Global Information Assurance Certification Paper

Intrusion Detection In Depth
GCIA Practical Assignment
Version 3.0 (revised August 13, 2001)

David Jenkins

Table of Contents

Assignment 1 - Describe the State of Intrusion Detection

Challenge – Trying to detect stealth attacks with Network Based Intrusion Systems.

Introduction

Most popular Network Based Intrusion Detections Systems (NIDS) utilize a rule base to match against known attacks. Using rules to match with, can create a number of problem in trying to detect stealthy attacks. Also the placement of the sensor can contribute to further problems in detecting attacks. Hackers, knowing how NIDS work, deliberately craft packets and create attacks to try and avoid detection by NIDS. In this paper, several of these methods are clearly stated to show the challenge at hand.

Avoiding a content pattern match by changing the way the target is referenced.

A common form of attack is exploiting vulnerabilities in the http cgi-bin scripts and programs. As a simple example a possible NIDS rule logic to detect any attacks with this method on the /etc/passwd file may be as in Rule 1 below. A hacker trying to target the /etc/passwd file with Attack 1, would trigger the alert from Rule 1. To avoid detection with Rule 1, a hacker could instead send Attack 2. Attack 2 would match on the first part of rule 1, but "/etc/./password" does not match "/etc/passwd". Therefore this attack would not trigger an alert based on Rule 1. The reason Attack 2 can still reference the target /etc/passwd file is because the target system translates /etc/./passwd to /etc/passwd. In practice, all the attacks from Attack 2 to Attack 7 translate to the same value of /etc/passwd. Each of these attacks use a different method of references the same file. A NIDS system has to try to also interpret these in the same way a web server would, and then match the interpreted string to the database. This can become more difficult when combinations of different references systems are used as in Attack 7.

To try and overcome these types of veiled attacks, Rule 1 could be replaced with Rule 2. Rule 2 would definitely detect attacks 1-5, and depending of the way the NIDS translate the reference string may detect attack 6 and 7. But one can quickly see here, that there is practically an unlimited way of combining these veiled attacks and thus complicating detection of such attacks.

Rule 1:
If "/cgi-bin" and "/etc/passwd" are in the packet content, send an alert.
Rule 2:
If "/cgi-bin" & "/etc" & "passwd" are in the packet content, send an alert.

Attack 1:
http://www.target.com/cgi-bin/php.cgi?/etc/passwd

Attack 2:
http://www.target.com/cgi -bin/php.cgi?/etc/./passwd
Attack 3:
http://www.target.com/cgi -bin/php.cgi?/etc/././.passwd
Attack 4:
http://www.target.com/cgi -bin/php.cgi?/etc//passwd
Attack 5:
http://www.target.com/cgi -bin/php.cgi?/etc/ *subdir*/../passwd
Attack 6:
http://www.target.com/cgi -bin/php.cgi?/etc/%70%61%73%73%77%64
Attack 7:
http://www.target.com/cgi -bin/php.cgi?/etc/./.subdir/../%70%61%73%73%77%64

To be more effective in bypassing detection, a hacker could download t he NIDS rulebase and see what strings are actually being searched for. For example, the Snort rulebase is publicly available for anyone to download. If there was an attack the hacker wanted to use and found that it was being looked with the NIDS then they could use an available method to pad and change the attack signature so as to not trigger an alert. If the NIDS rulebase was not available for download, then another avenue for the hacker could be to attempt to source an install disk of other vendor's N IDS systems, or download a pirate or cracked copy. Once downloaded, if they couldn't read the rules directly, actually install the software on a system in their test network, and see if the system detects their new veiled attacks.

As a specific example, the following is a rule taken from the Snort web - attacks.rules file:

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB -ATTACKS kill command attempt"; flags:A+; ***content:"/bin/kill"*** *;nocase; sid:1335; rev:1; classtype:web-application-attack;)*

Here we can see the rule is looking for the content "/bin/kill". The attacker could try and veil their attack by putting in any combination non -standard references. ie "/bin/./././kill" or /bin/%10%10%10%10. A further example from the snort web - attacks.rules file

> *alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB - ATTACKS chown command attempt"; flags:A+;*
> ***content:"/usr/sbin/chown"*** *;nocase; sid:1338; rev:1; classtype:web - application-attack;)*

We see here that the content being searched for is "/usr/sbin /chown". The attacker could try and veil the
attack by sending "/usr/sbin/../sbin/chown" instead.

Sample evidence of this type of attack veiling can be seen taken from an IDS log over the months of January and February, 2002. The attacks came from an I P

address registered to a Chinese Petrochemicals company and were targeted at a bank's web server.

Sample evidence:

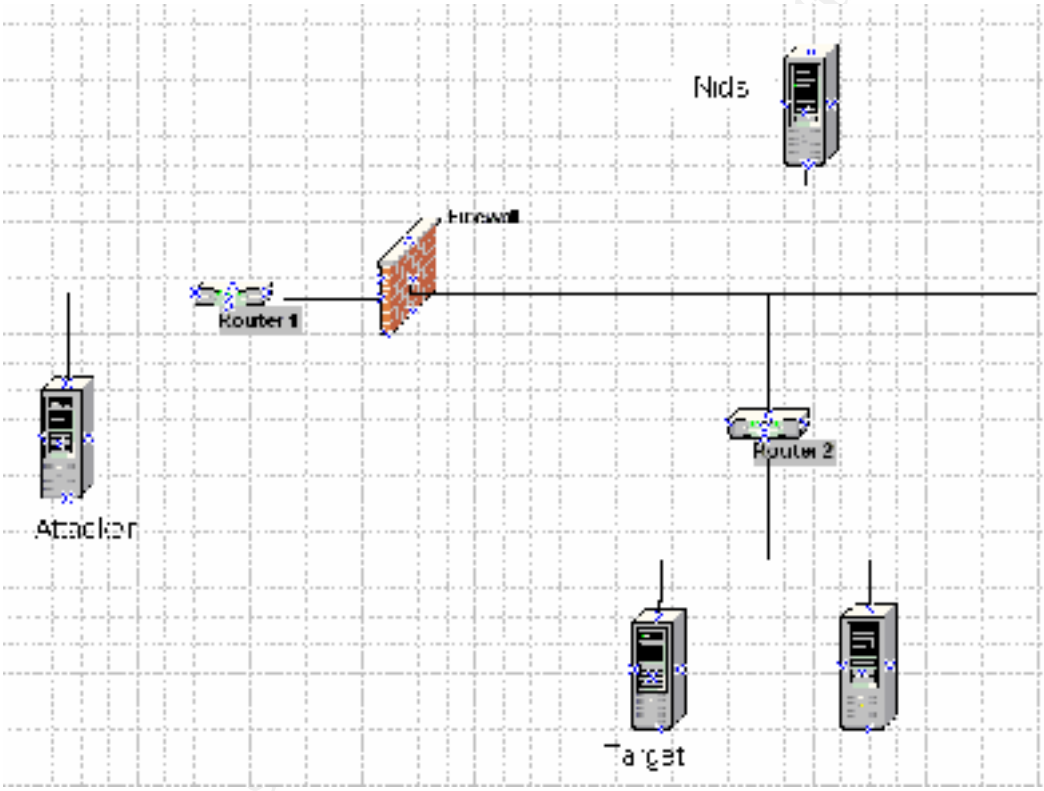
```
GET /c/winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET
/msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%
c1%1c.. /winnt/system3 2/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET
/_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.e
xe?/c+dir HTTP/ 1.0
GET
/_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.e
xe?/c+dir HTTP/1.0
GET /scripts/..%%35c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%252f../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%255c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
```

The first request does not attempt to veil the attempt. Although the rest of the thirteen attempts to run cmd.exe used the Unicode translation, with a likely intend to avoid detection.

Packet insertion

Packet insertion can be used to avoid detection by the NIDS. This works by sending the attack in a number of packets with the payload split across t hem to the destination, and inserting a bogus packet in the middle of the attack, that the NIDS sees but not the target.

Diagram 1.0



If the intended attack consists of attack string 'abcd', and the NIDS has this attack string in it's database. The n normally the NIDS will alert when the attacker sends the attack. To avoid this, the attacker can break up the attack into three packets.

Packet 1 payload = ab
Packet 2 payload = xy
Packet 3 payload = cd

Send packets 1,2,3 down the wire and per form some packet crafting to ensure that the target only sees packet 1 and 2. Effectively the NIDS system has seen the attack signature of 'abxycd', but the target system has received the attack signature of 'abcd'.

One way to achieve this result is to modify the Time to Live (TTL) field in the packet. As can be seen in diagram 1.0 the target is one more network hop away than the NIDS is from the attacker. If the attacker can set the TTL, so that it's value is 1 when it reaches Router 2, then at that point the packet will be discarded, effectively meaning the NIDS has seen the packet, but the target never gets to see the packet. Therefore with the above packets as an example, the attacker would set the TTL as follows:

Packet 1 payload = ab TTL = 55
Packet 2 payload = xy TTL = 3
Packet 3 payload = cd TTL = 55

In this case, packet 2 will have a TTL = 1 when it reaches Router 2 and will be discarded, effectively the target will not see packet 2.

Another method to do this, is to craft the IP or TCP checksum field. This way can work, if the NIDS does not discard a packet because of a bad checksum, when the target does. Using the sample 3 packets again. The attacker would assign the following for the packets:

Packet 1 payload = ab Valid checksum
Packet 2 payload = xy Bad checksum
Packet 3 payload = cd Valid checksum

In this case, again the NIDS sees all packets, summing to attack string 'abxycd'. The target, discards the second packet and therefore receives a valid attack string of 'abcd'.

A third variation on packet insertion is IP fragmentation overlap. If the NIDS system handles the fragmentation reassembly different to the way the target system does, then the attack string will not appear to the NIDS as it does appear to the target, thus achieving packet insertion. One may ask, why not just program the NIDS to reassemble the packets the same way as the target. This is perfectly possible for one type of operating system. But unfortunately for NIDS, not all TCP/IP stacks are programmed the same. Same react differently to IP fragmentation overlaps. It would be possible to program the NIDS to re-assemble the packets in all possible operating system ways, and check each reassembly for an attack signature, but this would slow down the operation of the NIDS. And in many cases, we are trying to speed up how much traffic NIDS can handle.

A similar type of insertion as above is the TCP sequence overlapping method. This is executed by the attacker reusing sequence numbers. Once again, different operating system TCP/IP stacks will react differently. For the NIDS to detect this attack, it needs react the same was as the target does.

The Maximum Transmission Unit (MTU) can also be used for packet insertion. If there is a situation where after route 2 from Diagram 1.0, has a smaller MTU

value than the network segment where the NIDS is sitting, then the attacker can craft the packets as follows:

Packet 1 payload	= ab	MTU valid for both network segments
Packet 2 payload	= xy	MTU to large for network segment after router 2
Packet 3 payload	= cd	MTU valid for both network segments

Once again, the NIDS sees all the packets, while packet two is stopped at router 2 so the target only sees packets 1 and 3.

Lastly, another insertion type is to utilize the scheme of "Prevent Against Wrapped Sequence numbers" (PAWS). If the NIDS system does not have react the same way as a target host that does have PAWS, then the NIDS system could accept all packets, while the target system rejects one of the p ackets. The packets to perform this would have values as below

Packet 1 payload	= ab	Valid Sequence number
Packet 2 payload	= xy	Old sequence number, will be dropped by PAWS
Packet 3 payload	= cd	Valid Sequence number

Packet evasion

An alternative method from inserting bogus packets, is to try and evade the NIDS from reading the packets.

This can be done by tricking the NIDS into thinking that the TCP session has finished. If we have an established TCP session in operation, and then a TCP Reset is sent to the target, but with a low TTL that will not reach the target, but will reach the NIDS. The NIDS will think the session is finished and then may not monitor further packets in that session.

Lastly, it is possible to send data on the TCP SY N packet. Some NIDS may not confirm to this which is guided out in the associated RFC that this is possible. If the NIDS ignores the data on the SYN packet, but the target buffers this data, then actions it once the connection is established, we have an evasion of the NIDS.

Further details of these types of attacks can be referenced at www.robertgraham.com/mirror/Ptacek -Newsham-Evasion-98.html

Morphing the attack, so as n ot to match the signature.

One of the regular methods to detect attacks is to look for common ports in either the source port or destination port of the packets. If we know that a certain trojan listens on port 55443 . Then a rule to match on this would be to look for any packets with a destination port of 55443.

For example, the following snort rule is used to detect an attempt to access the 'Girlfriend' trojan remotely.

Rule 3:
alert tcp $EXTERNAL_NET !80 -> $HOME_NET **21554** (msg:"BACKDOOR GirlFriendaccess"; flags: A+; content:" **Girl**"; reference:arachnids,98; sid:145; classtype:misc-activity; rev:3;)

In the above rule, the highlighted number part defines what destination port number is being looked for.

To avoid triggering this alert, the hacker can m odify the trojan, so instead of the trojan listening on port 21554, it will listen on 2536 instead. Then once the modified trojan has been successfully implemented on the target, the attacker can then connect to port 2536 without triggering Rule 3.

Secondly, we can see in Alert one, that the text Girl is highlighted also. The author of the rule in an attempt to reduce the false positives of snort, put a second check in the rule, not only does the packet have to be destined for port 21554 but also the con tent of the packet has to be Girl. This is effective in cutting down the false positives, but can make the job of avoiding detection by the snort system easier. The attacker instead of changing the port number the trojan listens on, can change the string in the content of the packet. So instead of 'Girl', the content of 'aa3re' can be put in place. Again, once the trojan has been changed to do this, implemented on the target, and an attempt to connect to a Girlfriend trojaned server is attempted, Rule 3 will not detect the attempt.

The following web sites contains the source code of a number of trojans:
http://www.tlsecurity.net/sourcecodeb.html

There is full source code for the Donald Dick Tr ojan there. To how easy it would be to perform as what's described above, the author has conveniently created a ddsetup.ini file. In their file there are the predefined port numbers, passwords….etc. Here is a paste from part of the file.

.
.
[Configura tionxxx]
SPX_ports = D,0x9015
TCP_ports = D,23477
Password = FuckYou
Hidden = 1

```
KeepPreviousSettings = 0
EraseItself = 1
;MailTo = FILE:ddsyslog.xxx
MailTo = yaworsky
MailOnce = 0
SMTP_relay = 192.168.1.1
MailWhenServerCrash = 1
MailHowManyLogRecords = 0
ServerName = ddd.exe
W9XServerName2 = ddd9x.xxx
WNTServerName2 = dddnt.exe
W9XLoaderName = vxdxd.vxd
WNTLoaderName = bootfuck.exe
W9XHomeKey = software \dodick9Xhome
WNTHomeKey = software \dodickNThome
ParamsVName = params
ChatVName = nvchat
EventName = pizdez
ACLName = ddacl.zzz
.
.
```

The snort rule for detecting the trojan is as follows:

Rule 4:
alert tcp $HOME_NET **23476** -> $EXTERNAL_NET any (msg:"BACKDOOR DonaldDick 1.53 Traffic"; flags: A+; content:"pINg"; sid:153; classtype:misc -activity; rev:3;)

Now it is easy to see that by changing the line 'TCP_ports = D,23477' to 'TCP_ports = D,2536, would avoid the snort detect if snort was looking for port 23477.

A final note about Donald Dick and other trojans is that is has encryption capabilities, which in e ffect stops the NIDS system being able to match on the content of the packet.

Take the NIDS out of action.

Finally, a less elegant way, but still effective is to somehow take the NIDS out of action, if not completely, so overload it, that it is of no im mediate value. Once the NIDS is out of the picture, a hacker can then attack the target systems without fear of detection through the NIDS.

There are a number of ways of achieving this goal, and below is described a few methods and references provided.

Crash the NIDS

This can be performed by either sending crafted packets with non -standard fields set that the NIDS system is not expecting and has not been programmed to expect and then crashes. An example of these was to send fragmented packets with the SYN bit set. Two references to this effect are as follows:

http://www.tlsecurity.net/cgi-bin/framer.pl?http://www.tlsecurity.net/archive/exploits/08_00/%5bEXPL.25.08.00%5d.Relsecure.DoS.txt
http://www.safermag.com/html/safer28/dos/05.html

Internet Security Systems (ISS) released an article January 2002, regarding the possibility of crashing snort. Here is an extract from the article

"it may be possible for remote attackers to send specially crafted ICMP [internet control message protocol] packets to the program, resulting in a segmentation fault that would crash the Snort engine"

A news article can be found on this, which includes the author of Snort's response.

http://www.vnunet.com/News/1128794

Another way reported to work is for an attacker to write a script or program, that sends attack packets as fast as possible at the target network. The goal of this type of attack is to overload the NIDS program with the intention of crashing the NIDS software.

A report of this type in regards to DoS'ing Computer Associates eTrust Intrusion Detection system can be found at Computer Associates web site at the following URL.

http://www3.ca.com/Virus/Threat.asp?ID=83

Overload the logs

This method doesn't even require a vulnerability in the Intrusion software. It is a simple manner of filling up the system logs so that there is no more room left on the operating system that the NIDS is running on and either, crashes, or overwrites the logs.

Further to the above point, even if the disk space is not filled up, then this attack can still be effective, where it would be like trying to find the needle in the haystack. There are so many noisy attacks, that the hacker can then slip the real one in the middle of everything. Either the real attack will never be noticed or it will take some time before it is discovered.

Conclusion

As has been shown above, there are numerous methods of nullifying the network intrusion detection system. Some are fairly technical and require a fair degree of skills in packet crafting, while others are very simple and can be still as effective.

Assignment 2- Network Detects

Network Detect 1

Date	Time	Src IP	Src Prt	Dest IP	Dst Prt	Prot	IDS Alert
14/02/2002	02:29	63.34.212.73	51626	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.212.73	51682	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.212.73	51683	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.212.73	51690	a.b.c.40	80	6	URL_Data_IIS Unicode Translation v2
14/02/2002	02:34	63.34.212.73	51786	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:35	63.3 4.212.73	51861	a.b.c.40	80	6	HTTP_TestCgi
14/02/2002	02:35	63.34.212.73	51860	a.b.c.40	80	6	HTTP_TestCgi
14/02/2002	02:35	63.34.212.73	51854	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:35	63.34.212.73	51852	a.b.c.40	80	6	HTTP_Uni x_Passwords
14/02/2002	02:35	63.34.212.73	51843	a.b.c.40	80	6	HTTP_WebSite_Uploader
14/02/2002	02:36	63.34.212.73	51946	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:36	63.34.212.73	51947	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:37	63.34.212.73	52023	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:38	63.34.212.73	52111	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:38	63.34.212.73	52094	a.b.c.40	80	6	HTTP_PHF
14/02/2002	02:39	63.34.212.73	52140	a.b.c.40	80	6	HTTP_NphTestCgi
14/02/2002	02:40	63.34.212.73	52197	a.b.c.40	80	6	HTTP_Netscape_PageServices
14/02/2002	02:40	63.34.212.73	52190	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:43	63.34.212.73	52288	a.b.c.40	80	6	HTTP_IndexServer_Webhits
14/02/2002	02:43	6 3.34.212.73	52293	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:46	63.34.212.73	52430	a.b.c.40	80	6	HTTP_IndexServer_Webhits
14/02/2002	02:46	63.34.212.73	52431	a.b.c.40	80	6	HTTP_IndexServer_Webhits
14/02/2002	02:46	63.34.212.73	52472	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:47	63.34.212.73	52506	a.b.c.40	80	6	URL_Data_cmd.exe Request
.							
.							
5,921 more detects							
.							
.							
14/02/2002	18:59	63.34.212.73	41993	a.b.c.103	80	6	URL_Data_IDA file access attempt
14/02/2002	19:01	6 3.34.212.73	42072	a.b.c.102	80	6	HTTP_Unix_Passwords
14/02/2002	19:01	63.34.212.73	42087	a.b.c.102	80	6	HTTP_Unix_Passwords
14/02/2002	19:02	63.34.212.73	42114	a.b.c.102	80	6	HTTP_Unix_Passwords
14/02/2002	19:02	63.34.212.73	42116	a.b.c .102	80	6	URL_Data_IDA file access attempt

1. Source of Trace.

This network detect originated from where I currently work and manage the NIDS systems.

2. Detect was generated by:

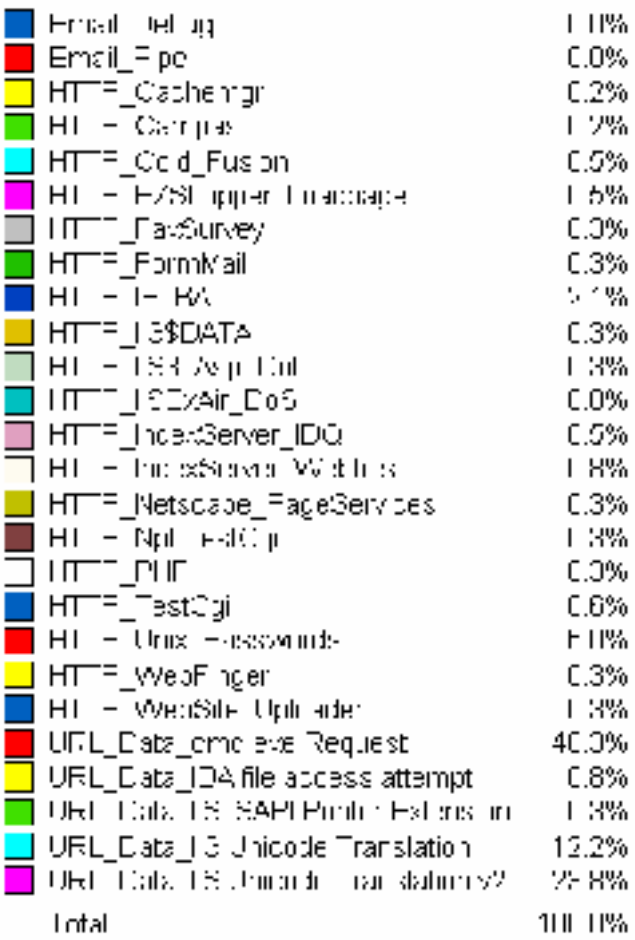
The detect was generated by ISS RealSecure intrusion detection system and the data reviewed and reported with Crystal Reports.

3. Probability the source address was spoofed:

The probability of the source address being spoofed is low for a couple of main reasons. Firstly the column in the detect headed "Prot" is the Proto col of the packet. All the detects had a value of 6. IP Protocol number 6 is TCP, which requires a three way handshake. This in conjunction with the fact that the attacker is performing a number of attacks against the system which to be worthwhile to p erform require feedback. The very first being "HTTP_Unix_Passwords". For this to work, they are trying to obtain the Unix /etc/passwd file. If the address is spoofed then they would not receive any file if the attack was successful, so there would be no sense in performing this specific attack. The same with the other attacks.

4. Description of attack:

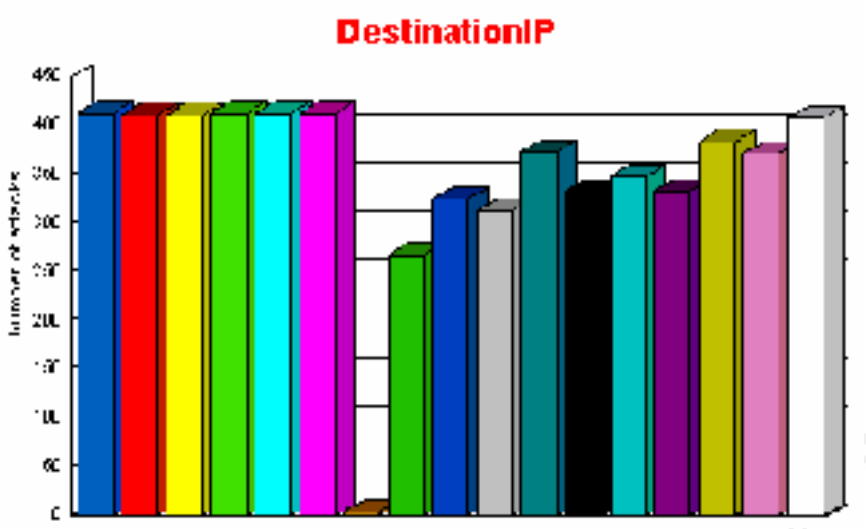
The entire attack consists of 5,950 attacks attempts. The breakdown from this total of each of these attacks is as follows:
Attack type breakdown tally 1.0



We can see that from the first part of the network detect, that the destination IP address is the same, but that the attack changes. Then looking at the second part of the network detect, we see that the destination address is a new one, but then the next line, another new destination IP address is targeted with again a number of different attacks types. It seems evident then that an IP address is targeted, a number of attacks are then tested against the destination. Once those attacks are exhausted, then the next IP address is tested, repeating the attacks types and so on, until all the IP addresses have been tested.

The following graph shows the spread of attacks against 17 different IP addresses.

Target breakdown graph 1.0



From this graph we can see that the attacker is indeed attacking a number of different systems. Also, at first glance is obvious that the first six IP addresses and the last one have the same total number of attacks against them, whilst the others have varying amounts. T his same amount for a number of different systems suggest a tool is being used. Backing up this assumption further, is the fact there are 2-5 attacks every minute being generated over a time span of approximately 16.5 hours for a total of 5,950 attacks.
We also see that the target destination port is always port 80. So we can assume it is a web attacking tool.

Looking at http://www.insecure.org/tools.html "Top 50 Security Tools", placed at number one is the Nessus tool with the following short description: "Remote network security auditor, the client The Nessus Security Scanner is a security auditing tool. It makes possible to test security modules in an attempt to find vulnerable spots that should be fixed." This sounds like it could be a match. The next possibility down the list is Whisker with the description: "Rain.Forest.Puppy's excellent CGI vulnerability scanner ". This too is a possibility. Then next is Internet Security Scanner, followed by Cybercop, Satan and so on. We see that there are quite a few possibilities, so the next stage is to find out if there is a telltale sign in the packets themselves.

Here is a sample of the raw data from a few of the attacks:
GET/iissamples/issamples/oop/q sumrhit.htw?CiWebHitsFile=/iissamples/issampl
es/oop/qsumrhit.htw&CiRestriction=none&CiHiliteType=Full HTTP/1.1
Connection: Close
Host: a.b.c.40
Pragma: no-cache
User-Agent: Nessus/1.0 [en] (X11, U; Nessus)
Accept: image/gif, image/x -xbitmap, image/jpeg, im age/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
GET/iissamples/issamples/oop/qfullhit.htw?CiWebHitsFile=/iissamples/issamples
/oop/qfullhit.htw&CiRestriction=none&CiHiliteType=Full HTTP/1.1

Connection: Close
Host: a.b.c.40
Pragma: no-cache
User-Agent: Nessus/1.0 [en] (X11, U; Nessus)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

From this raw data we can see a very obvious telltale sign. *User-Agent: Nessus/1.0 [en] (X11, U; Nessus)* . From this we can say that the tool being used is Nessus. There is a possibility of another tool being used, which pretends to be Nessus, but the most likely answer is that the tool actually is Nessus. Secondly , we also see the X11 in the brackets. This would point to the hackers systems being a unix flavour.

5. Attack mechanism:

The attack is coming from a single IP address. Performing a lookup on the address the following is returned:

```
whois -h whois.arin.net 63.34.212.73
UUNET Technologies, Inc. (NETBLK-NETBLK-UUNET97DU)
   3060 Williams Drive, Suite 601
   Fairfax, va 22031
   US

   Netname: NETBLK-UUNET97DU
   Netblock: 63.0.0.0 - 63.63.255.255
   Maintainer: UUDA

   Coordinator:
      UUNET, Technical Support  (OA12-ARIN)  help@uu.net
      (800) 900-0241
```

From the attack breakdown we can see they are utilising 26 different attacks. Looking at the raw data of four of the different types of attacks we have the following:

1. GET /null.htw?CiWebHitsFile=/default.asp%20&CiRestriction=none&CiHiliteType=Full HTTP/1.1
2. GET /scripts/..%c0%2f..%c0%2f..%c0%2f..%c0%2f../winnt/system32/cmd.exe?/c+dir+c:\+/OG HTTP/1.1
3. GET /cfdocs/expeval/ExprCalc.cfm?OpenFilePath=c: \windows\win.ini HTTP/1.1

Attack mechanism number one targets the webhits.dll that is installed for Microsoft Index server. There is a vulnerability in the dll file that could allow an

attacker to view any file on the server. If an attacker can view any file, then potentially confidential information can be obtained.
Attack mechanism number two attempts to run the cmd.exe command, and passing the dos command dir with the options of /OG which is sort the files and group the directories first. The whole attempt is obfuscated with unicode.
The last attack, number 3, attempts to attempts to exploit the expression evaluator that is included in the Cold Fusion. It was found that an attacker could exploit this and read, delete and create files on the server. Obviously if an attacker can do this, they can totally compromise the target system.
The reason they are attempting these types of attacks, is that they can access these servers on port 80 through the firewall. If one or some of them succeed, they will be able to potentially partially or completely compromise the system depending on which exploit was successful.

6. Correlations:

Given that this is one of the most popular security scanning tools in use today, there will be many detects of this sort.

7. Evidence of active targeting :

From the network detects, it is evident that the attacker is targeting the network a.b.c. This is no evidence of targeting a specific host over any others, this can be seen clearly from the Target breakdown graph 1.0.

8. Severity:

Criticality – Even though the scan was extensive, the servers targeted did not respond in the compromised manner.
Criticality(C) = 2

Lethality – If the attacks had been successful, then one or more servers could have been compromised, to listing, reading, deleting and creating files. Even though the Nessus tool does not aim to bring down or compromise servers but instead test the defences, one a vulnerability had been found, the attacker could then easily have exploited the vulnerable system.
Lethality(L) = 4

System Countermeasures – The counter measures on the targeted system was to have the latest security patches applied and unwanted services removed.
System Countermeasures(SC) = 3.

Network Countermeasures – The router blocks unwanted ports, and firewall only specifically allows traffic required.
Network Countermeasures(NC) = 4.

Severity = (C + L) – (SC + NC)
Severity = (2 + 4) – (3 + 4) = -1

As the severity of this attack is –1, it's a much wait and keep monitoring while keeping in line with the defensive recommendations below with existing and new servers.

9. Defensive recommendation:

As this is such an extensive scanner for vulnerabilities, the following recommendations are put forth.
Patch all systems with the latest security fixes to stop known exploits
Continue to monitor bug track lists and patch as soon as new vulnerabilities are reported
Block any unecessary ports are the firewall
Harden the systems expo sed to the internet, ie - Turn off any services or daemons running on the servers that are not required. Close unused ports on the servers.
Implement further security layers on the critical servers.

10. Multiple choice test question:

Date	Time	Src IP	Src Prt	Dest IP	Dst Prt	Prot	IDS Alert
14/02/2002	02:29	63.34.212.73	51626	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.2 12.73	51682	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.212.73	51683	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:32	63.34.212.73	51690	a.b.c.40	80	6	URL_Data_IIS Unicode Translation v2
14/02/2002	02:34	63.34.212.73	51786	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:35	63.34.212.73	51861	a.b.c.40	80	6	HTTP_TestCgi
14/02/2002	02:35	63.34.212.73	51860	a.b.c.40	80	6	HTTP_TestCgi
14/02/2002	02:35	63.34.212.73	51854	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:35	63.34.212.73	51852	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:35	63.34.212.73	51843	a.b.c.40	80	6	HTTP_WebSite_Uploader
14/02/2002	02:36	63.34.212.73	51946	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:36	63.34.212.73	51947	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:37	63.34.212.73	52023	a.b.c.40	80	6	HTTP_Unix_Passwords
14/02/2002	02:38	63.34.212.73	52111	a.b.c.40	80	6	HTTP_IE_BAT
14/02/2002	02:38	63.34.212.73	52094	a.b.c.40	80	6	HTTP_PHF
14/02/2002	02:39	63.34.212.73	52140	a.b.c.40	80	6	HTTP_NphTestCgi
14/02/2002	02:40	63.34.212.73	52197	a.b.c.40	80	6	HTTP_Netscape_PageServices

Which of the following is MOST LIKELY shown in the trace above?
a)The source IP is spoofed.
b)This attack is being performed manually.
c)This is an automated scan for vulnerabilities
d)There is no server at IP address a.b.c.40

Answer: c

Network Detect 2

These detects all transpired on the 18/03/2002

Time	Actn	Dst Prt	Source IP	Dest. IP	Proto	Da ta
0:06:22	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:24	acpt		z.x.7.107	a.b.40.200	icmp	icmp -type 8 icmp-code 0
0:06:24	acpt		a.b.40.200	z.x.7.107	icmp	
0:06:25	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:28	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:30	acpt	2370	a.b.20.201	z.x.200.1	tcp	
0:06:31	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:31	acpt	2370	a.b.20.201	z.x.200.3	tcp	
0:06:34	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:37	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:40	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:06:43	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:06:46	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:49	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:50	acpt	2370	a.b .20.201	z.x.200.1	tcp	
0:06:51	acpt	2370	a.b.20.201	z.x.200.3	tcp	
0:06:52	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:55	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:58	drop	1721	z.x.2 00.32	a.b.40.200	tcp	unknown established TCP packet
0:07:01	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:07:04	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:07	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:10	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:10	acpt	2370	a.b.20.201	z.x.200.1	tcp	
0:07:11	acpt	2370	a.b.20.201	z.x.200.3	tcp	
0:07:13	drop	1721	z.x.200.32	a .b.40.200	tcp	
0:07:16	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:19	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:07:22	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown establish ed TCP packet
0:07:25	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:28	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:30	acpt	2370	a.b.20.201	z.x.200.1	tcp	
0:07:31	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:31	acpt	2370	a.b.20.201	z.x.200.3	tcp	
0:07:34	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:07:36	acpt		z.x.9.97	a.b.40.200	icmp	icmp -type 8 icmp-code 0
0:07:36	acpt		a.b.40.200	z.x.9.97	icmp	
.						
0:51:01	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:51:04	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:51:07	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:51:10	drop	1721	z.x.20 0.32	a.b.40.200	tcp	
0:51:10	acpt	2370	a.b.20.201	z.x.200.1	tcp	

```
0:51:11  acpt  2370  a.b.20.201     z.x.200.3      tcp
0:51:13  drop  1721  z.x.200.32     a.b.40.200     tcp
```

1. Source of Trace.

This network dete ct originated from where I currently work and manage the NIDS systems.

2. Detect was generated by:

The network detect was generated by exporting logs from Checkpiont Firewall -1 client, then loading into excel and removing unnecessary columns and saving a s a comma separated file, then loading into word, and replacing the commas with spaces. The first two octets of the network address was changed to a.b. The column labeled "Actn" stands for Action. This is what the firewall performed with the packet based on the current firewall rules. In this detect, the action was either a drop or acpt (accept). The heading "Proto" stands for protocol.

3. Probability the source address was spoofed:

An IP lookup of z.x.200.32, z.x.200.1 and z.x.200.1 reveals the foll owing:

whois -h whois.arin.net z.x.200.32

```
US Army Site
   xxxx-xxx-xx
   xxxxxxxxxxx, xxxx
   xxxxx, xx xxxx-xxxx
   US

   Netname: xxxx
   Netblock: z.x.0.0 - z.x.255.255
   Maintainer: DNIC

   Coordinator:
      xxx,xxx

   Domain System inverse mapping provided by:

   NS01.ARMY.MIL 140.153.43.44
   NS02.ARMY.MIL 192.82.113.7
   NS03.ARMY.MIL 130.114.200.6

   Record last updated on 22-Feb-2000.
   Database last updated on  20-Mar-2002 19:58:52 EDT.
```

whois -h whois.arin.net z.x.200.1

```
USAxxx
   xxxx
   xxxxx  xxxx,
```

```
US

Netname: xxxx-xxx
Netblock: z.x.0.0 - z.x.255.255

Coordinator:
   xxxx, xxxx

Domain System inverse mapping provided by:

NS01.ARMY.MIL  140.153.43.44
NS02.ARMY.MIL  192.82.113.7
NS03.ARMY.MIL  130.114.200.6

Record last updated on 07-Mar-2001.
Database last updated on  20-Mar-2002 19:58:52 EDT.
```

As the IP's in concern happen to be military sites, this raises the possibility of source address being spoofed. I emailed the military coordinators and receiv ed a reply that although they owned that address space, the addresses I had specified were not being used internally and that they were not routing them. My conclusion at this stage is that the addresses are spoofed.

Icmp packets list 2.0

0:06:24	z.x.7.107	a.b.40.200	icmp	" type 8 code 0"" "
0:06:24	a.b.40.200	z.x.7.107	icmp	
0:07:36	z.x.9.97	a.b.40.200	icmp	" type 8 code 0"" "
0:07:36	a.b.40.200	z.x.9.97	icmp	
0:08:18	z.x.160.101	a.b.40.200	icmp	" type 8 code 0"" "
0:08:18	a.b.40.200	z.x.160.101	icmp	
0:08:20	z.x.7.107	a.b.40.200	icmp	" type 8 code 0"" "
0:08:20	a.b.40.200	z.x.7.107	icmp	
0:10:20	z.x.7.107	a.b.40.200	icmp	" type 8 code 0"" "
0:10:20	a.b.40.200	z.x.7.107	icmp	
0:11:41	z.x.160.101	a.b.40.200	icmp	" type 8 code 0"" "
0:11:41	a.b.40.200	z.x.160.101	icmp	

Looking at this we can clearly see that z.x.7. 107 is performing a icmp packet type 8, which is echo request, to server a.b.40.200. Then in the same minute, server a.b.40.200 replies with an icmp echo reply packet. On the third line we see z.x.9.97 performing an icmp echo request to server a.b.40.200 , then that server echo replying to z.x.9.97. Then the process is repeated again but with another server z.x.160.101. We know the first two originators of the icmp echo reply are military sites, but upon lookup of z.x.160.101 we find that this address is not a registered one.

The fact that there is one source address not registered, and the other are military sites, which upon face value would either indicate the US military are

pinging an Australian bank or a number of their systems have been compromised, which both not very likely. This then starts to swing the balance in the favour that the addresses are spoofed.

Investigating further, back at the original trace, network detect 2, we see that the non icmp packets are TCP. For TCP communication to wo rk, a 3 way handshake is required. If the address is spoofed, then the 3 way handshake wont complete. This sways the balance back in the direction that the source isn't spoofed.

We have conflicting indicators here whether the source is spoofed or not. Although, currently, the for's for the spoofing seem to outweigh the against's. Further determination of this will come later in the analysis.

4. Description of attack:

Apart from the echo request, there are attempted tcp connections, and some are being dropped because of " unknown established TCP packet ". An attacker may be trying to start the TCP session halfway through without a three way TCP handshake to initialise.

As traffic was also originating on our server a.b.40.200, I decided to investigat e the situation further on the server itself.

Upon performing a netstat –ap (the a flag, shows both listening and non -listening sockets, and p flag shows the associated program that has the port open), I found there were several established connections to the z.x.0.0 network.

Next I performed a netstat –r to review the route table on the Unix server. This revealed a big clue as to what was going on. I found on here that the internal addresses were being routed to the same device as the external addresse s.

Next I looked in the /etc/hosts file and found the key to the whole situation. In the hosts file, I found systems defined on the z.x.0.0 network, but they were not defined as military sites, but internal banking servers. This pointed to the possibility that the packets weren't going out on the internet, but staying internal. If this was true, this also meant the internal IP addresses were not defined in the reserved customary area (10.0.0.0 – 10.255.255.255 or 172.16.0.0 – 172.31.255.255), but actual ly valid external internet IP's. But the fact still remained that the logs were generated by the firewall.

Further follow up revealed that the firewall and the servers are part of a relatively new system and that the firewall itself was actually an int ernal one to segregate internal zones.

The Unix system itself, is running an application which allows load balancing between sites, by finding the closest server to the client. Further information on how this is performed is contained in the attack mecha nism section.

The connection attempts to port 1721 and 2370 were the ports being used by the application on those servers.

Reviewing the network detect 2, there are quite a few dropped packets. After speaking with the people who manage the firewall, it was found that what rules were necessary. Obviously with all the dropped packets, either the applications need to be reconfigured, or the firewall rules changed.

The conclusion at this point is that this is not actually an attack, but we are seeing a special load balancing application and a misconfigured application or firewall.

At this point, we can also return to the question of whether the source address is spoofed or not. Because this is legitimate traffic, for a valid business purpose, the source address is not spoofed.

5. Attack mechanism:

As concluded above, this is not an attack. The applications on the servers were internally developed utilising ports 1721 and 2370. As such the information on the communication mechanism is not available.

In regards to the load balancing system, The Unix system itself, is running an application which allows load balancing between sites. Additionally not only does load balancing occur, but an efficient system is used whereby the closest server to the client is determined and then the session is directed there.

This suitably explains why there are frequent icmp echo request and echo reply packets. Using echo requests with two or more destinations, one can measure which is the fastest path to a server t o work with the client. Once this is determined, the client is directed to run a session on that server.

6. Correlations:

This detect was first detected in the firewall logs on the 18/3/2002 and subsequently found again in the firewall logs four days l ater.

7. Evidence of active targeting:

The systems were definitely being targeted, but for a valid business purpose.

8. Severity:

Criticality – The criticality is low, because no servers are in danger and the system is up and working, although there i s some work to perform to tune the system.
Criticality(C) = 1

Lethality – Negligible, no systems are in danger
Lethality(L) = 0

System Countermeasures – The unix system has the latest security vulnerability patches applied. And only required ports are o pen. Also, security auditing tools are in place.
System Countermeasures(SC) = 4.

Network Countermeasures – The firewall blocks unwanted ports and IP addresses and only allows what should come through, although this may need tuning.
Network Countermeasu res(NC) = 3

Severity = (C + L) – (SC + NC)
Severity = (1 + 0) – (3 + 3) = -5

As the severity of this attack is –5, it's is not of any security concern. Although to improve on further efficiency of system some further work is required.

9. Defensive reco mmendation:

Recommended that the application and the firewall rules be reviewed to determine why they are dropped packets and rectify.

Also, review of the recommendations at the following site http://www.phoneboy.com/faq/0408.html . This FAQ page specifically talks about the Checkpoint firewall-1 error message "unknown established TCP packet" and the reasons why it is received and some possible actions to apply to remedy the situation.

10. Multiple choice test question:

Time	Actn	Dst Prt	Source IP	Dest. IP	Proto	Data
0:06:40	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:06:43	drop	1721	z.x.200.32	a.b. 40.200	tcp	unknown established TCP packet
0:06:46	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:49	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:52	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:55	drop	1721	z.x.200.32	a.b.40.200	tcp	
0:06:58	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet
0:07:01	drop	1721	z.x.200.32	a.b.40.200	tcp	unknown established TCP packet

whois -h whois.arin.net z.x.200.32

```
US Army Site
   xxxx-xxx-xx
   xxxxxxxxxxx, xxxx
```

```
xxxxx, xx xxxx-xxxx
US

Netname: xxxx
Netblock: z.x.0.0 - z.x.255.255
Maintainer: DNIC
```

Given that the a.b network is a commercial bank site, which of the following is MOST LIKELY shown in the trace and lookup above?

a) This is legitimate traffic.
b) The source address is spoofed.
c) The firewall needs to be reconfi gured to allow the traffic.
d) All of the above

Answer: b

Network Detect 3

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.907559 208.1.82.9:22224  -> www.xxx.yyy.2:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.936448 208.1.82.9:22224  -> www.xxx.yyy.4:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x5 97B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.948149 208.1.82.9:22224  -> www.xxx.yyy.5:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack : 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.950141 208.1.82.9:22224  -> www.xxx.yyy.6:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B   Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.976345 208.1.82.9:22224  -> www.xxx.yyy.8:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D  2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/30-14:09:01.991116 208.1.82.9:22224  -> www.xxx.yyy.10:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen:   20
.
.02/01-12:34:42.100938 208.1.82.9:22227  -> www.xxx.yyy.2:22227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
******S* Seq: 0x6345036  Ack: 0x686A6B36  Win: 0xFCA4  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/01-12:34:42.116733 208.1.82.9:22227  -> www.xxx.yyy.4:22227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
******S* Seq: 0x6345036  Ack: 0x686A6B36  Win: 0xFCA4  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/01-12:34:42.143022 208.1.82.9:22227  -> www.xxx.yyy.6:22227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
******S* Seq: 0x6345036  Ack: 0x686A6B36  Win: 0xFCA4  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/01-12:34:42.143351 208.1.82.9: 22227 -> www.xxx.yyy.5:22227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
******S* Seq: 0x6345036  Ack: 0x686A6B36  Win: 0xFCA4  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/01-12:34:42.177385 208.1.82.9:22227  -> www.xxx.yyy.8:22227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
******S* Seq: 0x6345036  Ack: 0x686A6B36  Win: 0xFCA4  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/01-12:34:42.180084 208.1.82.9:22227  -> www.xxx.yyy.10:22 227
TCP TTL:119 TOS:0x0 ID:12147 IpLen:20 DgmLen:40
```

******S* Seq: 0x6345036 Ack: 0x686A6B36 Win: 0xFCA4 TcpLen: 20
=+=

1. Source of Trace.

http://www.incidents.org/archives/intrusions/msg03726.html

2. Detect was generated by:

Snort IDS.

3. Probability the source address was spoofed:

The source address is most likely not spoofed the attacker seems to be performing a search for an application listening on a certain port. To be able to obtain the responses from the destinations that are being checked, the address must not be spoofed.

A lookup on the source IP address reveals the following:
whois -h whois.arin.net 208.1.82.9
Sprint (NETBLK-SPRINTLINK-BLKS) SPRINTLINK-BLKS 208.0.0.0 - 208.35.255.255
City of Ashland (NETBLK-SPRINT-D00150-2) SPRINT-D00150-2
208.1.80.0 - 208.1.83.255

This shows that the address range 208.0.0.0 to 208.35.255.255 which 208.1.82.9 is a part of is a valid owned range. This then does not rule out that the source address is not spoofed.

4. Description of attack:

The first packet in the 'Network Detect 3' attempts to connect from source address 208.1.82.9 to system www.xxx.yyy.2 targeting port 22224. The next is from the same source IP and the packet targets the same destination port but on different server, being www.xxx.yyy.4. This repeats with the last octet advances by small increments, ie .5 .6 .8 .10.

There is a second par t to the network detect, which is two days later with the same characteristics. The source address is the same, and the destinations are the same, but the destination port targeted is now 22227.

The scanner seems to be scanning for a on program on that p ort, most likely a trojan type, listening to port 22224 or to port 22227.

Searching the port database on the Snort web page at http://www.snort.org/ports.html, reveals no known programs associated with the ports. Checking the nmap files nmap -services, this is no entry for either of the ports.

Below is a paste of the timestamps from when the packets were captured on the wire.
Packet 1 01/30-14:09:01.907559

Packet 2 01/30-14:09:01.936448
Packet 3 01/30-14:09:01.948149
Packet 4 01/30-14:09:01.950141
Packet 5 01/30-14:09:01.976345
Packet 6 01/30-14:09:01.991116

Obviously from this, the scan is being performed with the use of an automated tool as the packets are arriving at a rate of about 6 every 10 th of a second.

5. Attack mechanism:

The scan works by sending out a single TCP SYN packet (evidenced by the ******S* part of each packet in the detect) to a target server on port 22224, then repeating this with another server and so on, scanning through a target list of IP addresses. Two days later this process is again repeated but searching for a program on port 22227.

If the scanner receives a SYN ACK reply, then they will know that the port is open and responding. Otherwise if they receive nothing or a RES (Reset) then the port is not accessible or open respectively.

As port 22224 and port 22227 is not a known port for an application or trojan, we can only surmise what the target is. Most likely this is a search for a trojan. Once one is found by the sca nner, this trojan can be used to perform a variety of things, namely the basic functions of list, read, copy and delete files from the trojan'ed system is a makeup of the minimum instruction set of trojans today.

An interesting detail that appears in the detect, is that the TTL (Time to live) for the first set of packets is115 and the TTL for the second set of packets is119. This is the case even though the source address has stayed the same We can learn from this that the first lot of packets took four more hops to reach the same destination than the packets in the second scan two days later. There are a few possible explanations for this.

Either:
1) The path taken for the packets for the first and second scan from the scanner to the victim different . Possibly a new part of internet was 'grown' and the packets were routed more efficiently the second time around. Or that there was maintenance being perform on some routing device during the first scan and the packets had to find an alternative longer path to the target.
2) The tools being used, allows the user to enter a start TTL value. In this scenario, the scanner has chosen a different TTL for each run.
3) The user changed their O/S default values for TTL or installed a new version or different ve rsion of the operating system on their machine, thus changing the default TTL's.
4) Any combination of the above.

Another detail that catches the eye, is that the source port and destination port are the same. The network detect 3 source URL mentions the program Synscan produces this type of signature.

Lastly, the Win: (window) size is different from the first scan to the second. In the first, the window the value is 0x26D2 and in the second scan the window value is 0xFCA4.

6. Correlations:

This detect on these ports seems to be the only one posted. But there a been several other ones posted regarding detects from SynScan / T0rnScan. Here are a couple of incidents URL reporting this:
http://cert.uni-stuttgart.de/archive/incidents/2001/06/msg00206.html .
http://lists.jammed.com/incidents/2001/06/0195.html

7. Evidence of active targeting:

There is clear evidence of active targeting here. Below is listed just the destination hosts for the scans:

First Scan	Second Scan
www.xxx.yyy.2	www.xxx.yyy.2
www.xxx.yyy.4	www.xxx.yyy.4
www.xxx.yyy.5	www.xxx.yyy.5
www.xxx.yyy.6	www.xxx.yyy.6
www.xxx.yyy.8	www.xxx.yyy.8
www.xxx.yyy.10	www.xxx.yyy.10

From this, you can see there was no scan for servers with an IP last address octet of .3, .7 or .9. This indicates most likely that there is no machines at .3, .7 or .9. This picture could have been a built up from a previous scan o f the www.xxx.yyy.0 net. Then the second phase being, scanning known machines for the trojan on port 22224 and 22227.

8. Severity:

Note: As this detect is from a foreign site with little information provided about their network, a number of assumptions have been made below to best determine the severity of the scan.

Criticality – The criticality is low to medium because of the unknown status of a trojan on those two particular ports.
Criticality(C) = 2

Lethality – If there is a trojan on one of the machines in the scan, then this is serious as the scanner will most likely have full control of the machine.
Lethality(L) = 4

System Countermeasures – I'm making the assumption here that the systems on the network at lea st have the standard username and password requirements to stop any particular person accessing the system. As to specialised security software, such as security auditing tools, or host based, this is uncertain and has to be accounted for in the value giv en.
System Countermeasures(SC) = 2 – 4 (average 3).

Network Countermeasures – The assumption here, is that the site has a firewall, and that the firewall is configured to drop everything unless specifically configured to allow. This being the case, this sensor is most likely placed before the firewall to detect these scans. The firewall would thus be stopping the scans from entering the network and stopping the packets from ultimately reaching their goal.

Network Countermeasures(NC) = 4

Severity = (C + L) – (SC + NC)
Severity = (2 + 4) – (3 + 4) = -1

The severity has a value of –1.

9. Defensive recommendation:

Recommendation to check that the firewalls are indeed blocking ports 22224 and 22227. Additionally if the scans for this port keep persisting , then perhaps the scanner has a valid reason for this. It could be that a trojan has successfully been installed on a machine and emailed the scanner of it readiness. The second recommendation then would be to perform an internal scan and determine if there are any systems that are listening on that port. If one is found then investigate the machine in question and remove the trojan. If this is a new trojan, a recommendation of re -installing the operating system and applications is forwarded.

10. Multiple choice test question:

```
01/30-14:09:01.907559 208.1.82.9:22224  -> www.xxx.yyy.2:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
01/30-14:09:01.936448 208.1.82.9:22224  -> www.xxx.yyy.4:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
01/30-14:09:01.948149 208.1.82.9:22224  -> www.xxx.yyy.5:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9  Ack: 0x2957A66B  Win: 0x26D2  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
01/30-14:09:01.950141 2 08.1.82.9:22224  -> www.xxx.yyy.6:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
```

******S* Seq: 0x597B2DC9 Ack: 0x2957A66B Win: 0x26D2 TcpLen: 20
=+=
01/30-14:09:01.976345 208.1.82.9:22224 -> www.xxx.yyy.8:22224
TCP TTL:115 TOS:0x0 ID:15066 IpLen:20 DgmLen:40
******S* Seq: 0x597B2DC9 Ack: 0x2957A66B Win: 0x26D2 TcpLen: 20
=+=

Which field or combination of fields in the above trace, gives the most clue as to which scan tool is being used?

a) The time field.
b) The TOS field.
c) TcpLen field
d) The source port and destination port fields.

Answer: d

Network Detect 4

Date	Time	Action	D.Prt	Source.IP	Prot	S.Prt
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	26943
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	28777
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	25534
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	25518
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	28461
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	25650
25-Jan-02	10:16:07	drop	25000	167.216.183.64	udp	27268
25-Jan-02	10:16:07	drop	25000	167.216.183.64	udp	28288
25-Jan-02	10:16:07	drop	25000	167.216.183.64	udp	25557
25-Jan-02	10:16:16	drop	25000	167.216.183.64	udp	25418
25-Jan-02	10:16:16	drop	25000	167.216.183.64	udp	27261
25-Jan-02	10:16:16	drop	25000	167.216.183.64	udp	26440
25-Jan-02	10:16:24	drop	25000	167.216.183.64	udp	26135
25-Jan-02	10:16:24	drop	25000	167.216.183.64	udp	27996
25-Jan-02	10:16:24	drop	25000	167.216.183.64	udp	25831
.						
25-Jan-02	14:34:32	drop	25000	216.183.194.197	udp	31667
25-Jan-02	14:34:32	drop	25000	216.183.194.197	udp	33322
25-Jan-02	14:34:32	drop	25000	216.183.194. 197	udp	32675
25-Jan-02	14:34:40	drop	25000	216.183.194.197	udp	30560
25-Jan-02	14:34:40	drop	25000	216.183.194.197	udp	29227
25-Jan-02	14:34:40	drop	25000	216.183.194.197	udp	26224

1. Source of Trace.

http://www.incidents.org/archives/intrusions/msg03526.html . The detect was cleaned up the somewhat by removing the unnecessary information from the original post. The columns removed were, firew all event id and whether the firewall logged the event or not. A number of the rows were also removed to shorten the output. Included is enough information to analyse.

2. Detect was generated by:

The detect was generated by Checkpoint Firewall -1. Most likely the fields were selected, by searching only for packets with a destination port of 25000, then performing an export from the GUI firewall -1 client.

The column labelled action identifies what action the firewall took with this packet. In this network detect all the packets were dropped. The column labelled Prot is the Protocol field. D.Prt is the detination port and S.Prt is the source port.

3. Probability the source address was spoofed:

Firstly, the protocol of the packets are all UDP. Because there is no three -way handshake required, then it is possible that the attacker can spoof the IP address and still succeed. Unless, for the attack to succeed the attacker requires information back from the target.

A lookup on the source IP's reveals th e following:

whois -h whois.arin.net 167.216.183.64

Manoa Innovation Center (NET-MIC) MIC 167.216.0.0 - 167.216.255.255
Digital Island, Inc. (NETBLK-MIC-DIGISLE-D) MIC-DIGISLE-D
167.216.176.0 - 167.216.191.255

whois -h whois.arin.net 216.183.194.197

OpNIX, Inc. (NETBLK-OPNIX-1BLK)
2220 W. 14th St.
Tempe, AZ 85281
US

Netname: OPNIX-1BLK
Netblock: 216.183.192.0 - 216.183.223.255
Maintainer: OPNX

Coordinator:
Support, NOC (NS214-ARIN) arin@opnix.net
+1-408-966-2900 (FAX) 4809667551

Domain System inverse mapping provided by:

NS1.OPNIX.COM 216.183.194.131
NS2.OPNIX.COM 216.183.194.132

An nslookup on 216.183.194.197 reveals orb01.phx.opnix.net. Reviewing their web page at www.opnix.com, reveals on their main page " Opnix delivers leading edge IP Traffic Management products to allow efficient use and intelligent scaling of multi-homed IP networks." They have two main products, one is a hardware that measures performance statistics around IP pathways and the other is some software that measures the performance of internet routes. It seems somehow, Opnix has managed to include in the test IP list the person who posted these logs, routers IP addresses. When they perform some internet route statistics, the person's routers are being involved in the testing.

The nslookup, it appears that the addresses are not spoofed given that the source IP explains much of what is being seen.

4. Description of attack:

Referring back to the network detect, the first packet arrives at 10:15 and 52 seconds. This packet originates from 167.216.183.64, and is attempting to be sent to a target host, which is not shown in the detect but is described as one of several routers, on port 25,000. Then two more packets are received at the same time with the same source port and the same destination port. The 4 th, 5th and 6th packet arrive 9 seconds later again with the same source address and destination port. This process repeat, with slightly varying degrees of times between the groups of three packets. All these packets are being dropped by the firewall.

Searching the snort port databases, port 25,000 comes up as: icl-twobase1, secondly the original poster of this detect mentions that cisco products use this port for load balancing.

Removing just the time field fr om each packet from the network detect, we have the following.

From source IP 167.216.183.64

10:15:52 10:15:52 10:15:52
10:16:01 10:16:01 10:16:01
10:16:07 10:16:07 10:16:07
10:16:16 10:16:16 10:16:16
10:16:24 10:16:24 10:16:24 10:16:24 10:16:2 4 10:16:24
10:16:49 10:16:49 10:16:49
10:17:04 10:17:04 10:17:04 10:17:17 10:17:17 10:17:17
10:17:43 10:17:43 10:17:43
10:17:47 10:17:47 10:17:47 10:17:47 10:17:47 10:17:47

From source IP 216.183.194.197

14:34:32 14:34:32 14:34:32
14:34:40 14:34:40 14:34:40
14:34:56 14:34:56 14:34:56
14:36:14 14:36:14 14:36:14
14:36:29 14:36:29 14:36:29
14:37:10 14:37:10 14:37:10
14:37:16 14:37:16 14:37:16
14:37:31 14:37:31 14:37:31
14:37:59 14:37:59 14:37:59
14:38:52 14:38:52 14:38:52

14:38:58 14:38:58 14:38:58 14:38:58 14:38:58 14:38:58 14:38:58 14:38:58
14:38:58 14:38:58 14:38:58 14:38:58 14:38:58 14:38:58 14:38:58

From this we see some interesting timings. Looking firstly at packets from source IP of 167.216.183.64, 3 packets are detected at 10:15:52 and then another three at 10:16:01. This attempt pattern repeats until we get to the time of 10:16:24, where's instead of attempting 3 packets, the attacker sends 6 packets. Then at 10:16:49 back to three packets again. Next time sequence, 6 packets, then 3, then 6 again. The time increments from receiving each group of packets is as follows, 9, 6, 9, 8, 25, 15, 39 and 5 seconds.

On the other hand the packet timings and number of packets from the other source IP address is different. We see 10 attempts of 3 packets starting at 14:34:32 with increments of 8, 16, 18, 15, 41, 6, 15, 28 and 53 seconds. Then at 14:38:58, 6 seconds later, we see 15 packets received.

Because the packets are repeated over a time, and the fact that the first, second and all packets then after are dropped, this is not an attack. Once there is an unsuccessful attempt, the normal pattern of attack would be for the attacker to try a different destination port or a different target IP.

Looking at the source port information, we see that the numbers seem to be all over the place, with large jumps between each one. This would suggest that the attacker is sending many other packets to other destinations also.

Combining the two parts of information above, spurious source ports and time increments, it would indicate the originating system is heavily loaded down with sending many packets out. This nicely lines up with the part described in the "Is the source address spoofed", by the fact, if opnix is performing an internet pathway statistic collection, and it involves fair size network chunks, their system is going to be heavily driven to send packets out to many systems. Thus explaining the varying source port numbers and varying time increments for packets send attempts.

The technology being used that generated these packets is most likely the Orbit 1000 from Opnix.

5. Attack mechanism:

As mentioned above this is not an attack as such, but a technology to try and determine the best and fastest pathway. Naturally the originator of the packets is not intentionally trying to attack the posters routers, but is running performance metrics for the IP pathways.

Further information including white papers can be found on the www.opnix.com web site.

6. Correlations:

There were no other reported incidents of this type.

7. Evidence of active targeting:

As reported in the post from the originator, the target IP addresses all consisted of routers. The specific of the pac kets were for optimising network paths. Therefore this is definitely active targeting. Although in this case, as all the packets are being dropped by the firewall then is a mistaken target.

8. Severity:

Note: As this detect is from a foreign site with little information provided about their network, a number of assumptions have been made below to best determine the severity of the scan.

Criticality – The criticality is low as the analysis reveals the packets to be used for network optimisation.
Criticality(C) = 1

Lethality – Low, although, to properly confirm this, the packet raw data should be captured and determined if the payload is harmful or not.
Lethality(L) = 1

System Countermeasures – I'm making the assumption here that the routers on the network at least have the standard username and password requirements to stop any particular person accessing the system. As to specialised security software, this is uncertain and has to be accounted for in the value given.
System Countermeasures(SC) = 2 – 4 (average 3).

Network Countermeasures – The firewall is currently dropping these packets.
Network Countermeasures(NC) = 4

Severity = (C + L) – (SC + NC)
Severity = (1 + 1) – (3 + 4) = -5

The severity has a value of –5

.9. Defensive recommendation:

There is no immediate security concern. But the following two steps should be taken.

1) Capture the raw data of the packet to confirm that the packets are harmless.
2) Contact the sites and request for them to stop sending the packets to the routers in concern.

10. Multiple choice test question:

Date	Time	Action	D.Prt	Source.IP	Prot	S.Prt
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	26943
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	28777
25-Jan-02	10:15:52	drop	25000	167.216.183.64	udp	25534
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	25518
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	28461
25-Jan-02	10:16:01	drop	25000	167.216.183.64	udp	25650
25-Jan-02	10:16:07	drop	25000	167.216.183.64	udp	27268
25-Jan-02	10:16:07	drop	25000	167.216.183.64	udp	28288

Focusing on the source port numbers, this MOST likely means

a) That the o/s counts backwards when assigning etheral ports.
b) The source machine is working hard and sending out many packets other than the ones seen here.
c) The ports numbers are encrypted messages intended as commands for a trojan on the firewall.
d) The source ports numbers are randomly assigned.

Answer: b

Network Detect 5

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.201317 143.107.105.14:22 -> our.i.p.addr:22
TCP TTL:117 TOS:0x0 ID:23158
**S***** Seq: 0x37255DDE   Ack: 0x58730A6F   Win: 0x4FF1

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.212228 our.i.p.addr:22 -> 143.107.105.14:22
TCP TTL:64 TOS:0x0 ID:8785  DF
**S***A* Seq: 0x75A6691A   Ack: 0x37255DDF   Win: 0x7B88
TCP Options => MSS: 536
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.501353 143.107.105.14:22 -> our.i.p.addr:22
TCP TTL:238 TOS:0x0 ID:59493
****R*** Seq: 0x37255DDF   Ack: 0x0   Win: 0x0

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:51.311351 143.107.105.14:16742 -> our.i.p.addr:22
TCP TTL:47 TOS:0x0 ID:59513
**S***** Seq: 0xEE8B504F   Ack: 0 x0   Win: 0x200
TCP Options => MSS: 1460
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:51.611352 143.107.105.14:16742 -> our.i.p.addr:22
TCP TTL:47 TOS:0x0 ID:59514  DF
******A* Seq: 0xEE8B5050   Ack: 0x75BC90CF   Win:   0x7D78

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:51.921343 143.107.105.14:16742 -> our.i.p.addr:22
TCP TTL:47 TOS:0x0 ID:59520
***F**A* Seq: 0xEE8B5050   Ack: 0x75BC90E8   Win: 0x7D78

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+ =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:52.201358 143.107.105.14:16742 -> our.i.p.addr:22
TCP TTL:47 TOS:0x0 ID:59536  DF
******A* Seq: 0xEE8B5051   Ack: 0x75BC90E9   Win: 0x7D78
```

1. Source of Trace.

http://www.incidents.org/archives/intrusions/msg03249.html

2. Detect was generated by:

Snort intrusion detection system.

3. Probability the source address was spoofed:

All the packets in the trace are TCP. Of course to establish an TCP connection a three was handshake is required. In the network detect, there is no completed 3 way TCP handshake, from two SYN attempts. This would immediately suggest that the source address is being spoofed.

If we look at the first three packets solely, we see the first is from a SYN from 143.107.105.14 from port 22 to out.i.p.addr, destination port 22. The next packet is a replying SYN ACK from the server our.i.p.addr. Then the third is a RST from 143.107.105.14. If we assume that the first packet is spoofed, then the second is a reply to the spoofed address, and the third is a response from the surprised real machine at IP address 143.107.105.14 saying it didn't know about a SYN it was meant to have sent, and so sends back a RST, this fits well with what we are seeing. Further evidence to back this up, is the 1st packet has a TTL of 117, and 2nd packet supposedly from the same IP address has a different TTL of 238. This would indicate that the two packets came from a different machine.

Looking at packets 4, 5, 6 and 7, we see that they are all from the source IP of 143.107.105.14, without reply from the targeted server ou.i.p.addr. Normally their should be a answering communication from the targeted server. If we assume that the source address is again spoofed for these packets, then the attacker would not know what responses the targeted server has sent back (in this case none) and try and guess what the responses are and then continue on the TCP conversation that way. This assumption fits the packets we are seeing.

4. Description of attack:

The attack is TCP against port 22. This is an attack against ssh. The attack is broken into two phases

Phase 1) The attacker first spoofs the address and sends a SYN, the targets responds to the spoofed address with a SYN ACK. The real machine at the source address responds with a RST.

Phase 2) The attacker sends 4 more packets, TCP SYN, ACK, FIN ACK and ACK respectively all from the same source port of 16742.

5. Attack mechanism:

Phase 1 of the attack is most likely there for the attacker to see a sample of the current sequence number being used by the target so that they can then predict the next sequence number the victim will use.

As there is no further reset response from the legitimate source machine, the attacker probably performs a DOS on the machine at IP address 143.107.105.14.

Phase 2 of the attack, the hacker uses a specific program that attempts to exploits a vulnerability in ssh. This tool sets the TTL to some value, which when the packets arrive at the target, the TTL value is 47. This explains why the TTL values from the attacker in

the first packet are different from the TTL values in the packets from the attacker in packets 4,5,6 and 7.

As the attacker is spoofing the sour ce address, they are not aware that there is no response from the target of a SYN ACK. They therefore continue on with the third part of the 3 way handshake with an ACK. At this stage if the hacker was attempting to exploit the ssh vulnerability, like a buffer overflow, then one would expect the 6 th packet to be a PSH with data in the payload, not a FIN ACK and the next packet of ACK. It seems that phase two of the attack is really only testing out the TCP state table of the target or experimenting at th is stage to see what is possible.

6. Correlations:

There are no other traces of this nature that could be found. Although there are some exploits available for ssh. One of these http://www.dig italoffense.net/openssh_zlib/ .

7. Evidence of active targeting:

All the packets point to the fact that there is only one target involved in all this. Therefore this is definitely active targeting.

8. Severity:

Note: As this detect is from a fore ign site with little information provided about their network, a number of assumptions have been made below to best determine the severity of the scan.

Criticality – The criticality is medium as the packets so that the attacker is using some degree of effort in trying to attack the system. This is evidenced by the fact, that the network detect indicates they performed a DOS on the spoofed address as part of the attack on the target. This requires some level of technical knowledge.
Criticality(C) = 4

Lethality – Medium to High, although, to properly confirm this, the packet raw data should be captured and determined if the payload is harmful or not.
Lethality(L) = 4

System Countermeasures – The target system has ssh in place which indicates a heigh tened security mechanism in place. Although, in the network detect, the target does not respond in phase 2 of the attack. Further system counter measures would be appropriate.
System Countermeasures(SC) = 3

Network Countermeasures – I am assuming that there is a firewall in place, or why else would the attacker go to all the trouble of spoofing the source address. This plus the fact that there is also a Snort IDS system in place rates fairly well.

Network Countermeasures(NC) = 4

Severity = (C + L) – (SC + NC)
Severity = (4 + 4) – (3 + 4) = 1

The severity is 1.

9. Defensive recommendation:

Recommend to implement a host based IDS on the target machine and keep further close watch on the NIDS logs. Also, notifying the spoofed source address that they were probably DOS'd.

10. Multiple choice test question:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.201317 143.107.105.14:22  -> our.i.p.addr:22
TCP TTL:117 TOS:0x0 ID:23158
**S***** Seq: 0x37255DDE   Ack: 0x58730A6F   Win: 0x4FF  1

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.212228 our.i.p.addr:22  -> 143.107.105.14:22
TCP TTL:64 TOS:0x0 ID:8785  DF
**S***A* Seq: 0x75A6691A   Ack: 0x37255DDF   Win: 0x7B88
TCP Options => MSS: 536
--
=+=+=+=+=+=+=+=+=+=+=+=+ =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-23:11:49.501353 143.107.105.14:22  -> our.i.p.addr:22
TCP TTL:238 TOS:0x0 ID:59493
****R*** Seq: 0x37255DDF   Ack: 0x0   Win: 0x0

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

Which of the following is the MOST LIKELY, based on the Snort log extract above?

a) The source address is spoofed in the first packet.
b) The source address is spoofed in the third packet.
c) The source address is not spoofed in any of the packets.

Answer: a

Assignment 3 - "Analyze This" Scenario

Executive summary

It is evident from the log files that many of the University computers have been compromised by a computer worm and that hacking activity is rampant on the internal network, originating from outside sources as well as internal.

A number of changes need to be made, to clean up the problems and put up appropriate security defences to effective deal with the everyday hacking attempts. These recommended changes can be found in the 'Defensive Recommendations' section.

It is strongly advised that these recommendations be taken up and applied as soon as possible.

List of files that were analysed.

Five consecutive days worth of Network Intrusion Detection data was analysed from the data posted at http://www.research.umbc.edu/~andy . From these five days, three types of files were reviewed. These were scans, alerts and OOS (Out of Spec) data files. Listed below are the actual files reviewed.

Alerts	OOS	Scans
alert_020119.txt	oos_Jan_19_2002.txt	scans_020119.txt
alert_020120.txt	oos_Jan_20_2002.txt	scans_020120.txt
alert_020121.txt	oos_Jan_21_2002.txt	scans_020121.txt
alert_020122.txt	oos_Jan_22_2002.txt	scans_020122.txt
alert_020123.txt	oos_Jan_23_2002.txt	scans_020123.txt

List of detects prioritized by number of occurrences

To produce a list of detects prioritized by number of occurrences I used the tool called SnortSnarf. This is sourced from the Silicon Defense web site with the following URL http://www.silicondefense.com/software/snortsnarf/index.htm .

Before SnortSnarf would be able to process the files properly, the Unix editor vi was used to changed all occurrences of MY.NET to 10.1. For the re st of this report, MY.NET will be referred to as 10.1 network. After SnortSnarf produced it's output, the text was dumped from the html page and transferred into Microsoft Excel for re-sorting the most common occurrence to the top.

Signatures	#Alrts	#Srcs	#Dsts
connect to 515 from inside	25135	73	1
SNMP public access	21781	13	138
MISC Large UDP Packet	17653	15	7
spp_http_decode: IIS Unicode attack detected	13023	83	267
ICMP traceroute	10819	2	2
Watchlist 000220 IL -ISDNNET-990517	5125	35	7

INFO MSN IM Chat data	3870	64	66
High port 65535 udp - possible Red Worm - traffic	1554	50	103
ICMP Echo Request CyberKit 2.2 Windows	1479	2	4
ICMP Router Selection	1001	126	1
ICMP Fragment Reassembly Time Exceeded	949	14	25
Null scan!	871	44	6
ICMP Echo Request Windows	784	9	15
spp_http_decode: CGI Null Byte attack detected	658	8	11
SMB Name Wildcard	462	51	47
FTP DoS ftpd globbing	322	5	3
ICMP Echo Request BSDtype	314	2	3
SYN-FIN scan!	296	3	293
ICMP Echo Request Nmap or HPING2	243	8	14
ICMP Echo Request L3retriever Ping	234	18	6
WEB-MISC Attempt to execute cmd	148	13	6
ICMP Destination Unreachable (Host Unreachable)	112	1	35
ICMP Destination Unreachable (Comm Admin Prohibited)	110	1	10
NMAP TCP ping!	99	13	5
INFO FTP anonymous FTP	96	9	20
WEB-IIS view source via translate header	91	4	2
INFO Possible IRC Access	78	8	7
Incomplete Packet Fragments Discarded	74	6	3
EXPLOIT NTPDX buffer overflow	58	10	5
INFO Inbound GNUTella Connect request	56	50	1
MISC traceroute	53	3	3
ICMP Destination Unreachable (Protocol Unreachable)	48	1	1
WEB-CGI scriptalias access	29	1	1
SCAN Synscan Portscan ID	29	28	1
SCAN Proxy attempt	28	6	6
INFO Inbound GNUTella Connect accept	20	2	19
WEB-IIS _vti_inf access	19	10	2
Possible trojan server activity	18	3	3
WEB-FRONTPAGE _vti_rpc access	18	10	2
EXPLOIT x86 setuid 0	14	6	7
Queso fingerprint	13	6	2
SCAN FIN	13	5	2
Prt 55850 tcp -Possible myserver activity -ref. 010313 -1	12	4	4
WEB-MISC 403 Forbidden	9	2	5
IDS552/web -iis_IIS ISAPI Ovrflw ida nosize [arachNIDS]	8	7	6
INFO - Possible Squid Scan	6	2	4
MISC source port 53 to <1024	5	5	2
INFO Outbound GNUTella Connect accept	5	5	2
Attempted Sun RPC hi gh port access	5	3	4
WEB-MISC compaq nsight directory traversal	5	4	4
SUNRPC highport access!	5	1	1
High port 65535 tcp - possible Red Worm - traffic	4	3	3
EXPLOIT x86 NOOP	3	3	3
ICMP Source Quench	3	1	1
Back Orifice	2	2	2
MISC PCAnywhere Startup	2	1	2
EXPLOIT x86 setgid 0	2	2	2
ICMP Echo Request Cisco Type.x	2	1	1
Probable NMAP fingerprint attempt	2	1	1
INFO Napster Client Data	1	1	1

Signatures continued	#Alrts	#Srcs	#Dsts
WEB-CGI redirect access - Internal UDP	1	1	1
connection to external tftp server	1	1	1
TCP SRC and DST outside network	1	1	1
RFB - Possible WinVNC -010708-1 Port 55850 udp -	1	1	1
myserver activity - ref.010313-1	1	1	1
WEB-IIS Unauthorized IP Access Attempt	1	1	1
WEB-MISC http directory traversal	1	1	1

Explanation of findings

connect to 515 from inside – NIDS has detected connection attempts to port 515, from the internal network. 515 is defined as lpd spooler port. As there are over 25,000 packets of this type , this is highly suspicious and points to a compromise and or an attack of some sort.

SNMP public access – A person is attempting to use SNMP (Simple Network Management Protocol) to take control of SNMP enabled devices on the internal network. The time scale of when these alerts came, was about the same time as a major SNMP vulnerability was announced. This activity is probably a part of the attack/defend syndrome as soon as any new vulnerability is announced. The NIDS is seeing the attacks.

MISC Large UDP Packet – The nids system has detected a abnormally large UDP packet, data size greater than 4000 bytes. This could be an attempt to perform a buffer overflow exploit

spp_http_decode: IIS Unicode attack detected – A person externally is attempting to attack the web servers utilising Unicode. This is a common form of web attack.

ICMP traceroute – A person externally is trying to perform a traceroute to a device in your network, potentially trying to map the internal network.

Watchlist 000220 IL -ISDNNET-990517 – As the watch list is not included, this alert, specifically looks for triggers that the Nids user has setup.

INFO MSN IM Chat data – The NIDS systems has detected chat data. This indicates that someone is on a chat session. This is potent ially dangerous as hackers have exploited chat systems numerous times.

High port 65535 udp - possible Red Worm – traffic has been detected on the utmost port number and it possibly signifies that one or more systems has been infected with the red worm.

ICMP Echo Request CyberKit 2.2 Windows - Someone externally is using the Windows tool CyberKit to try and ping a server on the internal.

ICMP Router Selection – An external person is trying to use the ICMP protocol to try and select a router. This could b e for the intention to divert communications.

ICMP Fragment Reassembly Time Exceeded – An ICMP message has arrived externally to the internal network, notifying that the reassembly time limit to reassemble some packets has exceeded. This could be an indi cator that someone internally is trying to compromise a system outside by using packet fragmentation.

Null scan! – this is a common occurring stealth scan.

ICMP Echo Request Windows – An external windows machine is trying performing an echo request on a n internal target.

spp_http_decode: CGI Null Byte attack detected – An attacker is attempting to exploit a vulnerability with CGI script by passing a null byte.

SMB Name Wildcard - An attacker is trying to trying to discover information about a machine on the internal network. This is targeted at the web servers. The following page refers to an increase in this type of activity: http://www.sans.org/newlook/resources/IDFAQ/port_13_7.htm .

FTP DoS ftpd globbing – An attacker is trying to exploit a vulnerability in the ftp daemon. This vulnerability allows a remote attacker to run code of their choosing on the target server.

ICMP Echo Request BSDtype – This is a echo request originating from a Unix BSD type from the outside

SYN-FIN scan! – This is a form of stealth scan. An attacker is probably trying to map the internal network.

ICMP Echo Request Nmap or HPING2 – Either nmap or HPING2 is attempting to scan the intern al network.

ICMP Echo Request L3retriever Ping – Someone from the external network is using a tool called L3retriever to attempt to ping the internal network.

WEB-MISC Attempt to execute cmd – An external person from the outside is attempting to run a sy stem command through the URL, on an internal target.

ICMP Destination Unreachable (Host Unreachable) – An attempt has been made to connect to a server that does not exist. This is evidence of mapping the internal network.

NMAP TCP ping! – Someone is performing an nmap ping of an internal system. This is serious, because the person is trying to map your internal network.

INFO FTP anonymous FTP – A person externally is attempting to log on anonymously to an ftp server in the internal network.

WEB-IIS view source via translate header – An external party is attempting to view the source on the web servers via a known vulnerability.

INFO Possible IRC Access – The NIDS has detected possible IRC activity. This is a concern if true, as many systems have bee n compromised in the past through IRC, and also viruses can enter through chat channels.

Incomplete Packet Fragments Discarded – Packet fragments were detected, but there was no completion of all the fragments to re -assemble them. This could mean either a hacker is attempting to penetrate the defences or the network may be misconfigured.

EXPLOIT NTPDX buffer overflow – An attempt to trigger the NTPDX buffer overflow exploit has been detected.

INFO Inbound GNUTella Connect request – An inbound GNUTella connection has been detected. As GNUTella shares out system files from the local machine to the internet, confidential information could be disclosed.

MISC traceroute – An attempted traceroute was performed on one of the internal systems. This can be a concern, as an attacker can use this information to map the internal network.

WEB-CGI scriptalias access – an external person is attempting to exploit ScriptAlias function. If successful the attacker will be able to read the source code of the CGI programs.

SCAN Synscan Portscan ID – a person is performing a port scan using SYN packets of the internal network.

INFO Inbound GNUTella Connect accept – The request from the internal network to connect to a GNUTella machine has been accepted. This indicates that someone is sharing files with the internet.

Possible trojan server activity – the NIDS system suspects trojan activity is taking place.

EXPLOIT x86 setuid 0 - An attempt to exploit a Unix system has been detected. The exploit involves setting the userid to 0, which is root access.

Queso fingerprint – Queso is a scanning tool that can performing operating system identification. This identification process has been detected and alerted on.

SCAN FIN – A scan is being performed you FIN packets.

Prt 55850 tcp-Possible myserver activity-ref. 010313-1 – This is a distributed denial of service attack tool. Which once a system has been compromised through an exploit, then sets itself up to listen to further commands on port 55850.

Prt 55850 tcp-Possible myserver activity-ref. 010313-1 - This is an alert that shows that some kind of activity is being performed. As there is no indication as what ref.010313-1 means, the meaning behind this cannot be determined.

WEB-MISC 403 Forbidden – It has been detected that a HTTP 403 forbidden message has been sent. This means someone has tried to access an object on one of the web servers and was not allowed.

IDS552/web-iis_IIS ISAPI Ovrflw ida nosize [arachNIDS] – A vulnerability is being attempted on one of the IIS web servers.

INFO - Possible Squid Scan – A detection has been made that indicates likely use of the scanner called Squid. Someone may be trying to map the internal network.

MISC source port 53 to <1024 – An attempt to connect to a reserved port fro m an external machine, who's source port is 52. This is suspect activity as the normal operation of behaviour is usually, high port (>1023) to low port (<1024). Port 53 is DNS, so most likely there is an attacker trying to attack the internal network with a DNS exploit.

INFO Outbound GNUTella Connect accept - An outbound GNUTella connection has been detected. As GNUTella shares out system files from the local machine to the internet, confidential information could be disclosed.

Attempted Sun RPC high p ort access – an external person is attempting to connect to a Sun RPC high port. This is a concern because this is a common target for hackers.

WEB-MISC compaq nsight directory traversal – an attacker is trying to perform a directory traversal and exploi t the nsight vulnerability on one of the web servers.

SUNRPC highport access! – this indicates a possible compromise or one step away from a compromise of the system.

High port 65535 tcp - possible Red Worm – traffic – Port 65535 is the upper most TCP port number that can be used. The alert is reporting that this could be red worm traffic.

EXPLOIT x86 NOOP – An attacker is trying to exploit the x86 NOOP vulnerability.

ICMP Source Quench – An ICMP source quench has been detected. An attacker can repeat edly send source quenches to attempt an effective DOS, or it could be a sign of network problems.

Back Orifice – Packets have been detected that indicate they are packets for Back Orifice.

MISC PCAnywhere Startup – A PCAnywhere startup string has been de tected. If this is successful, it is a concern, as an external party is controlling an internal system.

EXPLOIT x86 setgid 0 – An attempt to exploit a Unix system has been detected. The exploit involves setting the group id to a priviledged 0 value.

Probable NMAP fingerprint attempt – It is likely that nmap is being used to try and identify what operating system a device is on the internal network. This is

dangerous, as once determined, an attacker can then specifically through those particular exploit s at the system.

INFO Napster Client Data – Traffic has been detected that indicates that Napster packets are on the network.

WEB-CGI redirect access - Internal UDP

connection to external tftp server – Someone internally is connecting to an external tftp server. Today, tftp is rarely used and the most likely reason is that someone is trying to hack a system.

TCP SRC and DST outside network – a packet has been detected that has the source and destination IP outside the internal network. This should ne ver happen and points to packet crafting.

myserver activity - ref.010313-1 – This is an alert that shows that some kind of activity is being performed. As there is no indication as what ref.010313 -1 means, the meaning behind this cannot be determined.

WEB-IIS Unauthorized IP Access Attempt – An external person has attempted to connect to one of the http servers, when there IP address is not in the allowed connections list.

WEB-MISC http directory traversal – An attacker has attempted to go outside the normal boundaries of the web directories by using a directory traversal technique. This is a sign of attempted access.

Top Talkers List

To produce the top 10 talkers the a number of ways was used.

Alerts

For the alert files, the five files wer e concatenated together into one large file using the Unix cat command. Then SnortSnarf was used with the following command line parameter

./snortsnarf.pl –top=10

This controlled SnortSnarf to output only the top 10 talkers instead of the default top 20.

Top 10 talkers from Alerts

Rank	Total # Alerts	Source IP	# Signatures triggered	Destinations involved
rank #1	10997 alerts	10.1.70.177	2 signatures	(25 destination IPs)
rank #2	10416 alerts	10.1.5.202	1 signatures	10.1.1.1
rank #3	4208 alerts	63.210.47.81	2 signatures	10.1.153.245
rank #4	4209 alerts	10.1.88.240	1 signatures	10.1.153.195
rank #5	3520 alerts	64.124.157.48	1 signatures	10.1.153.45
rank #6	3152 alerts	212.179.35.118	1 signatures	(3 destination IPs)
rank #7	2958 alerts	64.124.157.58	1 signatures	10.1.153.196, 10.1.153.45
rank #8	2804 alerts	10.1.153.114	4 signatures	(21 destination IPs)
rank #9	2731 alerts	10.1.153.210	3 signatures	(27 destination IPs)
rank #10	2558 alerts	10.1.150.198	1 signatures	(100 destination IPs)

The criteria used to determine the top 10 was to keep a running tally of source IP addresses. The highest tally is at rank number 1, the 10 th highest tally, is at rank number 10.

Scans

For the scan files, first the f ive files were concatenated together into one large file using Unix cat command. Then the file was imported into an Microsoft Access database and the unnecessary fields removed. Then the database was imported into Seagate's Crystal Reports and setup to p roduce totals of each source address and sorted in descending order.

Top 10 talkers from scans

Rank	No. packets	Source IP	Source Port
1	337052	10.1.60.43	123
2	87230	10.1.6.45	7000
3	29751	10.1.6.49	7000
4	27849	10.1.6.52	7000
5	20229	10.1.6.48	7000
6	15436	10.1.6.60	7000
7	15030	10.1.6.50	7000
8	13148	10.1.6.53	7000
9	12173	205.188.228.33	31652
10	9879	10.1.153.145	7001

Out of Band (OOB) packets.

For the scan files, first the five files were concatenated together into one large file using Unix cat command. Then the file was imported into an Microsoft Access database and the unnecessary fields removed. Then the database was imported into Seagate's Crystal Reports and setup to produce totals of each source address and sorted in descending order.

Top 10 talkers from out of band (OOB) traffic

Rank	Source IP	Number of packets
1	68.52.151.18	292
2	24.180.218.241	11
3	210.54.160.143	5
4	210.20.23.57	3
5	24.234.240.101	3
6	24.73.8.140	3
7	129.81.155.31	2
8	24.234.248.101	2
9	12.246.128.81	1
10	134.76.25.226	1

Internet Registration Information

The following five external internet addresses were picked to be further investigated by finding out their internet registration information. The reasons why these are picked are also discusse d.

63.210.47.81 – This IP was chosen as it rated rank three, highest external in the alerts section
64.124.157.48 - This IP was chosen as it rated rank number five, 2nd highest external, in the alerts section
216.106.172.147 – This source was chosen becau se from the link graph in the next section, the machine is sending packets on 65,535, which is one of the main alerts, and is also performing some attempts at exploiting some targets in the internal network.
205.188.228.33 – This ranked as the highest exte rnal source in the top 10 scans
68.52.151.18 – This machine was the rank number one top talk in the OOB traffic, attempting to perform a SYN FIN of the internal network.

---

whois -h whois.arin.net 63.210.47.81
Level 3 Communications, Inc. (NETBLK-LEVEL4-CIDR)
1450 Infinite Drive
Louisville, CO 80027
US

Netname: LEVEL4-CIDR
Netblock: 63.208.0.0 - 63.215.255.255
Maintainer: LVLT

Coordinator:
level Communications (LC-ORG-ARIN) ipaddressing@level3.com
+1 (877) 453-8353

Domain System inverse mapping provided by:

NS1.LEVEL3.NET 209.244.0.1
NS2.LEVEL3.NET 209.244.0.2

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 30-May-2001.
Database last updated on 25-Mar-2002 20:08:41 EDT.

---

whois -h whois.arin.net 64.124.157.48
Abovenet Communications, Inc. (NETBLK-ABOVENET)
50 W. San Fernando Street, Suite 1010
San Jose, CA 95113
US

Netname: ABOVENET
Netblock: 64.124.0.0 - 64.125.255.255
Maintainer: ABVE

Coordinator:
Metromedia Fiber Networks/AboveNet (NOC41-ORG-ARIN) noc@ABOVE.NET
408-367-6666
Fax- 408-367-6688

Domain System inverse mapping provided by:

NS.ABOVE.NET 207.126.96.162
NS3.ABOVE.NET 207.126.105.146

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 27-Apr-2001.
Database last updated on 25-Mar-2002 20:08:41 EDT.

---

whois -h whois.arin.net 216.106.172.147
iBEAM Broadcasting Corporation (NETBLK-IBEAM)
645 Almanor Ave., suite 100
Sunnyvale, CA 94085
US

Netname: IBEAM
Netblock: 216.106.160.0 - 216.106.175.255
Maintainer: BEAM

Coordinator:
Le, Stewart (SL895-ARIN) stle@ibeam.com
408-830-3572

Domain System inverse mapping provided by:

NS1.IBEAM.COM 204.233.70.15
NS2.IBEAM.COM 204.247.99.125

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 22-Jan-2002.
Database last updated on 25-Mar-2002 20:08:41 EDT.

---

whois -h whois.arin.net 205.188.228.33

America Online, Inc (NETBLK-AOL-DTC)
22080 Pacific Blvd
Sterling, VA 20166
US

Netname: AOL-DTC
Netblock: 205.188.0.0 - 205.188.255.255

Coordinator:
America Online, Inc. (AOL-NOC-ARIN) domains@AOL.NET
703-265-4670

Domain System inverse mapping provided by:

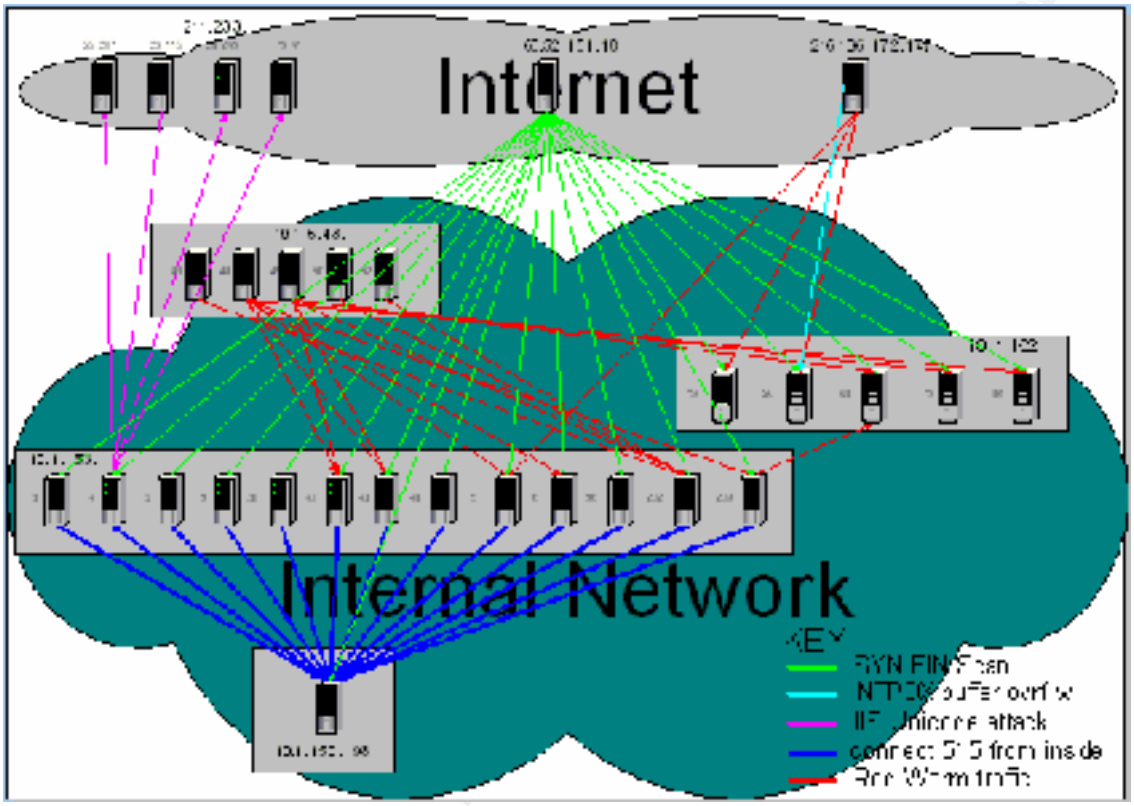
DNS-01.NS.AOL.COM 152.163.159.232
DNS-02.NS.AOL.COM 205.188.157.232

Record last updated on 27-Apr-1998.
Database last updated on 25-Mar-2002 20:08:41 EDT.

whois -h whois.arin.net 68.52.151.18
Comcast Cable Communications, Inc. (NETBLK-JUMPSTART-1) JUMPSTART-1
68.32.0.0 - 68.63.255.255
Comcast Cable Communications, Inc. (NETBLK-JUMPSTART-NASHVILLE)
JUMPSTART-NASHVILLE
68.52.0.0 - 68.53.175.255

Link Graph

The following link graph was produced by looking through the data, with Microsoft Access database, Crystal reports and SnortSnarf. Then the diagram was construc ted using Visio Professional.



Five different types of traffic are shown above. The red lines are 'Red Worm packets'. The dark blue lines at "connect to 515 from inside". The pink lines are "IIS Unicode attack". The light blue lines are "NTPDX bu ffer overflow" and the green lines are "SYN FIN Scans".

Analysis

If we look at the red lines in the link graph, we see that there is a lot of this type of traffic going around the network. We have some coming from the internet and targeting three internal servers. And internally, the there is traffic passing backwards and forwards amongst several computer. Keeping in mind this is just a small selection of the traffic. So the bigger picture, would be much more widespread.

The following URL has details on the worm
http://www.simovits.com/trojans/tr_data/y49.html

And the following description of the worm has been copied here

"The worm searches for known vulnerabilities in wu-ftpd, BIND, LPR and rpc.statd. If any of them are found, the worm hacks the Linux system and becomes root. It also mails information to one of four Chinese addresses."

Looking at the link graph, but this time at the dark blue line "connect to 515 from inside" we see a number of servers targeting one server. The thicker blue indicates that many packets are being sent from each host. Port 515 is the lpd spooler port. Referring back to the short description pasted above, we see one of the ways the worm propagates is through exploiting a known vulnerability of LPR. From this we can conclude that all the servers showing dark blue lines to the one server have been compromised by the 'Red Worm'. Secondly, any servers we red lines destined or originating from them, are a strong possibility for infection also by the 'Red Worm'.

Taking a look at the pink line, this shows a single machine, performing a IIS Unicode attack on a number of server out on the internet. This clearly shows that the attacks are coming from that one server. Either the attacker has a legitimate account and is using that to attack the systems out on the internet, or they have compromised that system and are using that for a base of attack.

The light blue line shows an attempt from the internet to exploit the NTPDX buffer overflow on one of the internal servers.

And lastly, the light green line, shows the SYN FIN scan attempt from a server out on the internet to a number of servers internally.

Referring back to the top 10 talkers for alerts and scans, we see that for alerts, 6 out of the top 10 are internal and for scans, 9 of the 10 are internal. This is of serious concern, because it reveals that attacks are originating internally. This either means there are legitimate people using the systems to hack other systems, or that the internal systems have been compromised from the outside and the internal systems are being used as spring boards to compromise other systems or a combination of both of these scenarios.

Correlations

In the following practicals, there is correlating evidence of the'Red Worm' in the university internal network.

http://www.giac.org/practical/Jeff_Holland_GCIA.doc
http://www.giac.org/practical/Marc_Kneppers_GCIA.zip
http://www.giac.org/practical/Chris_Payne_GCIA.zip

Defensive recommendation

The first recommendation would be to treat any computer that had traffic to or from it on port 65535 as compromised. These systems should be disconnected from the network immediately and the worm removed.

All servers, this consisting of infected and non-infected alike, should have the latest security patches applied to them, especially regarding the patches on the services that the worm uses to infect. These being wu-ftpd, bind, lpr and rpc.statd.

Serious consideration should be given to blocking the methods the worm uses to propagate at the firewall. Any of the services strictly not necessary to and from the internet should all be blocked.

Once the worm has been removed from all university computers, a final scan should take place to double check that all the systems are clean.

Additionally because of the wide spread of the infection, all other servers should be checked for the worm and any that are found should be cleaned.

A process should be put in place to ensure that new security patches are constantly and regularly applied in a timely manner to the University computers. If this was in place initially, it's most likely the worm would not have been successful in penetrating the internal network.

A number of IRC and other chat systems were detected in the logs. Block these chat systems at the firewall. These chat systems are notorious for being hacked and are frequent targets. This should not be allowed.

Naptster and GNUTella connection in and out of the University was detected. Block Napster and GNUTella ports at the firewall. File sharing out to the internet is an easy backdoor for hackers to exploit.

There is quite a high degree of hacker activity originating from the University computers. Part of this will be made up from students performing the hacking, and the other part will be made up of external people compromising the University computers and then using them to further explore and compromise.

To handle the first contingent of people, a clear University policy should be provided to all students informing them that hacking is not permitted and the consequences if they do. The NIDS log files should be reviewed on a regular basis to catch any hackers in the timely manner and the consequences carried out promptly. This should serve as a deterrent to further would be hackers.

To handle the threat of hacking from the outside, there are a number of steps that need to be taken, the first already recommended is to have an effective vulnerability management process to ensure security patches are applied in a timely manner. This will stop most of the system compromises. Additional to this, the firewall rules should be reviewed, and unnecessary services should be blocked. A policy of only allow traffic to originate from the inside to the outside and not vice versa is a good start. Any rules that can be further tightened, ie replace the ALL field for source and destination with specific addresses or nets is far more preferable.

Analysis process

My first step was to perform the top 10 talkers list. This really ga ve me a quick picture of where the main attacks were coming from. The next step is to produce the link graph. This revealed a wealth of information, that quite clearly showed what traffic was going where and then revealed why that type of traffic was bei ng seen.

At this point, it's was necessary to start looking in detail at how certain attacks worked. For example, finding out the Red Worm attacks to four different exploits, and that one of the exploits was the number one alert found in the logs.

Anoth er helpful technique I used, was to use SnortSnarf, to perform a continuous drill down. For example, clicking on one of the exploits Summary section, takes you to a list of all the sources and destinations involved in that scan. Then you can then click on any of the sources or destinations, which brings you to a html page that shows all the alerts associated with this IP. This gave me an feel for what was going on in the network in regards to the alerts.

Lastly, following a trail was important. For ex ample, from the link graph it was obvious that attacks were being launched from a particular IP address. Using SnortSnarf again clicking on this IP then showed me all the attempts to and from this box. This showed potential other systems that which were being targeted and also, which system may have attacked the hacked this system itself. One can then follow the trail back.

List of References

Protor, Paul E. Practical Intrusion Detection Handbook . Upper Saddle River, 2001.

The Honeynet Project. Know your enemy . Indianapolis: Addison-Wesley, October 2001.

Allen, Julia. The CERT Guide to System and Network Security Practices . Upper Saddle River, Addison-Wesley, May 2001.