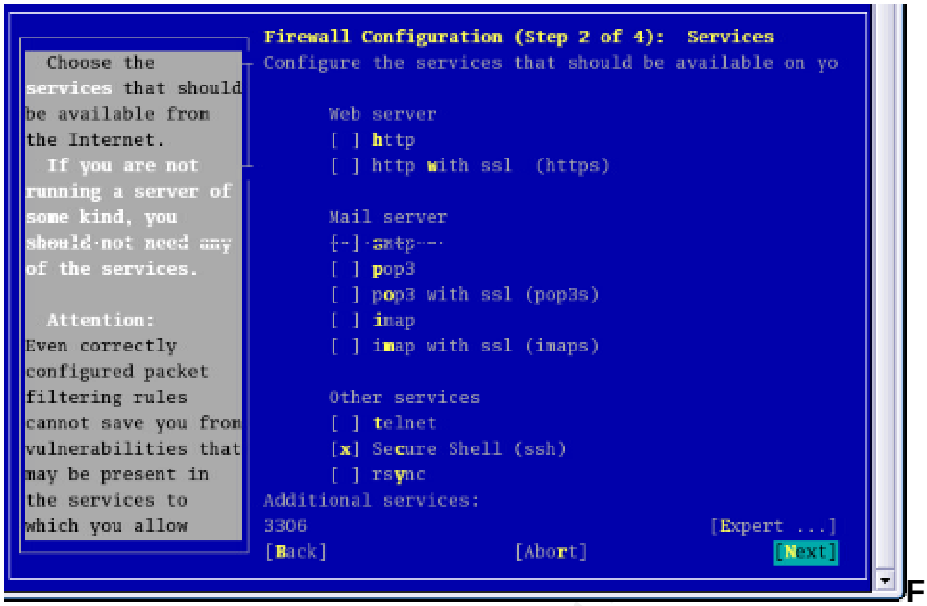


Figure3

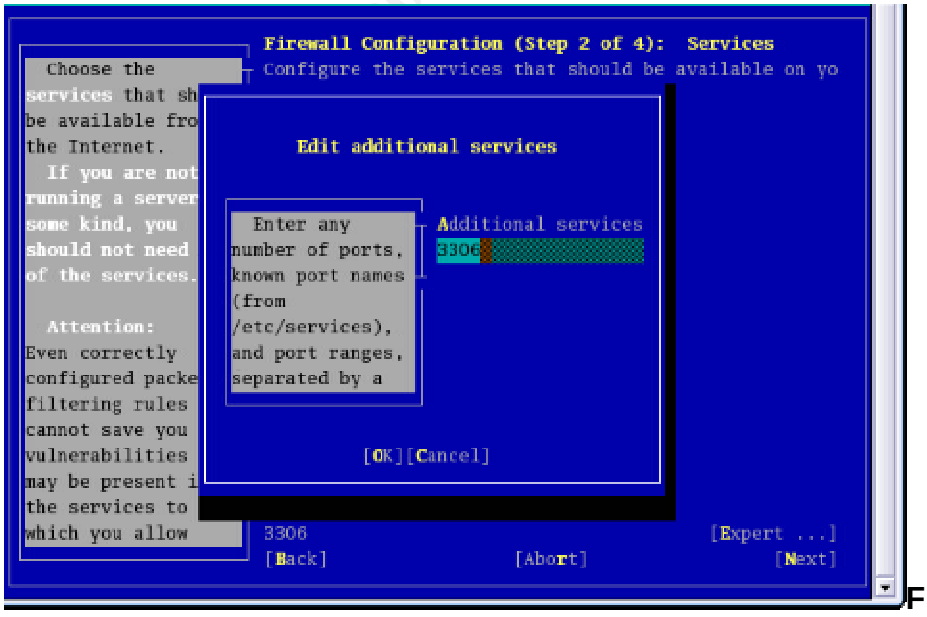


Figure4

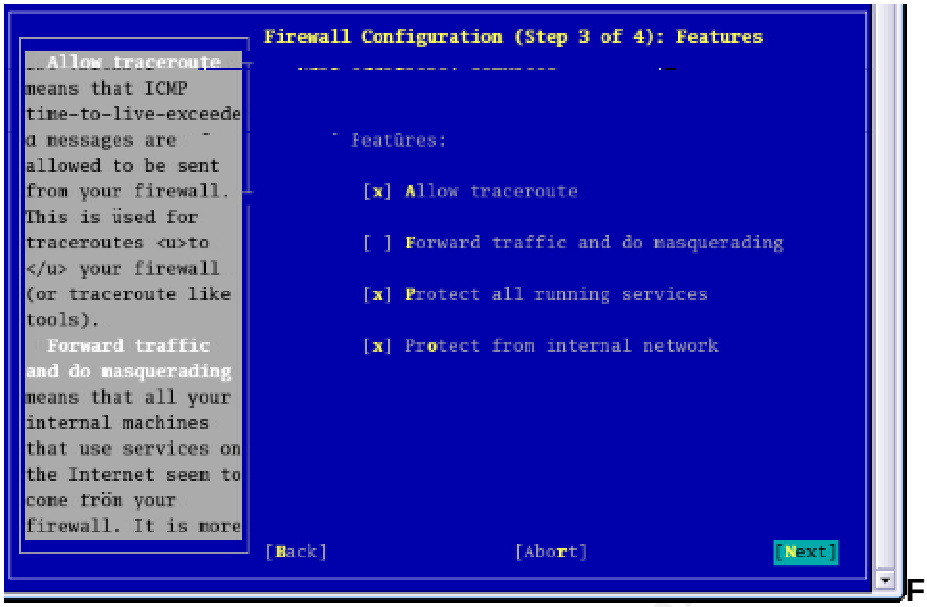


Figure5

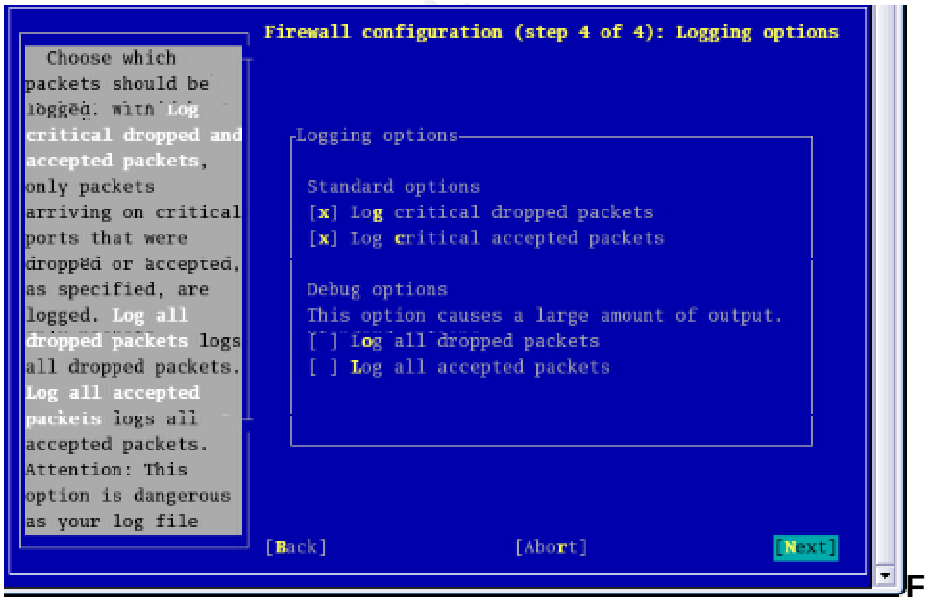


Figure6

References:

Nemeth, Evi , Snyder, Garth, et al. UNIX System Administration Handbook. Prentice Hall, 2001.

Anderson, Harry. “An Introduction to Nessus.”
URL: http://www.securityfocus.com/infocus/1741 (28 Oct 2003)

Fydoor. “Nmap man page.”
URL : http://www.insecure.org/nmap/data/nmap_manpage.html

Bastille Linux. “What people are asking…”
URL: http://www.bastille-linux.org/faq4.html

Weidner, Klaus. “SLES Security Guide.”
URL: http://ltp.sourceforge.net/docs/SLES-security-guide-EAL3.pdf (4 Dec 2003)

SuSE Linux. “Security Certification”
URL: http://www.suse.de/de/security/certification/index.html (21 Jan 2004)

“Internet Storm Center: Port Reports”. SANS Institute
http://isc.incidents.org/port_details.html?port=32783&repax=1&tarax=2&srcax=2&percent=N&days=40

Lasser, John, Beale, Jay et al. “Bastille Linux Interactive Mode” (15 Jun 2001)