Global Information Assurance Certification Paper

Network Intrusion In Depth

GCIA Practical Version 4.0

Michael Spellane

November 20, 2004

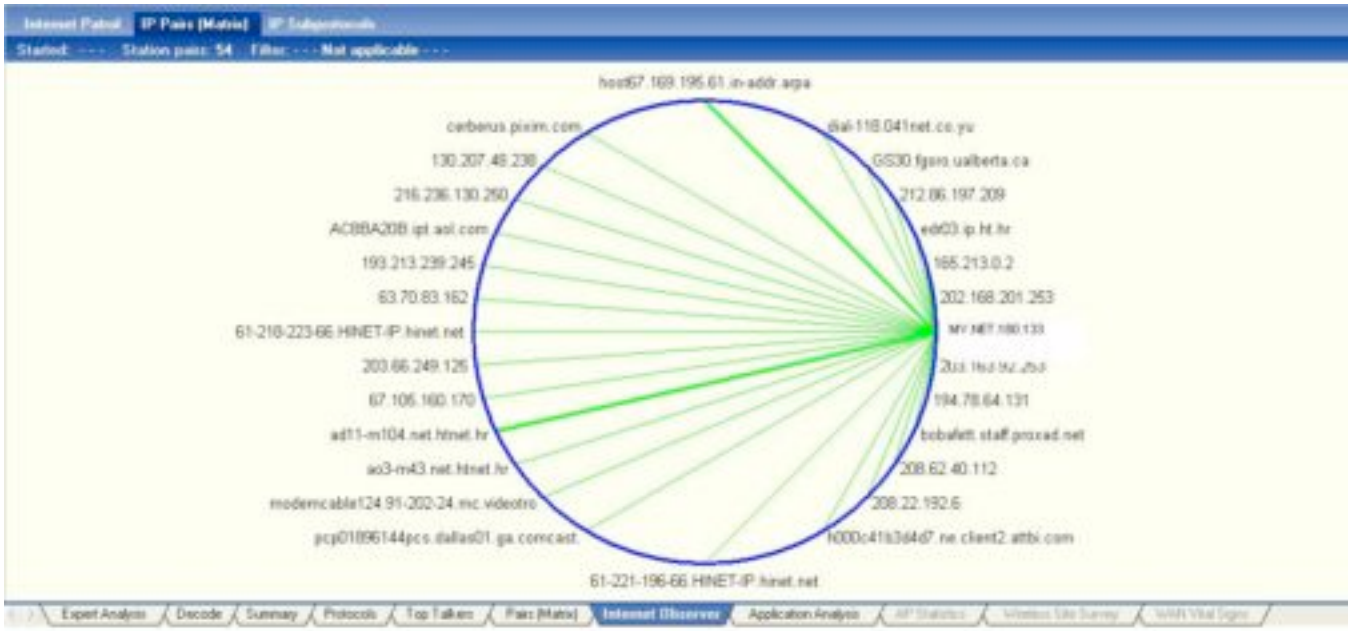
Abstract:
With this practical, I intend to demonstrate my effectives in the Intrusion Detection field and what I have learned and continue to learn in this vast field. I will analyze events that I have found of interest, and provide technical explanations about what is found.

Executive Summary:

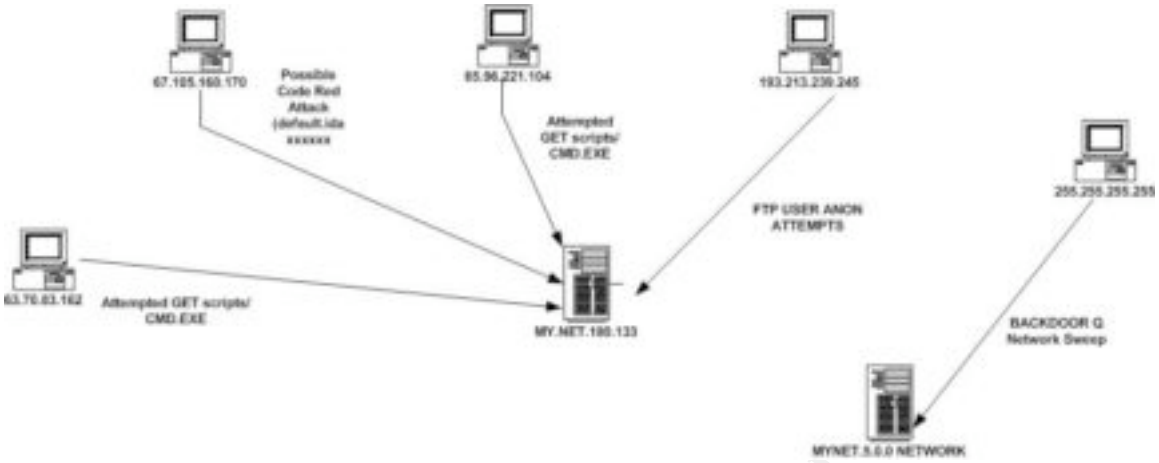
I have been tasked with analyzing the security posture of your University. The report that I have submitted contains detailed analysis pertaining to log files obtained from your Intrusion Detection System, and will point out alert's of concern as well as statistical data about the amount of traffic your network is generating. It is my intention to provide you with a clear view of the areas that require improvement, as well as professional recommendations. I have chosen in this report to focus in on, but not limited to, data that occurred on 06/09/2001. There is evidence of Peer to Peer network traffic, file sharing if you will, from many machines on your network. Universities typically have a plethora of bandwidth, as such, this becomes a perfect breeding ground for file sharing activity, as well as malicious activity. Upon reading this report, you will be asked what acceptable risk to your organization is. This will provide a clear path as to where your organization will go with network security, and the changes that you choose to implement recommended within this report.

Network relationships:

In looking at the data and discovering hosts' based on the protocols in use, here is a graph of what I believe to be a web/ftp server on the University network, and all hosts that have made attempted connections:



Link Graph:
Here is a link graph displaying a many to one relationship between external hosts and what is believed to be an internal web server. This graph also depicts a one to many network sweep from 255.255.255.255 (broadcast address of the class C network space). The GET default.ida xxxxxxx and the GET scripts/CMD.EXE are of interest here. They are typical of CODERED and NIMDA. These worms exploit vulnerable IIS web servers with Buffer Overflow attacks to achieve directory traversal. In looking at some packets where MY.NET.180.133 sent 403's (Forbidden) back to the requesting hosts, I do not believe this server is running IIS, and here's why: In looking at the 8th byte offset from 0 in the IP header, MY.NET.180.133 generated a TTL value of 61. Now, Microsoft had a couple of default TTL values, 32 for Win95, and WinNT3.51, and 128 for WinNT 4.0 on up to the current WinXP. So, unless the INTERNAL server is roughly 67 hops away from the SNORT sensor, I would say it is unlikely that this is a Microsoft IIS server. I would venture to say that this is more likely to be a Linux box running Apache.



Overview of Alert's:

The following is an overview of the alerts from file 2002.5.10 obtained from http://www.isc.sans.org/logs/raw and their aggregated counts. I will use this as the basis for Detects 1 and 2:

Signature	# Alerts	# Sources	# Destinations
BARE BYTE UNICODE ENCODING	416	1	10
DNS named version attempt	42	9	42
BACKDOOR Q access	36	1	36
WEBROOT DIRECTORY TRAVERSAL	24	4	8
OVERSIZE REQUEST-URI DIRECTORY	18	1	6
DOUBLE DECODING ATTACK	11	1	5
APACHE WHITESPACE (TAB)	10	1	5
NON-RFC HTTP DELIMITER	10	4	3
SHELLCODE x86 NOOP	5	1	1
BAD-TRAFFIC ip reserved bit set	1	1	1
MISC Tiny Fragments	1	1	1

Detect 1:

In order to understand what is going on in the capture file, we need to attempt to map out the network. Michael Meacle has a process in his practical that is very ingenious. He obtained the idea from Chris Reining. Due to page constraints implemented version 4.0 of the GCIA practical, I will not go into detail of every command typed to achieve the network layout, but simply provide a link to Michael Meacle's paper that has the instruction's on how to conclude the network layout.
http://www.giac.org/practical/GCIA/Michael_Meacle_GCIA.pdf

Below is what I believe the basic network layout of the file 2002.5.10 obtained from http://www.isc.sans.org/logs/raw to be:

INTERNET
|
CISCO DEVICE (0:3:e3:d9:26:c0)
|
SNORT SENSOR
|
CISCO DEVICE (0:0:c:4:b2:33)
|
NATTED DEVICE (REST OF NETWORK)

[**] [1:184:6] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/09-20:18:48.944488 255.255.255.255:31337 -> MY.NET.87.61:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS203]

[**] [1:184:6] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/09-20:20:27.934488 255.255.255.255:31337 -> MY.NET.172.250:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS203]

[**] [1:184:6] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/09-20:28:33.904488 255.255.255.255:31337 -> MY.NET.9.165:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS203]

Attack Description:

BACKDOOR Q is a Trojan backdoor that affects UNIX operating systems. The backdoor could allow an attacker to run commands remotely as root.
The Trojan is controlled by sending raw packets to the target computer. Once the backdoor has been delivered and installed, the attacker can send commands via TCP, UDP, or ICMP from the broadcast address of a class “C” network (255.255.255.255). This would allow an attacker to craft packets and evade detection of their true source. The snort rule description gave me better insight on this alert.
http://www.snort.org/snort-db/sid.html?sid=184

This attack was selected due to the use of port's 31337 and 515. 31337 being the port that Backorifice (http://www.symantec.com/avcenter/warn/backorifice.html) uses and 515 being the UNIX printer daemon. Older versions of the UNIX printer daemon have a vulnerability. Exploits have been released into the wild; such exploits include the lpdw0rm and the Ramen worm.
Ref: http://www.securityfocus.com/archive/75/180015 lpdw0rm
Ref: http://www.whitehats.com/library/worms/ramen Ramen

Detect was generated by:

Snort version 2.2 using default ruleset. Snort was ran against the raw logfile 2002.5.10 obtained from isc.sans.org/logs/raw.
The following rule triggered this event:
1:184 indicated by the [1:184:6] in the alert.
The rule that triggers this event contains the following:
alert tcp 255.255.255.0/24 any -> $HOME_NET any (msg:"BACKDOOR Q access"; dsize:>1; flags:A+; flow:stateless; reference:arachnids,203; classtype:misc-activity; sid:184; rev:6;)

Probability the source address was spoofed:

High – The IP is 255.255.255.255. This IP is the broadcast address of the class "C" network space. In reading another student's practical, he states that, per RFC950, this is normal traffic used by a host to discover it's subnet mask. However, in reading RFC950, it states that the protocol used to accomplish this is ICMP, whereas the traffic in question is TCP. Also, there are no signs of a completed 3 way handshake, the TCP packet is being sent with the ACK RST Flags set, no initial SYN. All of this on top of the fact that this, again, is TCP traffic coming from a broadcast address, and as we know, TCP only supports unicast.

Attack Mechanism:

The attacker is looking for UNIX boxes that have the BACKDOOR Q server installed. The packets are being directed to multiple IP's on the same network targeting port 515 (UNIX Printer Port). The payload of the packet contains "CKO", which, Perhaps, the backdoor that was installed responds to CKO data, and this could be a "wakeup call" instructing the Trojan to phone home, whether "home" be the attacker's computer, or an IRC server. This is obviously a crafted packet, either by hand or by other means, such as an automated tool. The IP of 255.255.255.255 is not the only key here. The IPID is 0 on all packets that are sent. This should not be. Additionally, the IPID should increment for each packet sent. The attacker is also using the source port of 31337, very odd, this is the port that BackOrifice utilizes for it's backdoor. The odd thing about this traffic (and is not reflected in the above raw data) is the DNS BIND requests within

minutes of the BACKDOORQ alert. This is ALMOST consistent throughout the beginning of the dataset, but not entirely. Could the BACKDOOR Q attack just be a decoy to throw an analyst off the real attack? Supporting evidence could be the fact that the attacker is using the source port of 31337, destination port of 515,and a TTL of 14 (ok, that one might be a stretch) this (to me, at least) sticks out like a sore thumb. This was my thinking at first blush, however, I am leaning towards maybe not since the two alerts are not consistent enough throughout the ENTIRE dataset.

Correlations:

There are many correlations to this event. It actually seems like this subject has been debated for some time.

Maslowski-Yerges – GCIA Practical
http://www.giac.org/practical/GCIA/Al_Maslowski-Yerges_GCIA.pdf

Whitehats.org
http://www.whitehats.com/info/IDS201

Evidence of Active targeting:

No. Throughout the entire dataset, the attacker seems to be sweeping the MYNET.5.x.x network space in hopes to kickoff a remote program.

Severity:

Criticality = 2

This appears to be a network sweep, the attacker is not targeting a specific host, so the OS is unknown. I did not see any traffic in response to this either.

Lethality = 5

If an attacker has the ability to send data to the backdoor, in order to run commands as root, this would be a very, very bad thing.

System Countermeasures = 4

The system's that are being targeted do not seem to be responding to this attack. Either it is not if the preferred OS flavor for the attack, or it has a defensive mechanism in place.

Network Countermeasures = 2

This is an unknown network, I have no idea which (if any) network defensive mechanisms are in place.

(2+5) – (4+2) = 1

Detect 2:
Again, since this detect came from the same file as above, this is what I believe the basic network layout of the file 2002.5.10 obtained from http://www.isc.sans.org/logs/raw to be:

INTERNET
|
CISCO DEVICE (0:3:e3:d9:26:c0)
|
SNORT SENSOR
|
CISCO DEVICE (0:0:c:4:b2:33)
|
NATTED DEVICE (REST OF NETWORK)

[**] [1:523:5] BAD-TRAFFIC ip reserved bit set [**]
[Classification: Misc activity] [Priority: 3]
06/10-09:41:19.544488 218.2.129.171 -> MY.NET.188.185
TCP TTL:231 TOS:0x0 ID:0 IpLen:20 DgmLen:40 RB
Frag Offset: 0x11F1 Frag Size: 0x0014

[**] [1:522:2] MISC Tiny Fragments [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
06/10-14:51:14.694488 218.2.129.171 -> MY.NET.142.232
TCP TTL:231 TOS:0x0 ID:0 IpLen:20 DgmLen:40 MF
Frag Offset: 0x0004 Frag Size: 0x0014

I know that these are two different alerts, but I believe that they coinincide with one another.

Attack description:

When IP fragmentation occurs, the packets are broken up into smaller fragments and reassembled at the destination. If the packets are sent out of order, the IDS must properly reassemble the packet. An IDS that does not reassemble the packet correctly is susceptible to evasion.

This alert was selected because I believe this could be some type of IDS evasion technique. The fact that we only see each alert once and that they are split up by roughly 5 hours, leads me to believe that this may be a “low and slow” attacker.

Detect was generated by:

Snort version 2.2 using default rule set. Snort was ran against the raw logfile 2002.5.10 obtained from isc.sans.org/logs/raw.
The following rule triggered this event:

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"BAD-TRAFFIC ip reserved bit set"; fragbits:R; classtype:misc-activity; sid:523; rev:5;)

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"MISC Tiny Fragments"; dsize:< 25; fragbits:M; classtype:bad-unknown; sid:522; rev:2;)

Probability the source address was spoofed:

HIGH – Given that the 9th byte offset from 0 in the IP header is 06 (TCP), I see no evidence of a 3-way handshake. It looks like the attacker has crafted this packet by hand, given that the ID is set to 0, and in the first packet, the reserved bit is set (not normal TCP traffic) and the second packet has the MF flag set, with only 40 bytes as a total length in the packet.

Attack Mechanism:

An attacker can send a fragmented packet with a proper TCP header and the firewall will allow this to pass. The second portion of the fragmented traffic will contain information that will overwrite the original TCP header that is now on the other side of the firewall if it is vulnerable. This is known as fragmentation overlap and fragmentation overwrite. The key difference between the two is that fragmentation overlap only overwrites the last byte in the first packet. Fragmentation overwrite overwrites the entire previous fragment. There is an excellent write up from Kevin Timm on **http://www.securityfocus.com/infocus/1577** that explains various methods of IDS evasion. Here is a line from that write up which cites an example of overlapping;

“Fragmentation overlap involves sending packets so one fragment overwrites data from a previous fragment therefore evading network detection. An example could be as follows:

Packet #1 GET x.idd
Packet #2 a.?(buffer overflow here)

The packets are assembled so that packet #2 overwrites the last byte of packet #1 so that when they are re-assembled on the destination host the string is GET x.ida?(buffer overflow). "

Another interesting fact I took away from this paper is the technique of fragmentation timeout's. The attacker will send a packet with the More Fragments bit set, and then wait for the IDS to discard the packet. Older or improperly configured IDS will discard the packet after 60 seconds.
The attacker may also fragment an exploit that will elude IPS's.
http://www.snort.org/snort-db/sid.html?sid=522

This alert is interesting in the fact that NORMAL traffic will not have the reserved bit set. When you look at the IPID of 0, the TTL of 231 on top of the fact that the reserved bit is set, you can pretty easily come to the conclusion that this packet is crafted.

"MISC Tiny Fragments" This alert is also interesting. The fact that the TTL is again 231 and the IPID is again 0 leads me to believe that this packet is crafted as well. Another interesting fact about this packet is the fact that the More Fragments flag is set, yet, the total Datagram length is only 40 bytes. According to the README file on the http://isc.sans.org/logs/raw URL, all of the "normal" traffic has been extracted. In going with that, there could be more packets that are related to this traffic that we are not seeing here because SNORT did not pick them up, or they were just pulled out of the dataset all together. I have used nemesis to emulate this traffic on my home network to see what it would do:

Nemesis ip –S MY.NET.SPOOFED.SOURCE –D MY.NET.HOME.NET –I 0 –p 6 –F R 32

My SNORT box picked up this packet just like it was supposed to. I did not evade my IDS, nor did I crash the machine.

If you look at the 9th byte offset of the IP header, you will see a 06, indicating that the protocol is TCP; however, I do not see the TCP header in the packet. I believe that the second fragment is sent with no TCP header. I understand fragmentation; I now want to see it in action. To test this, I used the standard ping command to send a packet with 1500 bytes of data:

Ping –l 1500 MY.NET.VICTIM.MACHINE
The l -1500 tells ping to send 1500 bytes of data. Here are the results:

First part of fragmented packet:

16:56:59.123505 IP (tos 0x0, ttl 128, len 1500) MY.NET.SPOOFED.SOURCE > MY.NET.VICTIM.MACHINE: icmp 1480: echo request seq 23552 (wrong icmp cksum 675 (->e247)!) (frag 28745:1480@0+)

```
0x0000    4500 05dc 7049 2000 8001 236e c0a8 000b    E...pI....#n....
0x0010    c0a8 000e 0800 0675 0200 5c00 6162 6364    .......u..\.abcd
0x0020    6566 6768 696a 6b6c 6d6e 6f70 7172 7374    efghijklmnopqrst
0x0030    7576 7761 6263 6465 6667 6869 6a6b 6c6d    uvwabcdefghijklm
0x0040    6e6f 7071 7273 7475 7677 6162 6364 6566    nopqrstuvwabcdef
0x0050    6768 696a 6b6c 6d6e 6f70 7172 7374 7576    ghijklmnopqrstuv
0x0060    7761 6263 6465 6667 6869 6a6b 6c6d 6e6f    wabcdefghijklmno
0x0070    7071 7273 7475 7677 6162 6364 6566 6768    pqrstuvwabcdefgh
0x0080    696a 6b6c 6d6e 6f70 7172 7374 7576 7761    ijklmnopqrstuvwa
0x0090    6263 6465 6667 6869 6a6b 6c6d 6e6f 7071    bcdefghijklmnopq
0x00a0    7273 7475 7677 6162 6364 6566 6768 696a    rstuvwabcdefghij
0x00b0    6b6c 6d6e 6f70 7172 7374 7576 7761 6263    klmnopqrstuvwabc
0x00c0    6465 6667 6869 6a6b 6c6d 6e6f 7071 7273    defghijklmnopqrs
0x00d0    7475 7677 6162 6364 6566 6768 696a 6b6c    tuvwabcdefghijkl
0x00e0    6d6e 6f70 7172 7374 7576 7761 6263 6465    mnopqrstuvwabcde
0x00f0    6667 6869 6a6b 6c6d 6e6f 7071 7273 7475    fghijklmnopqrstu
```

Second part of fragmented packet:

```
16:56:59.128483 IP (tos 0x0, ttl 128, len 48) MY.NET.HOME.11 >
MY.NET.HOME.14: icmp (frag 28745:28@1480)
0x0000    4500 0030 7049 00b9 8001 4861 c0a8 000b    E..0pI....Ha....
0x0010    c0a8 000e 6162 6364 6566 6768 696a 6b6c    ....abcdefghijkl
0x0020    6d6e 6f70 7172 7374 7576 7761 6263 6465    mnopqrstuvwabcde
```

As you can see, the packet first packet is sent with 1500 bytes total length (ethereal will show this as ICMP protocol) and the second part of the packet is sent with 48 bytes total length (ethereal sees this as IP protocol) with no ICMP header information. If you look at the 6th byte offset in the IP header, you will see this field zeroed out; the More Fragments flag is not set. This is normal fragmentation at work.

CORRELATIONS:
http://www.securityfocus.com/infocus/1577
http://www.faqs.org/rfcs/rfc1858.html

Evidence of active targeting: NONE – The two packets are being sent to two different machines. I believe the attacker is trying to see if he can evade the IDS, it does not appear that he has a specific target in mind.
SEVERITY:

Criticality: 2 Not knowing the utility of these machines, it is hard to gauge. However, through the entire dataset, this destination IP only shows up twice. If it were a resource that serviced many user's and computers (ie. Web server, DNS server ect.) then we would probably see much more traffic to and from these machines.

Lethality: 5 If an attack like this were to succeed, an attacker would have successfully evaded an IDS. To me, this is a VERY bad thing.

System Countermeasure: 5 This, again, is hard to determine without knowing the system itself. As I stated above, I see no other traffic going to or coming from these machines, which leads me to believe that the machine is not servicing any specific requests (no IIS, no FTP, no SMTP ect.).

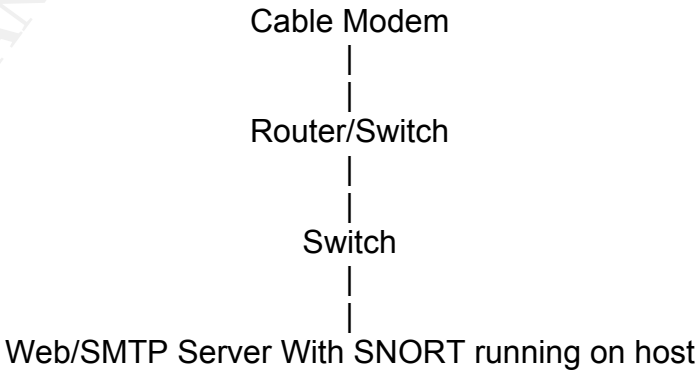
Network Countermeasure: 3 While snort picked up these packets as an alert, should the packet have made it that far? I think not, if the external firewall was capable of dropping packets with the IP reserved bit set, and the MF flag set on a fragment that is small in nature.

Detect 3:

The following is an aggregated view of the alert's from a single day on my home network:

Signature	# Alerts	# Sources	# Destinations
DOUBLE DECODING ATTACK	16	1	1
OVERSIZE REQUEST-URI DIRECTORY	14	7	1
BARE BYTE UNICODE ENCODING	14	7	1
WEBROOT DIRECTORY TRAVERSAL	4	1	1
ICMP PING	4	1	1
ICMP L3retriever Ping	4	1	1
NON-RFC HTTP DELIMITER	2	1	1

This detect was taken from my home network. Below is the layout of the network:



CodeRed II still making lots of noise.

It is still amazing to me that in the day's of technological choices that consumers have, that a virus, 3 years old now, is still running wild. How can it be that with all of the Antivirus programs, personal hardware firewall's, software firewalls, and widely available OS and Application patches, that a Virus is still able to go without the user knowing about it?

[**] [119:3:1] (http_inspect) U ENCODING [**]
08/09-19:27:53.556903 68.83.35.132:4011 -> MY.NET.HOME.5:80
TCP TTL:118 TOS:0x0 ID:35728 IpLen:20 DgmLen:1500 DF
A* Seq: 0x6BA48E87 Ack: 0xFFAFEFD9 Win: 0x4470 TcpLen: 20

[**] [119:13:1] (http_inspect) NON-RFC HTTP DELIMITER [**]
08/09-19:27:53.556903 68.83.35.132:4011 -> MY.NET.HOME.5:80
TCP TTL:118 TOS:0x0 ID:35728 IpLen:20 DgmLen:1500 DF
A* Seq: 0x6BA48E87 Ack: 0xFFAFEFD9 Win: 0x4470 TcpLen: 20

[**] [119:4:1] (http_inspect) BARE BYTE UNICODE ENCODING [**]
08/09-19:27:53.598209 68.83.35.132:4011 -> MY.NET.HOME.5:80
TCP TTL:118 TOS:0x0 ID:35729 IpLen:20 DgmLen:1500 DF
A* Seq: 0x6BA4943B Ack: 0xFFAFEFD9 Win: 0x4470 TcpLen: 20

[**] [119:15:1] (http_inspect) OVERSIZE REQUEST-URI DIRECTORY [**]
08/09-19:27:53.650470 68.83.35.132:4011 -> MY.NET.HOME.5:80
TCP TTL:118 TOS:0x0 ID:35738 IpLen:20 DgmLen:938 DF
AP Seq: 0x6BA499EF Ack: 0xFFAFEFD9 Win: 0x4470 TcpLen: 20

Attack Description:

"CodeRed II was discovered on August 4, 2001. It has been called a variant of the original CodeRed Worm because it uses the same "buffer overflow" exploit to propagate to other Web servers. Symantec Security Response received reports of a high number of infected IIS Web servers. CodeRed II is considered to be a serious threat." – Symantec website.
http://securityresponse.symantec.com/avcenter/venc/data/codered.ii.html

Codered II was designed to exploit a known Buffer overrun vulnerability in IIS servers. It is intended to allow an attacker to have full control of the system by sending the http get request to run scripts/root.exe.

Attack Mechanism:

Codered II propagates itself by by sending Default.ida and a buffer overrun to an unpatched IIS server. The worm exploits a known vulnerability in Microsoft's indexing service which is part of the Microsoft IIS server. Microsoft has released

a patch and bulletin to deal with this problem,
http://www.microsoft.com/technet/security/bulletin/MS01-033
Once infected, the worm copies CMD.EXE to
%Systemdrive%\Inetpub\Scripts\Root.exe and
%systemdrive%\Progra~1\Common~1\System\MSADC\Root.exe. If the Trojan that the worm drops has modified the registry key:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\
Services\W3SVC\Parameters\Virtual Roots

and, by adding a few new keys and setting the user group to 217, a hacker can control the Web server by sending an HTTP GET request to run scripts/root.exe on the infected Web server.

The main thread of the worm goes dormant for 48 hours on Chinese systems and 24 hours on other systems, while the other threads continue to make attempts to infect other IIS servers. When the main thread wakes up from its sleep, it causes the computer to restart. Further, all the threads check whether the month is October or whether the year is 2002. If so, the computer is restarted.

The worm copies the command shell (Cmd.exe) to the default execution-enabled directory of the IIS Web server, allowing remote control. It also drops a file (which has its attributes set to Hidden, System, and Read Only) onto the root drive as either or both C:\Explorer.exe or D:\Explorer.exe. Norton AntiVirus identifies these Trojan files as Trojan.VirtualRoot. The worm carries this file inside itself in a packed format and unpacks it when it is dropped.
The infection lasts 24 or 48 hours, and then the compute is restarted. However, the same computer can be re-infected until it is patched with the latest update from Microsoft.

If the month is October or if the year is 2002, the computer will also be restarted. When the computer is restarted, Trojan.VirtualRoot is executed when the system attempts to execute Explorer.exe (due to the way that Windows NT resolves or searches program paths when executing a program). The Trojan (C:\Explorer.exe) sleeps for a few minutes and resets these keys to assure that the registry keys are modified.

NOTE: After a restart, the memory-resident worm will be inactive, meaning that, on an infected system that has been restarted, the worm will not attempt to spread itself to other machines unless it happens to get re-infected. The Trojan also alters the registry key:

HKEY_LOCAL_MACHINE\Software\Microsoft\
Windows NT\CurrentVersion\Winlogon

so that the value SFCDisable is set to 0xFFFFFF9D. This disables the System File Checker (SFC). System file checker is a feature of Windows that scans the systems files to ensure that they are the correct versions provided by Microsoft.

Additionally, you can see the strings that the worm sends to a web server by looking in the web server log's:

2004-08-09 23:46:31 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /<Rejected-By-UrlScan> ~/scripts/..%c1%1c../winnt/system32/cmd.exe 401 5 www -
2004-08-09 23:46:32 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /scripts/winnt/system32/cmd.exe /c+dir 401 5 www -
2004-08-09 23:46:34 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /winnt/system32/cmd.exe /c+dir 401 5 www -
2004-08-09 23:46:35 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /winnt/system32/cmd.exe /c+dir 401 5 www -
2004-08-09 23:46:36 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /<Rejected-By-UrlScan> ~/scripts/..%%35%63../winnt/system32/cmd.exe 401 5 www -
2004-08-09 23:46:37 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /<Rejected-By-UrlScan> ~/scripts/..%%35c../winnt/system32/cmd.exe 401 5 www -
2004-08-09 23:46:39 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /<Rejected-By-UrlScan> ~/scripts/..%25%35%63../winnt/system32/cmd.exe 401 5 www -
2004-08-09 23:46:40 68.32.113.86 - W3SVC3 WEB MY.NET.HOME.5 80 GET /<Rejected-By-UrlScan> ~/scripts/..%252f../winnt/system32/cmd.exe 401 5 www –

Here is the packet capture of the traffic:

```
02:46:01.772737 IP 68.34.216.159.4715 > MY.NET.WEB.5.80: .
1117379848:1117381296(1448) ack 2524299361 win 64240 <nop,nop,timestamp
614113 0>
0x0000       4500 05dc 2ae7 0000 7406 38c6 4422 d89f   E...*...t.8.D"..
0x0010       c0a8 0005 126b 0050 4299 dd08 9675 c061   .....k.PB....u.a
0x0020       8010 faf0 3fe3 0000 0101 080a 0009 5ee1   ....?.........^.
0x0030       0000 0000 4745 5420 2f64 6566 6175 6c74   ....GET./default
0x0040       2e69 6461 3f58 5858 5858 5858 5858 5858   .ida?XXXXXXXXXXX
0x0050       5858 5858 5858 5858 5858 5858 5858 5858   XXXXXXXXXXXXXXXX
0x0260       0000 e80a 0000 0043 6f64 6552 6564 4949   .......CodeRedII
0x04b0       813c 0347 6574 5075 f581 7c03 0472 6f63   .<.GetPu..|..roc
0x04c0       4175 eb03 4a10 49d1 e103 4a24 0fb7 0c0b   Au..J.I...J$....
0x04d0       c1e1 0203 4a1c 8b04 0b03 c389 4424 2464   ....J.......D$$d
0x04e0       678f 0600 0058 61c3 e851 ffff ff89 5dfc   g....Xa..Q....].
0x04f0       8945 f8e8 0d00 0000 4c6f 6164 4c69 6272   .E......LoadLibr
0x0500       6172 7941 00ff 75fc ff55 f889 45f4 e80d   aryA..u..U..E...
0x0510       0000 0043 7265 6174 6554 6872 6561 6400   ...CreateThread.
0x0520       ff75 fcff 55f8 8945 f0e8 0d00 0000 4765   .u..U..E......Ge
0x0530       7454 6963 6b43 6f75 6e74 00ff 75fc ff55   tTickCount..u..U
```

```
0x0540        f889 45ec e806 0000 0053 6c65 6570 00ff   ..E......Sleep..
0x0550        75fc ff55 f889 45e8 e817 0000 0047 6574   u..U..E......Get
0x0560        5379 7374 656d 4465 6661 756c 744c 616e   SystemDefaultLan
0x0570        6749 4400 ff75 fcff 55f8 8945 e4e8 1400   gID..u..U..E....
0x0580        0000 4765 7453 7973 7465 6d44 6972 6563   ..GetSystemDirec
0x0590        746f 7279 4100 ff75 fcff 55f8 8945 e0e8   toryA..u..U..E..
0x05a0        0a00 0000 436f 7079 4669 6c65 4100 ff75   ....CopyFileA..u
0x05b0        fcff 55f8 8945 dce8 1000 0000 476c 6f62   ..U..E......Glob
0x05c0        616c 4669 6e64 4174 6f6d 4100 ff75 fcff   alFindAtomA..u..
0x05d0        55f8 8945 d8e8 0f00 0000 476c             U..E......Gl
```

I pasted this in with a font of 10. The 12 font was messing the display of the packet. Sections of the packet that did not show the point I am making were removed to save space. As you can see, the Trojan attempted to fill the buffer up with the XXXXXXXX (585858585858585858). Then the packet moves on, where, in plain text, CodeRedII is displayed. After further inspection of the packet, you can see that the Trojan is attempting to run system call's, described in Symantec's write up of this virus. You can see that it is attempting to LoadLibraryA (Map's the executable module into the address space of the calling process), CreateThread, (Creates a thread to execute within the virtual address space of the calling process), GetTickCount (Get's the number of milliseconds that have elapsed since the system was started). Next, it looks like it puts the thread to sleep. Then, it looks to be grabbing the default language of the machine with GetSystemDefaultLangID (this must be to tell the Trojan how long to stay dormant, remember, Chinese will be dormant for 48 hours, everyone else, 24 hours. The Trojan then issues the GetSystemDirectory, which is used to find the system directory on a system. This is where system files such as .DLL's will be located. Next, it issues the CopyFile command, which copies an existing file to another filename. In this case, CodeRedII copies CMD.EXE to ROOT.EXE. The next command run is the GlobalFindAtom command, which searches the global atom table for the specified character string and retrieves the global atom associated with that string. Now, this one threw me off. What the hec is an Atom? Well, per the MSDN, an atom is a normal character, a character class, or a parenthesized regular expression.

Probability source address was spoofed: LOW
CodeRed II Does not care about hiding it's origin. I believe it is also looking for a SYN-ACK from a web server to spread it's virus.

CORRELATRIONS:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/dataexchange/atoms/atomreference/atomfunctions/globalfindatom.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fileio/base/copyfile.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vjref98/html/com.ms.wfc.app.Locale_getDefaultLanguage.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/getsystemdirectory.asp

http://securityresponse.symantec.com/avcenter/venc/data/codered.ii.html

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/loadlibrary.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/createthread.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/gettickcount.asp

http://www.microsoft.com/technet/security/bulletin/MS01-033

Evidence of Active Targeting: No

Codered II targets computers by generating an IP list. The packets are then sent to port 80 (WEB) of the selected IP's. These IP's are generated randomly, the worm is simply looking for a SYN-ACK response from the server to continue on it's merry little way.

Severity:

Criticality = 5

The only machine on my network that answers to a call to port 80 is my web server. I hold this machine to be very important to me.

Lethality = 5

If this attack were to succeed, an attacker would have full control of my web server. Once the attacker has full control of that server, he or she could then start attacking other machines on my internal network, use my web server to attack other machines on the big I internet.

System Countermeasures = 4

The machine in question is a fully patched IIS web server. It is running NAV Corporate and SNORT IDS on the host itself. All services that I do not use (FTP, POP, Terminal services, ect.) have been disabled. You might be wondering why if the system is fully patched and protected, then why didn't this value earn the

highest score? Well, although it is fully patched and protected, it is still an Microsoft IIS server, which, seems to be the preferred servers to exploit on the internet.

Network Countermeasures = 2

I have a firewall in place at my point of presence, but I allow port 80 through the firewall, being forwarded to my web server. My firewall does not allow me to filter by the packet content's itself, therefore, the network countermeasures are low.

Taking all this in mind, here is the severity rating:

(5+5) – (4+2) = 4

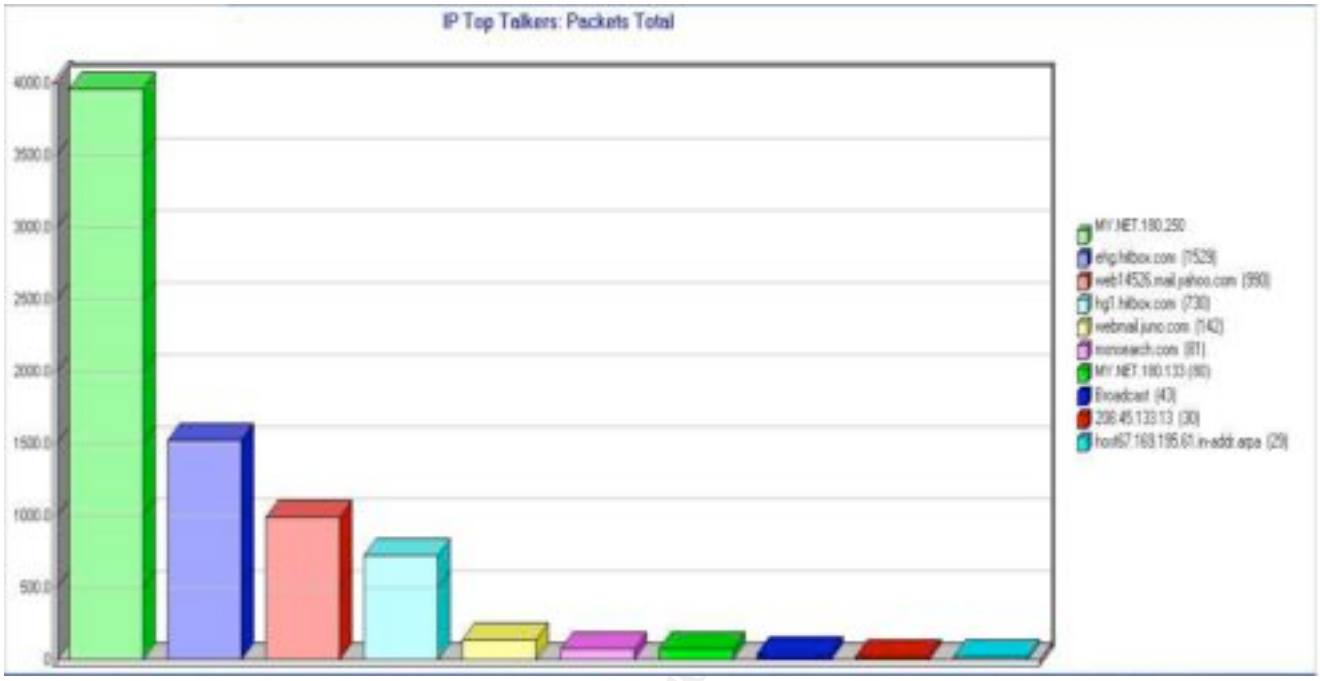
Network Statistics:

The following is a list of the top five talkers in terms of total # of packets, either Transmitted or received, from the 2002.5.10 file:

DNS NAME	IP ADDRESS	Packets Rx	Packets Tx	Packet's Total
	MY.NET.180.250	111	3850	3961
ehg.hitbox.com	64.154.80.51	1529	0	1529
web14526.mail.yahoo.com	216.136.224.55	990	0	990
hg1.hitbox.com	64.154.80.50	730	0	730
webmail.juno.com	64.136.20.82	142	0	142

As you can see, the majority of the packet's in the dataset are coming from MY.NET.180.250. I believe this address to be one of the routers/natting devices that the snort sensor is between.

Here is a graphical depiction if this data, with the caveat that this chart depicts the top 10 talkers by packet's sent or received. Note that this graph is outside of the alert's that snort has generated, this is simply ALL traffic from the 2002.5.10 file:



Here are the top 5 talkers in terms of **SOURCE ADDRESSES THAT GENERATED ALERT'S:**

DNS NAME	IP ADDRESS	# of occurrences
MY.NET.180.250	MY.NET.180.250	462
	255.255.255.255	36
	203.122.47.137	11
ppp102-110.pppcal.vsnl.net.in	203.197.102.110	9
custnets-63-70-83-162.rinc.net	63.70.83.162	8

On top of this list, I want to add a chart that shows the top talkers in terms of **DESTINATION** address:

DNS NAME	IP ADDRESS	# of occurrences
ehg.hitbox.com	64.154.80.51	361
hg1.hitbox.com	64.154.80.50	38
web14526.mail.yahoo.com	216.136.224.55	18
	64.94.89.210	9
MY.NET.180.250	MY.NET.180.250	7

Now that we have a list of the top IP's that triggered alerts, and the top IP's by packet's sent and received, let's look at the port's that are being targeted on the network:

DST PORT	SERVICE ASSOCIATED	# OF OCCURENCES
80 TCP	WEB TRAFFIC	495
53 UDP	DNS	42
515 TCP	Print Spooler	36
63599 TCP	Not Well Known port	5
0	No association	2

As you can see, the web traffic has the highest # of occurrences for alerts. DNS is also being targeted, this is held true by the DNS Named version attempts that snort triggered on.

There is evidence of a network sweep for port 515. A crafted packet with the source IP of 255.255.255.255 is responsible for this port being targeted. I saw no evidence of successful penetration to any machines using this port.

Port 63599 is not a well known port, it is an ephemeral port, meaning, any port above 1024. In looking at the connection that is using this port, a user on the internal seems to have connected to an FTP site. After further inspection of the site, which was wide open to anonymous user's, it looks to be a Linux computer, or at least it has a Linux file system, eg. /usr /bin ect, on an AOL server. The machine that connected to this site should be examined for malicious activity. Unfortunately, the destination is the MY.NET.180.250 address. We will need to inspect the logs of that routing/natting device to track this connection to the actual machine.

Port 0 is interesting because it is just not a normal thing to see. Usually, as with the case of many connections from this file, this is an indication of crafted packets. Depending on the intent of the packet, ie. Recon, exploit ect. We cannot trust the validity of the source IP generating this packet. No specific service is being targeted with this port, the packets associated with this port show indication of an attempt to evade the IDS.

Now, looking at all the total's from the data previously presented, It is my opinion that the 3 most suspicious EXTERNAL ip's are as follows:

1) 218.2.129.171 – This IP only show's up twice, but it is extremely suspicious to me due to the fact that it seems to be a "low and slow" attacker. I came to this conclusion with this fact in mind; the attacker hit's again in the 2002.5.11 file, with the same low number of occurrences. The attacker seems to be attempting to hide himself between all of the other attacks. The fact that this IP is coming out of China should also raise

concern levels. This individual is attempting to evade the IDS, which is a major concern of mine.

Registration:

Server Used: [whois.apnic.net]

218.2.129.171 = [ ]

inetnum: 218.2.0.0 - 218.4.255.255
netname: CHINANET-JS
descr: CHINANET jiangsu province network
descr: China Telecom
descr: A12 Xin-Jie-Kou-Wai Street
descr: Beijing 100088
country: CN
admin-c: CH93-AP
tech-c: CJ186-AP
mnt-by: MAINT-CHINANET
mnt-lower: MAINT-CHINANET-JS
mnt-routes: maint-chinanet-js
changed: hostmaster@ns.chinanet.cn.net 20020209
changed: hostmaster@ns.chinanet.cn.net 20030306
status: ALLOCATED non-PORTABLE
source: APNIC
route: 218.2.0.0/16
descr: CHINANET jiangsu province network
country: CN
origin: AS23650
mnt-by: MAINT-CHINANET-JS
changed: ip@jsinfo.net 20030414
source: APNIC
role: CHINANET JIANGSU
address: No.268 Hanzhong Road Nanjing 210029
country: CN
phone: 86-25-6588783
fax-no: 86-25-6588740
e-mail: ip@jsinfo.net

trouble: send anti-spam reports to spam@jsinfo.net

trouble: send abuse reports to abuse@jsinfo.net

trouble: times in GMT8
admin-c: CH360-AP
tech-c: CS306-AP
tech-c: CN142-AP
nic-hdl: CJ186-AP
remarks: www.jsinfo.net
notify: ip@jsinfo.net

mnt-by: MAINT-CHINANET-JS
changed: dns@ptt.js.cn
20020530
changed: ip@jsinfo.net
20021213
source: APNIC
person: Chinanet Hostmaster
address: No.31 jingrong street beijing
address: 100032
country: CN
phone: 86-10-66027112
fax-no: 86-10-58501144
e-mail: hostmaster@ns.chinanet.cn.net

e-mail: anti-spam@ns.chinanet.cn.net

nic-hdl: CH93-AP
mnt-by: MAINT-CHINANET
changed: hostmaster@ns.chinanet.cn.net
20021016
remarks: hostmaster is not for spam complaint please send spam complaint to anti-spam@ns.chinanet.cn.net

source: APNIC

2) 64.154.80.51 – This IP is suspicious in that it is associated with the hitbox counter. There are numerous connections from MY.NET.180.250 to this server. Hitbox has been listed as spyware. The data being sent look's to be encrypted; that is, it does not contain human readable data. It would be my intention to locate the machine that is sending data to this site, and perform extensive forensics, that is, take an image using DD, and possibly run through the content's using Autopsy. (Yes, I am also preparing for the track 8 practical)

Ref: http://www.pestpatrol.com/PestInfo/e/ehg-tmgolf_hitbox.asp

Registration:

Server Used: [whois.arin.net]

64.154.80.51 = [ehg.hitbox.com]

OrgName: Level 3 Communications Inc.
OrgID: LVLT
Address: 1025 Eldorado Blvd.
City: Broomfield
StateProv: CO
PostalCode: 80021
Country: US
NetRange: 64.152.0.0 - 64.159.255.255
CIDR: 64.152.0.0/13
NetName: LC-ORG-ARIN
NetHandle: NET-64-152-0-0-1
Parent: NET-64-0-0-0-0
NetType: Direct Allocation
NameServer: NS1.LEVEL3.NET
NameServer: NS2.LEVEL3.NET
Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate: 2000-06-08
Updated: 2001-05-30
TechHandle: LC-ORG-ARIN
TechName: level Communications
TechPhone: 1-877-453-8353
TechEmail: ipaddressing@level3.com

OrgAbuseHandle: APL8-ARIN
OrgAbuseName: Abuse POC LVLT
OrgAbusePhone: 1-877-453-8353
OrgAbuseEmail: abuse@level3.com

OrgTechHandle: TPL1-ARIN
OrgTechName: Tech POC LVLT
OrgTechPhone: 1-877-453-8353
OrgTechEmail: ipaddressing@level3.com

OrgTechHandle: ARINC4-ARIN
OrgTechName: ARIN Contact
OrgTechPhone: 1-800-436-8489
OrgTechEmail: arin-contact@genuity.com

ARIN WHOIS database last updated 2004-11-05 19: 10
Enter ? for additional hints on searching ARIN's WHOIS database.

3) 61.194.28.1 – This seems to be yet another low and slow attacker. This IP has sent one packet, utilizing the TCP protocol, with 0 bytes of data in the TCP header. This seems to be another firewall/IDS evasion tactic that does not look to be successful.

Registration:

Server Used: [whois.nic.ad.jp]

61.194.28.1 = [fws.futabaunyu.co.jp]
[ ]
InfoSphere (NTT PC Communications Inc.)
SUBA-029-392 [Sub Allocation] 61.194.28.0
futabaunyu. corp
FUTABA-LOGI [61.194.28.0 <-> 61.194.28.7] 61.194.28.0/29

CORRELATIONS:

MSDN correlations for detect 3:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/winui/windowsuserinterface/dataexchange/atoms/atomreference/atomfunctions/globalfindatom.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fileio/base/copyfile.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vjref98/html/com.ms.wfc.app.Locale_getDefaultLanguage.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/getsystemdirectory.asp

http://securityresponse.symantec.com/avcenter/venc/data/codered.ii.html

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/loadlibrary.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/createthread.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/gettickcount.asp

http://www.microsoft.com/technet/security/bulletin/MS01-033

Symantec's website:
http://securityresponse.symantec.com/avcenter/venc/data/codered.ii.html

Michael Meacle's practical:
http://www.giac.org/practical/GCIA/Michael_Meacle_GCIA.pdf

Al Maslowski Yerges practical:
http://www.giac.org/practical/GCIA/Al_Maslowski-Yerges_GCIA.pdf

Security focus article on IDS evasion techniques:
http://www.securityfocus.com/infocus/1577

Security Considerations for IP Fragment Filtering:
http://www.faqs.org/rfcs/rfc1858.html

Possible Compromise:

Although it is tough to pick out any signs of compromise using the data provided, there is a strong indication of spyware on the network. MY.NET.180.250 (you remember, that natted device we love to hate) show's multiple connections to a known spyware source.

MY.NET.180.133 seems to have a considerable amount of web requests sent to this machine. Some of the more notable ones are those that send the default.ida request to this machine. This could possibly be a web server setup by the university, an individual set this up on their own, or the machine has been compromised by a Trojan that sets the machine up to listen on port 80. There are numerous worms and Trojans that operate in this manner, here is a link to the SANS port 80 detailed report: http://isc.sans.org/port_details.php?port=80 This machine has also had FTP requests sent to it. This looked to be active targeting, as there were multiple attempts to port 21 on this box. This machine should require further inspection to obtain a clearer indication of positive compromise or not.

MY.NET.180.250 also has multiple GNUTELLA connections going to and from this address. I am unaware of the universities policies on file sharing, but this would be something to look at. What exactly is being shared on your network? We will need inspection of the offending machines to decipher this.

Analysis Process:
1) Downloaded log files from http://isc.sans.org/logs/raw onto Windows XP, Redhat 9.0.

2) Ran files through SNORT v2.2 with default ruleset:
Snort –r 2002.5.10 –c /etc/snort/etc/snort.conf –l /var/log/snort –X –k none

3) Used Ethereal to view entire dataset.

4) Imported 2002.5.10 logfile into ACID and MySQL database.
Here is the install guide used for this setup.
http://www.snort.org/docs/Snort_SSL_FC2.pdf

5) Used Observer to achieve graph of Top 10 talker's and to view other network statistics. Ie. Total packets Tx and Rx and Total Bytes Tx and Rx.
http://www.networkinstruments.com/assets/pdf/observer_brochure.pdf

6) Implemented Michael Meacle's method to map out network.

ACID and SNORT was setup on a 1.7GHZ x86 box running Redhat 9.0.
List of programs used and their versions:
ACID version 0.9.6b23
SNORT version 2.2
WINDUMP version 3.6.2
ETHEREAL version 10
NEMESIS 1.4 for Windows
OBSERVER version 9 by Network Instruments.

Most of the analyzing was performed from a Window's XP 2.8GHZ machine with 1GB RAM. Snort, nemesis, ETHEREAL, WINDUMP, and OBSERVER were all loaded on this machine.

Defensive Recommendation's:

Detect 1: Backdoor Q network sweep.
This attack, like almost all other's out there, seems to target a specific OS, Linux/Unix. One of the key defense policies to defend against OS specific threats is to ensure the OS is fully patched. This will mitigate the risk of compromise in many situations. If there are any known Linux/Unix machines on the network, it would be wise to ensure all users understand the patching process, and the importance of keeping their systems up to date. If possible, a network vulnerability scanner should be used to assist in weeding out vulnerable machines. A process such as this could save the organization a lot of time and money. Another defensive move to make in this chess match we call network security would be to block all incoming traffic to the broadcast address space, if it is not already. From this, you would be able to eliminate the possibility of traffic such as this coming from outside of your network, and can begin to track down the source on the internal side of the network. In this case, port 515 should be blocked at the firewall from entering the network as well.

Detect 2: Tiny Fragments.
In the case of fragmentation attacks, IDS evasion, it is best to ensure that you are running a network intrusion detection system, such as SNORT, and that the NIDS is fully patched and up to date. Also, as stated above, ensure that an OS patching policy is in place. Make sure that your NIDS properly performs packet reassembly for inspection. Check with your firewall vendor for known issues and patches that may exist to prevent fragmentation attacks, as well as DoS attacks.

Detect 3: Codered II
IIS has many vulnerabilities associated with it. As such, Microsoft has release patches to address these issues. In the case of Codered, it is recommended to apply the patch http://www.microsoft.com/technet/security/bulletin/MS01-033