GIAC Certified Intrusion Analyst (GCIA)
Practical Assignment
Version 3.3

Carl Madzelan
SANS 2003 San Diego participant

Intrusion Detection

Note: Typographical conventions used in this practical assignment are:

12 point Arial is used for standard text.

10 point Arial is used for command line output; anything entered into a terminal console; log entries for all events; any results returned from any network registrar. It is also used for endnote documentation containing references to URLs of key interest in the practical assignment.

SUMMARY

This practical assignment consists of three parts. The first section is a discussion of the snmpXdmid exploit, and details surrounding the capture of the exploit utilizing an intrusion detection system. The second section consists of three network detects with each detect dissected into 10 items of detail. Finally in the last section, five days of a particular University's logs are analyzed to formulate a set of defensive recommendations.

DESCRIBE THE STATE OF INTRUSION DETECTION

An acceptable approach to illustrating the current state of intrusion detection for the practical assignment is to highlight an attack using an available exploit. The intention is to highlight the capture of the exploit with an intrusion detection product and discuss the results. I hope to impart upon the reader the severity of the exploit, and methods by which the reader may mitigate the risk of a successful attack. Without adequate measures to prevent a system from becoming compromised and a subsequent launching pad for further intrusion, the threat to hosts connected to an unsecured network is very real.

The intrusion detection product that was chosen is Snort. Snort when properly tuned to the network conditions serves as an excellent choice for capturing a wide variety of malevolent and anomalous behavior. Unless the IDS product is being utilized to provide monitoring in as close to real time as possible, a compromised system becomes a launching pad for further malicious behavior. Recently concepts such as Intrusion Prevention Systems[1] have been introduced as a component of a total defensive strategy for infrastructure. But it is suggested to the reader that if some best practices were implemented, attacks would have less of a chance at success. The IDS alert and response mechanisms could be tailored for maximum efficiency. The best defense against attack is a strong foundation. Always operate from a position of strength.

The exploit I have chosen to discuss is the Sun Solaris snmpXdmid Format String Vulnerability detailed in CVE Name CAN-2001-0236[2]. The source code for the exploit was found at http://lsd-pl.net/code/SOLARIS/solsparc_snmpxdmid.c According to the Entercept Ricochet Advisory[3] the Solaris SNMP to DMI mapper daemon vulnerability was first published in 2001, and is categorized as a remotely exploitable buffer overflow. A remotely exploitable buffer overflow exploit involves the passing of more data to a program's storage area than is allocated or expected. Often the 'spilling over' of this information overwrites data beyond that what is allocated for the storage area. Programming practices usually insure error checking logic exists to guarantee the size of the buffer is acceptable for the data it will receive. Sometimes this logic is faulty or does not exist in an application. Individuals who craft these exploits have the necessary programming experience to be keenly aware of these potential 'overflows', and therefore will attempt to exploit these weakness. If the exploitable buffer exists in a privileged process, the host may then execute the data that 'overflowed' as if it were a program[4]. This situation allows the programmer of the exploit to elevate their privilege level on a machine of which they may not be privileged to utilize at all. Additional discussion concerning buffer overflows is available from a variety of sources and a complete discussion of best programming practices is beyond the scope of this discussion[5].

In order to demonstrate the Solaris snmpXdmid exploit a Sun Ultra 5 was prepared with Solaris 8 issue 10/2000. I installed the default developer package set to simulate the package set of interest to most desktop workstation end users. The Solaris 8 10/2000 host provides information listed below in the messages file after booting. This information provides current OS release and patch level for the kernel. In this case it is running 64 bit SunOS 5.8 release at kernel patch level 03.

```
Jun 27 19:35:48 localhost genunix: [ID 540533 kern.notice] SunOS Release 5.8 Version
Generic_108528-03 64-bit
Jun 27 19:35:48 localhost genunix: [ID 784649 kern.notice] Copyright 1983-2000 Sun
Microsystems, Inc.  All rights reserved.
```

To illustrate the number of open ports after a default installation readily visible to external hosts, I utilized nmap to produce this information. The network topology of my testing lab was closed, that is, only these two machines were connected together via a hub utilizing a private IP address range. Utilization of nmap for information gathering on a public network is not recommended as this activity is often viewed as aggressive, and a precursor to an attack. Nmap by Fyodor is an excellent tool available at http://www.insecure.org/nmap/, but should be used with caution. By scanning my own machine, I am able to generate a (truncated below) listing of open ports from the default install:

```
Script started on Wed 25 Jun 2003 09:39:05 PM EST
root@localhost:/ [root@localhost /]# nmap -sT -vv -O 192.168.0.102
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Host  (192.168.0.102) appears to be up ... good.
Initiating Connect() Scan against  (192.168.0.102)
The Connect() Scan took 4 seconds to scan 1601 ports.
```

```
For OSScan assuming that port 7 is open and port 1 is closed and neither are firewalled
Interesting ports on  (192.168.0.102):
(The 1570 ports scanned but not shown below are in state: closed)
Port       State       Service
7/tcp      open        echo
9/tcp      open        discard
13/tcp     open        daytime
19/tcp     open        chargen
21/tcp     open        ftp
23/tcp     open        telnet
25/tcp     open        smtp
37/tcp     open        time
79/tcp     open        finger
111/tcp    open        sunrpc
512/tcp    open        exec
513/tcp    open        login
514/tcp    open        shell
515/tcp    open        printer
540/tcp    open        uucp
587/tcp    open        submission
898/tcp    open        unknown
4045/tcp   open        lockd
6000/tcp   open        X11
6112/tcp   open        dtspc
7100/tcp   open        font-service

Remote operating system guess: Solaris 8 early access beta through actual release
Uptime 0.018 days (since Wed Jun 25 21:14:17 2003)
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=58620 (Worthy challenge)
TCP ISN Seq. Numbers: 169434C6 16977C37 1698EE3D 169C288B 169D4596 169FC81A
IPID Sequence Generation: Incremental
Nmap run completed -- 1 IP address (1 host up) scanned in 10 seconds
```

There are many ports viewable by an external host on this Solaris machine. The danger inherent with this configuration is that the machine is not suitable for exposure to the Internet. Despite best efforts at education, many end users on campus complete an OS installation on the Internet 'wide open'. Often many will not employ some combination device with built-in NAT technology or a firewall hardware solution when building their Sun workstation. On campus it has been noted before that Solaris machines have been rooted/had binaries replaced (netstat/ps/ls) while an individual was preparing the machine.

A mitigation strategy for or all new Solaris OS installations would be to have a preconfigured flash archive (flar) installed on a workstation or server via a private network utilizing Jumpstart technology[6]. It is possible to create a set of profiles related to types of configurations which meet the needs of users for workstations, and server class machines. Sun also offers a set of scripts known as JASS[7] which can be customized to disable many of the default services. These scripts from the vendor are customizable and provide the flexibility to tighten the default configuration. Fewer services listening provide fewer paths for exploitation. Note the machine to be exploited has not been patched. Following vendor patch recommendations[8] is another component of best practices. The same exploit did not function on a patched host.

This Ultra 5 is by default running daemons which rely upon the SNMP and DMI remote management protocols:

```
# ps -elf | grep snmp
 8 S     root   261     1  0  44 20       ?    265       ? 19:53:25 ?       0:00 /usr/lib/snmp/snmpdx -y -c
/etc/snm
 8 S     root   272     1  0  43 20       ?    452       ? 19:53:27 ?       0:00 /usr/lib/dmi/snmpXdmid -s
localhost
```

The purpose of Simple Network Management Protocol is to assist with management of all types of equipment over the network. The Desktop Management Interface is also a management technology. But, both DMI and SNMP do not automatically interoperate.

The Desktop Management Interface (DMI) is the product of a task force that was formed in 1992 to deliver a common framework to manage desktop systems[9]. A component of DMI is the DMI service provider which is charged with handling messages related to desktop information. The "management" and "component"' interfaces to the service provider exchange messages called "events" or "indications". The term "event" describes a runtime condition being responded to by a DMI "component". The term "indication" refers to the notification message sent by an "event generator" to alert the service provider that an event occurred. The service provider simply coordinates requests between applications and components. The DMI Service Provider then passes the "indication" along to any remote "user" of this information. An example of a user would be an application management tool[10], not an actual physical user. It is this snmpXdmid daemon which may experience a buffer overflow while handling an 'indication' over RPC service 100249.

Sun implemented this SNMP to DMI functionality specifically in the snmpXdmid mapper daemon. Sun's reasoning for this was to facilitate communication with SNMP management applications. An SNMP management application may send requests to snmpXdmid which then turns SNMP requests into DMI requests. This enables the SNMP management application to participate in active management of DMI-enabled components[11]. The buffer overflow exploit can be triggered by the event *DmiComponentAdded* which is a function that the DMI service provider utilizes to signify that a component has been added[12]. Overall, this is another example where ease of use and flexibility engineering are perceived as assets but if enabled prove to be a liability.

To capture the exploit I chose the intrusion detection product Snort 2.0.0. Snort was configured with the default rule set enabled on a Dell PIII 1GHz Optiplex. The Dell had RedHat Linux version 2.4.20-9 installed. Several packages were downloaded from their respective sources and compiled to complete my combination IDS and analysis workstation. Those of most importance are httpd-2.0.46, php-4.3.1, mysql-4.0.12, ACID 0.9.6, the ADODB library for PHP4, libpcap-0.7.2, and Tcpdump-3.7.2. There are several procedures available from the Internet that detail the appropriate steps required to set up an intrusion detection system and an analysis console. A google.com search will reveal many sites with further information; each highlighting different methods for the installation (one chooses RPM packages or downloading source and compiling).

In order to demonstrate the exploit in action, the chosen topology was a private network consisting of the two hosts connected by a hub. I compiled the code on the Linux host and executed the exploit against the Solaris host on the network. The Solaris host's IP address is 192.168.0.102 (target) and the Linux machine (source) host's IP address is 192.168.0.101.

Below we see the execution of the compiled exploit is simple. By passing a '–v 8' parameter to indicate the version of Solaris to exploit and the target host IP address, I observed the output of the Solaris "uname –a" information displayed on my console. This proves the exploits success:

```
[root@localhost /]#./solsparc_snmpxdmid.o 192.168.0.102 -v 8
copyright LAST STAGE OF DELIRIUM mar 2001 poland  //lsd-pl.net/
snmpXdmid for solaris 2.7 2.8 sparc

adr=0x000e69c0 timeout=10 port=633 connected! sent!
SunOS localhost 5.8 Generic_108528-03 sun4u sparc SUNW,Ultra-5_10

Script done on Fri 27 Jun 2003 08:03:06 PM EST
```

The code to execute the command (proving contact) is written to the socket that was created:

```
<SNIP>
   stat=clnt_call(cl,SNMPXDMID_ADDCOMPONENT,xdr_req,&req,xdr_void,NULL,tm);
   if(stat==RPC_SUCCESS) {printf("\nerror: not vulnerable\n");exit(-1);}
   printf("sent!\n");
   write(sck,"/bin/uname -a\n",14);
<SNIP>
```

On the Solaris machine we see the effects of the exploit from this truncated process listing. UID 0 – root.

```
# ps –elf | more
 F S      UID   PID  PPID  C PRI NI     ADDR     SZ    WCHAN    STIME TTY      TIME CMD

8 S     root   412   411  0  41 20        ?     38        ? 19:54:56 pts/5    0:00 sh -i

# pfiles 412
412:                    sh -i
  Current rlimit: 256 file descriptors
   0: S_IFCHR mode:0620 dev:136,0 ino:34098 uid:0 gid:7 rdev:24,5
      O_RDWR
```

```
script done on Fri Jun 27 20:00:55 2003
```

On the Linux machine Snort produced the following output to an alert file:

```
[**] [1:569:5] RPC snmpXdmi overflow attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
06/27-20:01:01.446616 192.168.0.101:633 -> 192.168.0.102:32776
TCP TTL:64 TOS:0x0 ID:14375 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x82F56FF0  Ack: 0x568B924  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 42883018 35420
[Xref => http://www.cert.org/advisories/CA-2001-05.html][Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0236][Xref => http://www.securityfocus.com/bid/2417]
```

The snort.conf was changed to allow logging to the ACID/MySQL database:

```
output database: log, mysql, user=root password=gosnort dbname=snort host=localhost
```

I executed snort using the command line switches –c to read in my configuration file and –l log to the /tmp directory and –v verbosely show the IP and TCP/UDP/ICMP headers:

```
./snort  -c ../rules/snort.conf -l /tmp  -v
```

The snort rule which matched this attack pattern was:

```
rpc.rules:alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"RPC snmpXdmi overflow attempt"; flow:to_server,established; content:"|0000 0f9c|"; offset:0; depth:4; content:"|00018799|"; offset: 16; depth:4; reference:bugtraq,2417; reference:cve,CAN-2001-0236; reference:url,www.cert.org/advisories/CA-2001-05.html; classtype:attempted-admin; sid:569; rev:5;)
```

I re-ran the exploit and submit the truncated network trace of the attack as follows. We see the initial conversation begin as the exploiting machine connects to the Sun and receives from RPC the port on which the snmpXdmid daemon is listening – 32861. The exploit begins from port 957 on the attacker to 32861.

```
22:31:17.645187 192.168.0.101.956 > 192.168.0.102.sunrpc: S [tcp sum ok]
478420956:478420956(0) win 5840 <mss 1460,sackOK,timestamp 1933663 0,nop,wscale 0>
(DF) (ttl 64, id 40579, len 60)
0x0000  4500 003c 9e83 4000 4006 1a1d c0a8 0065      E..<..@.@......e
0x0010  c0a8 0066 03bc 006f 1c84 1fdc 0000 0000      ...f...o........
0x0020  a002 16d0 ed13 0000 0204 05b4 0402 080a      ................
0x0030  001d 815f 0000 0000 0103 0300          ..._........
22:31:17.645468 192.168.0.102.sunrpc > 192.168.0.101.956: S [tcp sum ok]
2837613610:2837613610(0) ack 478420957 win 24616 <nop,nop,timestamp 1159328
1933663,nop,wscale 0,nop,nop,sackOK,mss 1460> (DF) (ttl 64, id 45902, len 64)
0x0000  4500 0040 b34e 4000 4006 054e c0a8 0066      E..@.N@.@..N...f
0x0010  c0a8 0065 006f 03bc a922 8c2a 1c84 1fdd      ...e.o...".*....
0x0020  b012 6028 aba5 0000 0101 080a 0011 b0a0      ..`(............
0x0030  001d 815f 0103 0300 0101 0402 0204 05b4      ..._............
22:31:17.645528 192.168.0.101.956 > 192.168.0.102.sunrpc: . [tcp sum ok] ack 1 win 5840
<nop,nop,timestamp 1933663 1159328> (DF) (ttl 64, id 40580, len 52)
```

```
0x0000  4500 0034 9e84 4000 4006 1a24 c0a8 0065      E..4..@.@..$...e
0x0010  c0a8 0066 03bc 006f 1c84 1fdd a922 8c2b      ...f...o.....".+
0x0020  8010 16d0 35c9 0000 0101 080a 001d 815f      ....5.........._
0x0030  0011 b0a0                                    ....
22:31:17.657233 192.168.0.101.956 > 192.168.0.102.sunrpc: P [tcp sum ok] 1:61(60) ack 1 win
5840 <nop,nop,timestamp 1933664 1159328> (DF) (ttl 64, id 40581, len 112)
0x0000  4500 0070 9e85 4000 4006 19e7 c0a8 0065      E..p..@.@......e
0x0010  c0a8 0066 03bc 006f 1c84 1fdd a922 8c2b      ...f...o.....".+
0x0020  8018 16d0 80bf 0000 0101 080a 001d 8160      ...............`
0x0030  0011 b0a0 8000 0038 343b f206 0000 0000      .......84;......
0x0040  0000 0002 0001 86a0 0000 0002 0000 0003      ................
0x0050  0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0060  0001 8799 0000 0001 0000 0006 0000 0000      ................
22:31:17.657390 192.168.0.102.sunrpc > 192.168.0.101.956: . [tcp sum ok] ack 61 win 24616
<nop,nop,timestamp 1159330 1933664> (DF) (ttl 64, id 45903, len 52)
0x0000  4500 0034 b34f 4000 4006 0559 c0a8 0066      E..4.O@.@..Y...f
0x0010  c0a8 0065 006f 03bc a922 8c2b 1c84 2019      ...e.o...".+....
0x0020  8010 6028 ec31 0000 0101 080a 0011 b0a2      ..`(.1..........
0x0030  001d 8160                                    ...`
22:31:17.657837 192.168.0.102.sunrpc > 192.168.0.101.956: P [tcp sum ok] 1:33(32) ack 61 win
24616 <nop,nop,timestamp 1159330 1933664> (DF) (ttl 64, id 45904, len 84)
0x0000  4500 0054 b350 4000 4006 0538 c0a8 0066      E..T.P@.@..8...f
0x0010  c0a8 0065 006f 03bc a922 8c2b 1c84 2019      ...e.o...".+....
0x0020  8018 6028 c54c 0000 0101 080a 0011 b0a2      ..`(.L..........
0x0030  001d 8160 8000 001c 343b f206 0000 0001      ...`....4;......
0x0040  0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0050  0000 805d                                    ...]
```

The exploit attempts begin:

```
22:31:17.658523 192.168.0.102.32861 > 192.168.0.101.957: S [tcp sum ok]
2837745856:2837745856(0) ack 493933456 win 24616 <nop,nop,timestamp 1159330
1933664,nop,wscale 0,nop,nop,sackOK,mss 1460> (DF) (ttl 64, id 45906, len 64)
0x0000  4500 0040 b352 4000 4006 054a c0a8 0066      E..@.R@.@..J...f
0x0010  c0a8 0065 805d 03bd a924 90c0 1d70 d390      ...e.]...$...p..
0x0020  b012 6028 727b 0000 0101 080a 0011 b0a2      ..`(r{..........
0x0030  001d 8160 0103 0300 0101 0402 0204 05b4      ...`............
22:31:17.658589 192.168.0.101.957 > 192.168.0.102.32861: . [tcp sum ok] ack 1 win 5840
<nop,nop,timestamp 1933664 1159330> (DF) (ttl 64, id 21232, len 52)
0x0000  4500 0034 52f0 4000 4006 65b8 c0a8 0065      E..4R.@.@.e....e
0x0010  c0a8 0066 03bd 805d 1d70 d390 a924 90c1      ...f...].p...$..
0x0020  8010 16d0 fc9e 0000 0101 080a 001d 8160      ...............`
0x0030  0011 b0a2                                    ....
22:31:17.658525 192.168.0.102.sunrpc > 192.168.0.101.956: F [tcp sum ok] 33:33(0) ack 62 win
24616 <nop,nop,timestamp 1159330 1933664> (DF) (ttl 64, id 45907, len 52)
0x0000  4500 0034 b353 4000 4006 0555 c0a8 0066      E..4.S@.@..U...f
0x0010  c0a8 0065 006f 03bc a922 8c4b 1c84 201a      ...e.o...".K....
0x0020  8011 6028 ec0f 0000 0101 080a 0011 b0a2      ..`(............
0x0030  001d 8160                                    ...`
22:31:17.658627 192.168.0.101.956 > 192.168.0.102.sunrpc: . [tcp sum ok] ack 34 win 5840
<nop,nop,timestamp 1933664 1159330> (DF) (ttl 64, id 0, len 52)
0x0000  4500 0034 0000 4000 4006 b8a8 c0a8 0065      E..4..@.@......e
0x0010  c0a8 0066 03bc 006f 1c84 201a a922 8c4c      ...f...o.....".L
0x0020  8010 16d0 3568 0000 0101 080a 001d 8160      ....5h.........`
0x0030  0011 b0a2                                    ....
22:31:17.661653 192.168.0.101.957 > 192.168.0.102.32861: . [tcp sum ok] 1:1449(1448) ack 1
win 5840 <nop,nop,timestamp 1933665 1159330> (DF) (ttl 64, id 21233, len 1500)
0x0000  4500 05dc 52f1 4000 4006 600f c0a8 0065        E...R.@.@.`....e
0x0010  c0a8 0066 03bd 805d 1d70 d390 a924 90c1        ...f...].p...$..
0x0020  8010 16d0 45c6 0000 0101 080a 001d 8161        ....E..........a
0x0030  0011 b0a2 0000 0f9c 4446 da78 0000 0000        ........DF.x....
0x0040  0000 0002 0001 8799 0000 0001 0000 0101        ................
0x0050  0000 0001 0000 0020 3f0f 8105 0000 0009        ........?.......
0x0060  6c6f 6361 6c68 6f73 7400 0000 0000 0000        localhost.......
0x0070  0000 0000 0000 0000 0000 0000 0000 0000        ................
0x0080  0000 0000 0000 0000 0000 0001 0000 0000        ................
0x0090  0000 0001 0000 0644 0000 0000 0000 000d        .......D........
0x00a0  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
0x00b0  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
0x00c0  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
0x00d0  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
0x00e0  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
<SNIP>
22:31:17.661707 192.168.0.101.957 > 192.168.0.102.32861: P [tcp sum ok] 2897:4001(1104)
ack 1 win 5840 <nop,nop,timestamp 1933665 1159330> (DF) (ttl 64, id 21235, len 1156)
0x0000  4500 0484 52f3 4000 4006 6165 c0a8 0065        E...R.@.@.ae...e
0x0010  c0a8 0066 03bd 805d 1d70 dee0 a924 90c1        ...f...].p...$..
0x0020  8018 16d0 dc84 0000 0101 080a 001d 8161        ...............a
0x0030  0011 b0a2 ffff ffc0 0000 0000 0000 000d        ................
0x0040  0000 006f ffff ffc0 0000 0000 0000 000d        ...o............
```

Later followed by the sending of the command to execute:

```
22:31:27.719903 192.168.0.101.957 > 192.168.0.102.32861: P [tcp sum ok] 519065:519079(14)
ack 1 win 5840 <nop,nop,timestamp 1934671 1159345> (DF) (ttl 64, id 21600, len 66)
0x0000  4500 0042 5460 4000 4006 643a c0a8 0065        E..BT`@.@.d:...e
0x0010  c0a8 0066 03bd 805d 1d78 bf28 a924 90c1        ...f...].x.(.$..
0x0020  8018 16d0 e7a6 0000 0101 080a 001d 854f        ..............O
0x0030  0011 b0b1 2f62 696e 2f75 6e61 6d65 202d        ..../bin/uname.-
0x0040  610a                               a.
```

And the response in return from the Sun:

```
22:31:27.727238 192.168.0.102.32861 > 192.168.0.101.957: P [tcp sum ok] 1:67(66) ack 519079
win 24616 <nop,nop,timestamp 1160336 1934671> (DF) (ttl 64, id 45958, len 118)
0x0000  4500 0076 b386 4000 4006 04e0 c0a8 0066        E..v..@.@......f
0x0010  c0a8 0065 805d 03bd a924 90c1 1d78 bf36        ...e.]...$...x.6
0x0020  8018 6028 9dd9 0000 0101 080a 0011 b490        ..`(............
0x0030  001d 854f 5375 6e4f 5320 6c6f 6361 6c68        ...OSunOS.localh
0x0040  6f73 7420 352e 3820 4765 6e65 7269 635f        ost.5.8.Generic_
0x0050  3130 3835 3238 2d30 3320 7375 6e34 7520        108528-03.sun4u.
0x0060  7370 6172 6320 5355 4e57 2c55 6c74 7261        sparc.SUNW,Ultra
0x0070  2d35 5f31 300a                               -5_10.
```

If a user possesses root privileges then complete control is available to that user. Although the command “uname” was executed on the target machine, the true purpose of this exploit is not to gain knowledge of which version of the Solaris operating system is currently running. The purpose is to gain control of the system for other potentially negative reasons not limited to: harvesting information for further exploitation, creation of a repository for exchange of copyrighted or illicit materials, or possibly creating a zombie machine for future denial of service attacks. As noted from http://www2.fedcirc.gov/advisories/FA-2001-05.html affected sites have reported discovering the following things on compromised systems:

> Evidence of extensive scanning for RPC services (port 111/{udp,tcp}) with explicit requests for the snmpXdmid service port prior to the exploit attempt. A core file from snmpXdmid on the / partition.* An additional copy of inetd running (possibly using /tmp/bob as a configuration file). A root-privileged telnet backdoor installed and listening on port 2766 (although any port could be used). An SSH backdoor installed and listening on port 47018 (although any port could be used). An IRC proxy installed as /var/lp/lpacct/lpacct and listening on port 6668. A sniffer installed as /usr/lib/lpset

It is of paramount importance to stress to end users that if there are services they will not utilize, they need to be disabled. It is not often that groups of Solaris machines are installed simultaneously across campus, but the simplicity of this attack and its effectiveness should serve as a warning. The source code compiled without issue and ran perfectly. Often these NAT/firewall devices are available relatively inexpensively. End users should check to see what tools vendors provide to help mitigate the chances of a security event occurring.

The Ultra 5 should have been prepared behind a NAT box with scripts from the Security Solaris Security Toolkit (JASS) to harden the default configuration, and configured with a host based firewall such as IPfilter[13], or utilized Wietse Venema's TCP Wrappers, and had been patched with Sun’s interactive PatchPro™ patch management application before making its appearance on the Internet. Filtering rules on border routers, firewall appliances to secure systems on specific VLANs, and implementation of secure communication protocols (Ssh/SSL/VPNs) are all components of campus wide best practices which may lessen the chance of security events occurring.

It is the responsibility of all to educate faculty, staff and students of the dangers lurking on the Internet. If the measures taken above were the norm, then the practice of intrusion detection on a large campus network would be much easier because a strong foundation would have already been established.

PART TWO: NETWORK DETECTS

DETECT NUMBER ONE - SCAN PROXY (8080) ATTEMPT

Posted To: intrusions@incidents.org Mon, 14 Jul 2003 22:41:48 -0500
Received a question from John Ruiz <flippedman@yahoo.com> Tue, 15 Jul 2003 01:47:02 -0700 (PDT)
Requested additional comments Wed, 23 Jul 2003 08:24:54 -0500.

1. Source of Trace:

This trace was obtained from http://www.incidents.org/logs/Raw. The raw data from which this trace originated was 2002.5.15. The first timestamp in the file is '19:00' and the last is '18:57'. I assume that this is a 24 hour period of capture collected on May 15, 2002 as noted by the file name. Printing the link-level header (tcpdump –e) on each dump line reveals two MAC addresses on each of the 1404 entries in the tcpdump capture file: 0:0:c:4:b2:33 and 0:3:e3:d9:26:c0.

```
[root@localhost logs]# /usr/sbin/tcpdump -n -e -r 2002.5.15 | grep -L "0:3:e3:d9:26:c0" | wc -l
     0
[root@localhost logs]# /usr/sbin/tcpdump -n -e -r 2002.5.15 | grep -L "0:0:c:4:b2:33" | wc -l
     0
```

Searching http://standards.ieee.org/regauth/oui/oui.txt reveals both CISCO SYSTEMS, INC. for 00000C and 0003E3. I assume that the IDS was on a segment between two Cisco devices. Google searches for these MAC vendor identifications seem to indicate that one device is a router/firewall (00:00:0C) and the other is a router product (00:03:E3). It appears that destination IP's within the internal network are from an IANA reserved netblock of 46.0.0.0 – 46.255.255.255.

Total packets for this time period were 1404:

```
#/usr/sbin/tcpdump  -vvn -r 2002.5.15 | wc -l
1404
```

About 77% of the alert packets were protocol http flowing from the internal network:

```
#/usr/sbin/tcpdump  -vvn -r 2002.5.15 src net 46.5.0.0/16 and ether src 0:0:c:4:b2:33 | wc -l
1076
#/usr/sbin/tcpdump  -vvn -r 2002.5.15 src net 46.5.0.0/16  | awk {'print $4'} | awk -F. {'print $5'} |
uniq
http:
```

About 22% of the alert packets were flowing in from the external device.

```
# /usr/sbin/tcpdump  -vvn -r 2002.5.15 dst net 46.5.0.0/16 and ether src 0:3:e3:d9:26:c0 | wc -l
328
```

2. Detect was generated by:

Files 2002.5.10 to 2002.5.26 were analyzed with Snort 2.0.0 Build 72 configured with the default rule set enabled on a Dell PIII 1GHz Optiplex running RedHat Linux version 2.4.20-9. Several packages were downloaded from their respective sources and compiled to complete my combination IDS and analysis workstation. Those of most importance were httpd-2.0.46, php-4.3.1, mysql-4.0.12, ACID 0.9.6, the ADODB library for PHP4, libpcap-0.7.2, and Tcpdump-3.7.2.

Raw files were replayed into the database to generate alerts using the following command:

```
#  for list in `ls  /u01/snort/logs/2002*`; do ./snort -c /u01/snort/rules/snort.conf -r  $list -l
/u01/sans_caps; done
```

The alert generated in the ‘alert’ log file was:

```
[**] [1:620:2] SCAN Proxy (8080) attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/15-18:34:43.634488 194.108.153.205:4609 -> 46.5.182.131:8080
TCP TTL:106 TOS:0x0 ID:50829 IpLen:20 DgmLen:48 DF
******S* Seq: 0x5CC14FFF  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

The pattern matching rule that generate this output was from scan.rules:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 8080 (msg:"SCAN Proxy \(8080\) attempt";
flags:S; classtype:attempted-recon; sid:620; rev:2;)
```

Reference : http://www.snort.org/snort-db/sid.html?sid=620

The command used to extract the packets was:

```
[root@localhost logs]# /usr/sbin/tcpdump -Xx -vvn -e -r 2002.5.15  dst port 8080 and host
46.5.182.131
18:34:43.634488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S [bad tcp cksum f9f9!] 1556172799:1556172799(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50829, len 48, bad cksum f7e!)
0x0000   4500 0030 c68d 4000 6a06 0f7e c26c 99cd        E..0..@.j..~.l..
0x0010   2e05 b683 1201 1f90 5cc1 4fff 0000 0000        ........\.O.....
0x0020   7002 4000 2a11 0000 0204 05b4 0101 0402        p.@.*...........
18:34:46.864488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S [bad tcp cksum f9f9!] 1556172799:1556172799(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50878, len 48, bad cksum f4d!)
0x0000   4500 0030 c6be 4000 6a06 0f4d c26c 99cd        E..0..@.j..M.l..
0x0010   2e05 b683 1201 1f90 5cc1 4fff 0000 0000        ........\.O.....
0x0020   7002 4000 2a11 0000 0204 05b4 0101 0402        p.@.*...........
18:34:53.434488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S [bad tcp cksum f9f9!] 1556172799:1556172799(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50991, len 48, bad cksum edc!)
0x0000   4500 0030 c72f 4000 6a06 0edc c26c 99cd        E..0./@.j....l..
0x0010   2e05 b683 1201 1f90 5cc1 4fff 0000 0000        ........\.O.....
0x0020   7002 4000 2a11 0000 0204 05b4 0101 0402        p.@.*...........
```

3. Probability the source address was spoofed:

I conclude that is unlikely the source address was spoofed. Nslookup reveals:

```
Non-authoritative answer:
205.153.108.194.in-addr.arpa    canonical name = 205.192/27.153.108.194.in-addr.arpa.
205.192/27.153.108.194.in-addr.arpa     name = ftp.intertech.cz.

Authoritative answers can be found from:
192/27.153.108.194.in-addr.arpa nameserver = ns.intertech.cz.
ns.intertech.cz internet address = 194.108.153.201
```

It appears to be related to a company named “DATA Intertech s.r.o. - DATA Group member”. From RIPE it appears to have contact valid data (truncated here): address: DATA Intertech, s.r.o. address: The Czech Republic. Since this is a probe, it requires the completion of the TCP three-way handshake. Given the FQDN and P0f information, the information suggests that the source machine is really generating the network stimuli observed.

4. Description of attack:

I do not believe this is an actual attack, but a reconnaissance attempt. The same source IP sent 3 SYN packets to the same destination IP on port 3128 looking for an answer from a Squid proxy.

```
[root@localhost logs]# /usr/sbin/tcpdump  -vvn -e -r 2002.5.15  dst port 3128 and host
46.5.182.131
18:34:18.004488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4501 >
46.5.182.131.squid: S [bad tcp cksum f9f9!] 1545057514:1545057514(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50346, len 48, bad cksum 1161!)
18:34:21.274488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4501 >
46.5.182.131.squid: S [bad tcp cksum f9f9!] 1545057514:1545057514(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50394, len 48, bad cksum 1131!)
18:34:27.834488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4501 >
46.5.182.131.squid: S [bad tcp cksum f9f9!] 1545057514:1545057514(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) (ttl 106, id 50511, len 48, bad cksum 10bc!)
```

Proxies are used for positive and negative purposes. Some of the more nefarious uses of a proxy are sending mass unsolicited email, masquerading as a machine from another network thereby providing anonymity to the attacker, and other illicit behaviors. If there was a proxy service on port 8080, this attempt at information gathering could lead to an attack.

The probing host could be a Windows 2000 type (p0f.fp passive OS fingerprinting[14]) machine possibly functioning as an ftp server. It’s logically named ftp.intertech.nz thereby designating its use. SYN packets have a TCP window size of 16384; we see “sackOK” along with the flag “nop” set along with the DF flag. The incorrect TCP checksum of 0x2a11 versus 0x240b with all packets is most likely an artifact of the sans.org packet obfuscation process. Interestingly, there is a Windows 2000 server[15] Microsoft Knowledge Base Article discussing packets generated with wrong checksums.

5. Attack mechanism:

This is not an attack, but a reconnaissance attempt. The host in this scan at IP 194.108.153.205 is attempting to complete the 3 way handshake to establish a connection to the host with the destination (obfuscated) IP of 46.5.182.13. The initial set of 3 SYNs is an attempt to establish a connection. If the destination host was listening for a connection on port 8080, the destination host would normally respond with a SYN-ACK returning a new sequence number and an acknowledgement sequence number reflecting the initial sequence number from the source host. The conversation would continue with the ACK in return, setting the stage for the exchange of information. Since there is no SYN-ACK returned, the source can not proceed and create a socket (IP + port = socket), and subsequently read and write to that socket descriptor returned. What transpired in this network trace was the equivalent of throwing a ball against a wall and it did not bounce back, but disappeared (hopefully on the ground). Interesting to note below is that all three packets had the same initial sequence number. This packet may be crafted.

```
[root@localhost logs]# /usr/sbin/tcpdump -vv -xX -n -e -r 2002.5.15  host 194.108.153.205 and
host 46.5.182.131 | grep webcache| awk '{print $14}' | uniq
1556172799:1556172799(0)
```

6. Correlations:

Submission to MyNetWatchman did not provide any other results. This trace is over 1 year old and this probing for an available (and open) proxy may have been an isolated incident and not an example of an ongoing launching pad for a variety of probes and attacks. The http://isc.incidents.org/country_report.html for 2003-07-13 listed CZ as approximately ranked 47th in country reports. For the dates 2002-06-15 to 2002-06-22 no data was returned from the country report query.

7. Evidence of active targeting:

I continued to examine logs from http://www.incidents.org/logs/Raw beginning with file 2002.5.10 to 2002.5.26 inclusive. In total this source IP generated 35 alerts from 2002-06-15 18:34:43 to 2002-06-22 11:20:

```
]# for list in `ls /u01/snort/logs/2002*`; do echo $list  `/usr/sbin/tcpdump -nv  -r $list host
194.108.153.205 and dst port 8080 | wc -l`; done
/u01/snort/logs/2002.5.10 0
/u01/snort/logs/2002.5.11 0
/u01/snort/logs/2002.5.12 0
/u01/snort/logs/2002.5.13 0
/u01/snort/logs/2002.5.14 0
/u01/snort/logs/2002.5.15 3
/u01/snort/logs/2002.5.16 6
/u01/snort/logs/2002.5.17 6
/u01/snort/logs/2002.5.18 3
/u01/snort/logs/2002.5.19 6
/u01/snort/logs/2002.5.20 0
/u01/snort/logs/2002.5.21 5
/u01/snort/logs/2002.5.22 6
/u01/snort/logs/2002.5.23 0
/u01/snort/logs/2002.5.24 0
/u01/snort/logs/2002.5.25 0
/u01/snort/logs/2002.5.26 0
```

Data extracted from file 2002.5.16 exhibits some similar characteristics (3 SYNs of the same initial sequence number) as does the probe from 2005.5.15.

```
 [root@localhost logs]# /usr/sbin/tcpdump -n  -r 2002.5.16 host 194.108.153.205 and dst port 8080
22:24:02.054488 194.108.153.205.1869 > 46.5.92.131.webcache: S 3676586268:3676586268(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
22:24:05.404488 194.108.153.205.1869 > 46.5.92.131.webcache: S 3676586268:3676586268(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
22:24:11.854488 194.108.153.205.1869 > 46.5.92.131.webcache: S 3676586268:3676586268(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
04:31:54.124488 194.108.153.205.4537 > 46.5.62.131.webcache: S 717773599:717773599(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
04:31:57.364488 194.108.153.205.4537 > 46.5.62.131.webcache: S 717773599:717773599(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
04:32:03.934488 194.108.153.205.4537 > 46.5.62.131.webcache: S 717773599:717773599(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
```

There does not appear to be any periodicity to the timing of the probes.

```
[root@localhost logs]# /usr/sbin/tcpdump -n  -r 2002.5.17 host 194.108.153.205 and dst port 8080
07:43:39.484488 194.108.153.205.2165 > 46.5.26.131.webcache: S 3208196366:3208196366(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
07:43:42.734488 194.108.153.205.2165 > 46.5.26.131.webcache: S 3208196366:3208196366(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
07:43:49.284488 194.108.153.205.2165 > 46.5.26.131.webcache: S 3208196366:3208196366(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
11:32:58.454488 194.108.153.205.2394 > 46.5.108.131.webcache: S 627850087:627850087(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
11:33:01.704488 194.108.153.205.2394 > 46.5.108.131.webcache: S 627850087:627850087(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
11:33:08.264488 194.108.153.205.2394 > 46.5.108.131.webcache: S 627850087:627850087(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
```

8. Severity:

The calculation for severity is from http://www.giac.org/GCIA_assignment.php: severity = (criticality + lethality) – (system countermeasures + network countermeasures) with each value ranked on a scale from 1 (lowest) to 5 (highest).

Criticality is calculated to be a value of 1 since there is no additional traffic capture flowing from or to this IP. Since this is a scan and not an actual attack I value this a 1 for lethality. System countermeasures are unknown; the host did not continue the 3 way handshake and establish a connection. I will value this a 2. I will make the assumption that one of these Cisco devices may be a router with some ACL capability. Router ACLs may be configured so I will estimate this at a 2 for network countermeasures. Therefore the Severity is calculated to be (1+1)-(2+2) = -2 quite low,

9. Defensive recommendation:

Since this probing was unanswered by the targeted machine, but other hosts were probed in the logs from file 2002.5.10 to 2002.5.26 inclusive, my suggestion would be to create border ACLs dropping inbound SYNs from 194.108.153.205. Insure that any hosts running a web cache and/or proxy are up to date with respect to operating system patches and application patches. Insure that the configurations of any machines on the internal network are properly configured so they no longer pose a risk.

10. Multiple choice test question:

Use the following trace to choose the best response:

```
[root@localhost logs]# /usr/sbin/tcpdump -n -e -r 2002.5.15  dst port 8080 and host 46.5.182.131
18:34:43.634488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S 1556172799:1556172799(0) win 16384 <mss 1460,nop,nop,sackOK>
(DF)
18:34:46.864488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S 1556172799:1556172799(0) win 16384 <mss 1460,nop,nop,sackOK>
(DF)
18:34:53.434488 0:3:e3:d9:26:c0 0:0:c:4:b2:33 ip 62: 194.108.153.205.4609 >
46.5.182.131.webcache: S 1556172799:1556172799(0) win 16384 <mss 1460,nop,nop,sackOK>
(DF)
```

What component of the above packet trace likely indicates crafting?
a) A window size of 16384?
b) Ip 62 – a packet length of 62 bytes?
c) An initial sequence number from 3 packets being the same 1556172799?
d) None of the above.

Answer: c -- An initial sequence number from 3 packets being the same 1556172799. RFC1948 discusses the prevention of sequence number guessing attacks by suggesting that each Initial Sequence Number be a function of the microsecond timer plus some function of source IP address, source port, destination IP and destination port). The equation given is ISN = M + F(localhost, localport, remotehost, remoteport). Therefore unique sequence numbers should be seen for every unique connection, in this case each SYN is a new attempt at a connection (IP+port=socket). Visit http://www.faqs.org/rfcs/rfc1948.html and http://razor.bindview.com/publish/papers/tcpseq.html for details.

From: John Ruiz <flippedman@yahoo.com>
Subject: Re: GIAC GCIA version 3.3 practical detect #1
Regarding the probability the source address was spoofed, how does doing a nslookup tell you that the source address was spoofed? It could still be spoofed whether the nslookup says it came from Data Tech in the Czech Republic or the Pentagon. Basically, did you "calculate" probability only on the basis of doing an nslookup or did you also consider the nature of the attack?

Response:
I executed the nslookup command to see if there was a FQDN for the source IP address. This is a probe looking for a destination port listening and waiting to accept the connection. It requires the completion of the TCP three-way handshake. I observe that the source would require the return and establishment of the connection. They are looking for a proxy, and I assume that given the FQDN, it's quite possibly a compromised server.

DETECT NUMBER TWO - OBSOLETE TCP OPTIONS FOUND

Posted to intrusions@incidents.org Wed, 16 Jul 2003 22:26:29 -0500.

Note: Destination IP subject to obfuscation. Destination IP address changed to IANA Private Class C IP address (192.168.XXX.XXX).

The packet information was extracted using the following command:

```
[root@localhost caps]# /usr/sbin/tcpdump -n -v -r snort.log.1051560674 host 140.31.33.6 -Xx -e
15:28:11.246061 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 78: 140.31.33.6.3811 > 192.168.XXX.XXX.ftp: S
[tcp sum ok] 766080929:766080929(0) win 65535 <mss 1460,wscale 4,nop,timestamp
3588676200 0,echo 3588676200> (DF) (ttl 52, id 30849, len 64)
0x0000   4500 0040 7881 4000 3406 6fa9 8c1f 2106        E..@x.@.4.o...!.
0x0010   0000 0000 0ee3 0015 2da9 77a1 0000 0000        .J0.....-.w.....
0x0020   b002 ffff bf8e 0000 0204 05b4 0303 0401        ................
0x0030   080a d5e6 da68 0000 0000 0606 d5e6 da68        .....h.........h
15:28:22.314866 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 78: 140.31.33.6.3812 >
192.168.XXX.XXX.34166: S [tcp sum ok] 768837331:768837331(0) win 65535 <mss
1460,wscale 4,nop,timestamp 3588687261 0,echo 3588687261> (DF) (ttl 52, id 31575, len 64)
0x0000   4500 0040 7b57 4000 3406 6cd3 8c1f 2106        E..@{W@.4.l...!.
0x0010   0000 0000 0ee4 8576 2dd3 86d3 0000 0000        .J0....v-.......
0x0020   b002 ffff d465 0000 0204 05b4 0303 0401        .....e..........
0x0030   080a d5e7 059d 0000 0000 0606 d5e7 059d        ................
15:29:17.602044 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 78: 140.31.33.6.3814 >
192.168.XXX.XXX.34168: S [tcp sum ok] 782632098:782632098(0) win 65535 <mss
1460,wscale 4,nop,timestamp 3588742548 0,echo 3588742548> (DF) (ttl 52, id 60770, len 64)
0x0000   4500 0040 ed62 4000 3406 fac7 8c1f 2106        E..@.b@.4.....!.
0x0010   0000 0000 0ee6 8578 2ea6 04a2 0000 0000        .J0....x........
0x0020   b002 ffff a5d0 0000 0204 05b4 0303 0401        ................
0x0030   080a d5e7 dd94 0000 0000 0606 d5e7 dd94        ................
15:29:41.768937 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 78: 140.31.33.6.3817 >
192.168.XXX.XXX.34170: S [tcp sum ok] 789062604:789062604(0) win 65535 <mss
1460,wscale 4,nop,timestamp 3588766720 0,echo 3588766720> (DF) (ttl 52, id 7830, len 64)
0x0000   4500 0040 1e96 4000 3406 c994 8c1f 2106        E..@..@.4.....!.
0x0010   0000 0000 0ee9 857a 2f08 23cc 0000 0000        .J0....z/.#.....
0x0020   b002 ffff c966 0000 0204 05b4 0303 0401        .....f..........
0x0030   080a d5e8 3c00 0000 0000 0606 d5e8 3c00        ....<.........<.
```

And again:

```
 [root@localhost caps]# /usr/sbin/tcpdump -n -v -r snort.log.1051537262 host 140.31.33.6 -Xx
13:05:21.055215 140.31.33.6.4941 > 192.168.XXX.XXX.ftp: S [tcp sum ok]
2589578022:2589578022(0) win 65535 <mss 1460,wscale 4,nop,timestamp 3580106420 0,echo
3580106420> (DF) (ttl 52, id 53328, len 64)
0x0000   4500 0040 d050 4000 3406 17da 8c1f 2106        E..@.P@.4.....!.
0x0010   0000 0000 134d 0015 9a59 d326 0000 0000        .J0..M...Y.&....
0x0020   b002 ffff 7b5c 0000 0204 05b4 0303 0401        ....{\..........
0x0030   080a d564 16b4 0000 0000 0606 d564 16b4        ...d.........d..
13:05:33.619094 140.31.33.6.4942 > 192.168.XXX.XXX.34157: S [tcp sum ok]
2592964060:2592964060(0) win 65535 <mss 1460,wscale 4,nop,timestamp 3580118982 0,echo
3580118982> (DF) (ttl 52, id 53397, len 64)
0x0000   4500 0040 d095 4000 3406 1795 8c1f 2106        E..@..@.4.....!.
0x0010   0000 0000 134e 856d 9a8d 7ddc 0000 0000        .J0..N.m..}.....
0x0020   b002 ffff e8f4 0000 0204 05b4 0303 0401        ................
0x0030   080a d564 47c6 0000 0000 0606 d564 47c6        ...dG........dG.
13:05:40.052047 140.31.33.6.4943 > 192.168.XXX.XXX.34158: S [tcp sum ok]
2594640813:2594640813(0) win 65535 <mss 1460,wscale 4,nop,timestamp 3580125416 0,echo
3580125416> (DF) (ttl 52, id 53459, len 64)
0x0000   4500 0040 d0d3 4000 3406 1757 8c1f 2106        E..@..@.4..W..!.
0x0010   0000 0000 134f 856e 9aa7 13ad 0000 0000        .J0..O.n........
0x0020   b002 ffff 20c4 0000 0204 05b4 0303 0401        ................
0x0030   080a d564 60e8 0000 0000 0606 d564 60e8        ...d`........d`.
13:06:20.057134 140.31.33.6.4954 > 192.168.XXX.XXX.34159: S [tcp sum ok]
2605836660:2605836660(0) win 65535 <mss 1460,wscale 4,nop,timestamp 3580165421 0,echo
3580165421> (DF) (ttl 52, id 53708, len 64)
0x0000   4500 0040 d1cc 4000 3406 165e 8c1f 2106        E..@..@.4..^..!.
0x0010   0000 0000 135a 856f 9b51 e974 0000 0000        .J0..Z.o.Q.t....
0x0020   b002 ffff 11bb 0000 0204 05b4 0303 0401        ................
0x0030   080a d564 fd2d 0000 0000 0606 d564 fd2d        ...d.-.......d.-
13:06:42.961643 140.31.33.6.4960 > 192.168.XXX.XXX.34160: S [tcp sum ok]
2612041277:2612041277(0) win 65535 <mss 1460,wscale 4,nop,timestamp 3580188323 0,echo
3580188323> (DF) (ttl 52, id 53901, len 64)
0x0000   4500 0040 d28d 4000 3406 159d 8c1f 2106        E..@..@.4.....!.
0x0010   0000 0000 1360 8570 9bb0 963d 0000 0000        .J0..`.p...=....
0x0020   b002 ffff b19f 0000 0204 05b4 0303 0401        ................
0x0030   080a d565 56a3 0000 0000 0606 d565 56a3        ...eV........eV.
[root@localhost caps]#
```

The alert is:

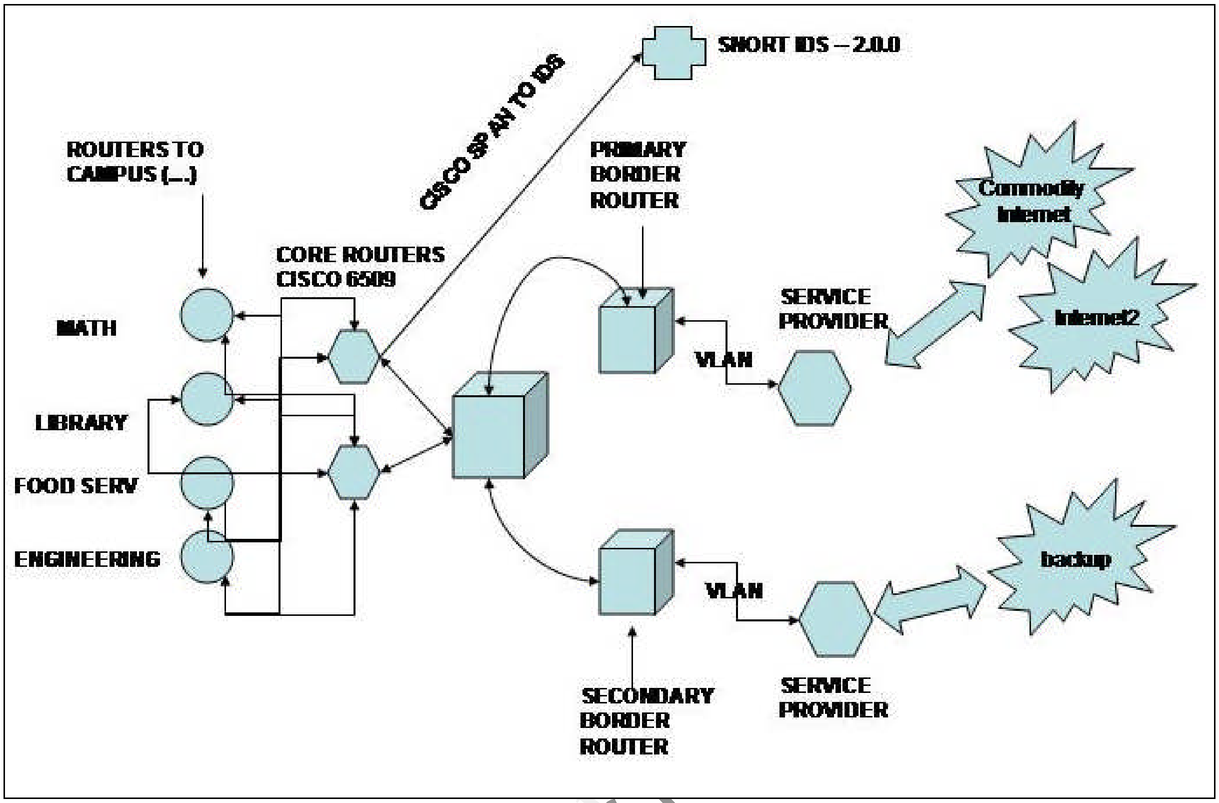
```
[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-13:05:21.055215 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:53328 IpLen:20 DgmLen:64 DF
******S* Seq: 0x9A59D326  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3580106420 0 Echo: 3580106420

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-13:05:21.055215 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:53328 IpLen:20 DgmLen:64 DF
******S* Seq: 0x9A59D326  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3580106420 0 Echo: 3580106420

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-13:05:40.052047 140.31.33.6:0 -> 192.168.XXX.XXX:0
```

```
TCP TTL:52 TOS:0x0 ID:53459 IpLen:20 DgmLen:64 DF
******S* Seq: 0x9AA713AD  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3580125416 0 Echo: 3580125416

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-13:06:20.057134 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:53708 IpLen:20 DgmLen:64 DF
******S* Seq: 0x9B51E974  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3580165421 0 Echo: 3580165421

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-13:06:42.961643 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:53901 IpLen:20 DgmLen:64 DF
******S* Seq: 0x9BB0963D  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3580188323 0 Echo: 3580188323

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-15:28:11.246061140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:30849 IpLen:20 DgmLen:64 DF
******S* Seq: 0x2DA977A1  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3588676200 0 Echo: 3588676200

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-15:28:22.314866 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:31575 IpLen:20 DgmLen:64 DF
******S* Seq: 0x2DD386D3  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3588687261 0 Echo: 3588687261

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-15:29:17.602044 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:60770 IpLen:20 DgmLen:64 DF
******S* Seq: 0x2EA604A2  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3588742548 0 Echo: 3588742548

[**] [116:57:1] (snort_decoder): Obsolete TCP Options found [**]
04/28-15:29:41.768937 140.31.33.6:0 -> 192.168.XXX.XXX:0
TCP TTL:52 TOS:0x0 ID:7830 IpLen:20 DgmLen:64 DF
******S* Seq: 0x2F0823CC  Ack: 0x0  Win: 0xFFFF  TcpLen: 44
TCP Options (5) => MSS: 1460 WS: 4 NOP TS: 3588766720 0 Echo: 3588766720
```

The event that matched this pattern of activity was actually the snort_decoder and not an actual snort rule.

1. Source of trace:

The source of this trace was from data gathered from my employer's network. The network is a large university campus class B size (/16) network. It was (and still is) sub-netted (or super-netted) into all sizes of networks (/19 - /30). The IDS sensor was placed on a spanned segment representative of commodity traffic passing thru the border.



According to Cisco documentation spanning (SPAN) mirrors traffic from one or more source ports on any VLAN (and from one or more VLANs) to a destination port for analysis. Spanning (SPAN) does not affect the switching of network traffic on source ports; a copy of the packets received or transmitted by the source ports is sent to the destination port. This may not be the best way to utilize an IDS, but that is a current work in progress at my employer.

2. Detect was generated by:

The IDS sensor was a Dual Xeon 2.4Ghz IBM power station running RedHat AE 2.1. Several packages were downloaded from their respective sources and compiled to complete my combination IDS and analysis workstation. Alerts were NOT logged to the snort database but logged to a tcpdump capture file for offline analysis. The build was Snort 2.0 build 72. Data was captured from 04/28/2003-08:41 until 04/29/2003-00:20.

Note: There was no rule that triggered the alert “Obsolete TCP Options found”, but the snort decoder. As noted by Jack Koziol in “Intrusion Detection with Snort” (ISBN: 157870281X) Snort is comprised of five pieces that make it the IDS it is. The initial component is the packet capturing library (libpcap) which is used to capture packets. Once packets are in raw form, they are passed to the packet decoder. The packet decoder translates specific protocol elements into Snort’s internal data structure. Once packets are captured and decoded pluggable preprocessors examine packets and send them off to the detection engine.

In this detect the decoder triggered on the discovery of RFC 1072[16] TCP Echo and TCP Echo Reply Options. Echo Reply was made obsolete by option "TSopt" - Time Stamp Option in RFC 1323[17].

3. Probability the source address was spoofed:

This activity requires the completion of the TCP three-way handshake. Spoofing is unlikely since the contact between hosts was connection oriented as it seemed to be a probe or stimulus; I determine it unlikely that the source address was spoofed. Dig information for the source IP is as follows:

```
Search results for: 140.31.33.6
OrgName:    DoD Network Information Center
OrgID:      DNIC
NetRange:   140.31.0.0 - 140.31.255.255
CIDR:       140.31.0.0/16
# ARIN WHOIS database, last updated 2003-07-15 22:50
```

If the IP was spoofed, it would be suggested that a different source IP would be injected each time to attempt to obscure the behavior in mountains of logs. I suggest that this was some sort of stimulus/response probe and correct routable source IP information is necessary.

4. Description of attack:

Neither Google nor Securityfocus or CERT provided info on exploits related to this traffic. This does not appear to be an outright attack. This is interesting behavior which exhibits traits similar to probing and or some solicitation of a response. Excerpted details from the ACID analysis console clearly shows this initial SYN on port 21 followed by additional SYNs on increasing destination ports. The increasing destination port number is interesting in that it increases by a value of 1.

```
Obsolete TCP Options found:
2003-04-28 13:05:21    140.31.33.6:4941    192.168.XXX.XXX:21       TCP options:
2003-04-28 13:05:33    140.31.33.6:4942    192.168.XXX.XXX:34157       TCP
2003-04-28 13:05:40    140.31.33.6:4943    192.168.XXX.XXX:34158       TCP
2003-04-28 13:06:20    140.31.33.6:4954    192.168.XXX.XXX:34159       TCP
2003-04-28 13:06:42    140.31.33.6:4960    192.168.XXX.XXX:34160       TCP
2003-04-28 15:28:11    140.31.33.6:3811    192.168.XXX.XXX:21          TCP
2003-04-28 15:28:22    140.31.33.6:3812    192.168.XXX.XXX:34166       TCP
2003-04-28 15:29:17    140.31.33.6:3814    192.168.XXX.XXX:34168       TCP
2003-04-28 15:29:41    140.31.33.6:3817    192.168.XXX.XXX:34170       TCP
```

According to RFC 1072, the purpose of the TCP ECHO option is to provide a method for measuring the RTT of a segment. The data is there with the initial SYN, what is unusual is that this is not the preferred method as this RFC is not in use according to references (listed above footnotes). Utilizing version 1.8.3 for windows of p0f from www.stearns.org/p0f/ did not facilitate the identification of a host. The files capture2a and capture2b contain the packets from the above alert.

```
p0f: passive os fingerprinting utility, version 1.8.3
(C) Michal Zalewski <lcamtuf@gis.net>, William Stearns <wstearns@pobox.com>
p0f: file: 'p0f.fp', 3001 fprints, iface: '\', rule: 'all'.
<Wed Jul 16 20:42:35 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3811 -> 192.168.XXX.XXX:21 (timestamp: 3588676200 @1058406155)
<Wed Jul 16 20:42:35 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3812 -> 192.168.XXX.XXX:34166 (timestamp: 3588687261 @1058406155)
<Wed Jul 16 20:42:35 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3814 -> 192.168.XXX.XXX:34168 (timestamp: 3588742548 @1058406155)
<Wed Jul 16 20:42:35 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3817 -> 192.168.XXX.XXX:34170 (timestamp: 3588766720 @1058406155)
C:\RAWFILES>p0f-1.8.3 -f p0f.fp -v -s c:\capture2b -t
p0f: passive os fingerprinting utility, version 1.8.3
(C) Michal Zalewski <lcamtuf@gis.net>, William Stearns <wstearns@pobox.com>
p0f: file: 'p0f.fp', 3001 fprints, iface: '\', rule: 'all'.
<Wed Jul 16 20:42:41 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3811 -> 192.168.XXX.XXX:21 (timestamp: 3588676200 @1058406161)
<Wed Jul 16 20:42:41 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3812 -> 192.168.XXX.XXX:34166 (timestamp: 3588687261 @1058406161)
<Wed Jul 16 20:42:41 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3814 -> 192.168.XXX.XXX:34168 (timestamp: 3588742548 @1058406161)
<Wed Jul 16 20:42:41 2003> 140.31.33.6: UNKNOWN [65535:52:1460:1:4:0:1:64].
 + 140.31.33.6:3817 -> 192.168.XXX.XXX:34170 (timestamp: 3588766720 @1058406161)
```

5. Attack mechanism:

This is not an attack, but a reconnaissance attempt initially and possibly some sort of response stimulation. The host in this scan at IP 140.31.33.6 is attempting to complete the 3 way handshake to establish a connection to the host with the destination (obfuscated) IP of 192.168.XXX.XXX. The initial SYN is an attempt to establish a connection. If the destination host was listening for a connection, the destination host would normally respond with a SYN-ACK returning a new sequence number and an acknowledgement sequence number reflecting the initial sequence number from the source host. The conversation would continue with the ACK in return, setting the stage for the exchange of information. Since there is no SYN-ACK returned, the source can not proceed and create a socket (IP + port = socket), and subsequently read and write to that socket descriptor returned. Interesting to note is that all packets had TCP options set for echo and included timestamp information in accord with RFC 1072. Passive OS fingerprinting did not reveal any additional information.

6. Correlations:

This probe has not been seen before in logs and recent IP flow data does not show traffic between the source and destination IP.

7. Evidence of active targeting:

This appears to be targeted at an individual host and was not seen against other machines on the network. Of note is that this alert “Obsolete TCP Options found” was not seen on any other traffic collected out of the 18 gigabytes of logs collected in for several days April 2003. Since collection was limited to several days, additional activity is unknown.

8. Severity:

Severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality is calculated to be a value of 1 since there is no additional traffic capture Netflow from or to this IP recently (as far as May 9, 2003). No information exists concerning the use of this machine. This is an example of unusual traffic; I submit a value of 3.5 for lethality. Unusual traffic from a DOD machine further increases the interest level. The pseudo-stimulation effect (as if knocking on the front door of a house – 1 SYN to ftp) and trying to open a window is interesting (SYNs with obsolete TCP options to higher destination ports increasing each time by a value of +1). The data in the TCP option field is ‘increasing’ as if it is generated from ‘time information’ at which the data segment was transmitted according to RFC 1072. The destination host did not continue the 3 way handshake and did not establish a connection as evident from logs.

As an additional University OIT staff may perform an nmap scan of a destination machine on the campus network for information gathering. Campus staff avoid the scanning of external IP addresses which is often viewed as aggressive behavior. It is acceptable policy for OIT (Office of Information Technology) staff to scan University machines. Since this machine is on campus permission from a 3rd party () is not required but instead given by the director of network engineering. As a side note, University OIT information security staff have an active program of utilizing Nessus to sweep subnets for machines and identify those with potential security loopholes.

An nmap v 3.00 scan of the University machine identified by the destination IP address in the above alert revealed the following information:

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Host  (192.168.XXX.XXX) appears to be up ... good.
Interesting ports on XXXX.nd.edu (192.168.XXX.XXX):
Port       State       Service
21/tcp     open        ftp
22/tcp     open        ssh
23/tcp     open        telnet
25/tcp     open        smtp
67/tcp     filtered    dhcpserver
68/tcp     filtered    dhcpclient
69/tcp     filtered    tftp
111/tcp    filtered    sunrpc
135/tcp    filtered    loc-srv
```

```
136/tcp    filtered    profile
137/tcp    filtered    netbios-ns
138/tcp    filtered    netbios-dgm
139/tcp    filtered    netbios-ssn
161/tcp    filtered    snmp
162/tcp    filtered    snmptrap
280/tcp    filtered    http-mgmt
445/tcp    filtered    microsoft-ds
512/tcp    open         exec
513/tcp    open         login
514/tcp    open         shell
515/tcp    filtered    printer
587/tcp    open         submission
601/tcp    open         unknown
665/tcp    open         unknown
898/tcp    open         unknown
1433/tcp   filtered    ms-sql-s
1434/tcp   filtered    ms-sql-m
1993/tcp   filtered    snmp-tcp-port
2049/tcp   open         nfs
2301/tcp   filtered    compaqdiag
4045/tcp   open         lockd
5000/tcp   filtered    UPnP
6000/tcp   open         X11
32771/tcp  open         sometimes-rpc5
32772/tcp  open         sometimes-rpc7
32773/tcp  open         sometimes-rpc9
32774/tcp  open         sometimes-rpc11
32778/tcp  open         sometimes-rpc19
```

The result of the scan indicates that some ports are open and appear not to be filtered, while some are filtered. OS Patch levels are unknown. Given the nmap scan, I submit a value for system countermeasures as 1. This is troublesome. The Internet is available to all University machines on the network. Router ACLs could be configured if necessary to quell any malevolent traffic from the border routers, but are not in place. I will estimate this at a value of 1 for network countermeasures.

Therefore the Severity is calculated to be (1+3.5)-(1+1) = 2.5 medium interest and worth mention.

9. Defensive recommendation:

The defensive recommendation is to ascertain the actual operating system running on the 192.168.XXX.XXX host and inquire as to the nature of the machine. Questions such as “Is the machine used for research?” “Have you checked to insure that a host based firewall is properly configured?” seem to be in order. Antivirus software and/or software which inspects the integrity of operating system files should be installed. Additional inspection of intrusion logs for events beyond this time period is warranted as part of a risk mitigation strategy.

10. Multiple choice test question:

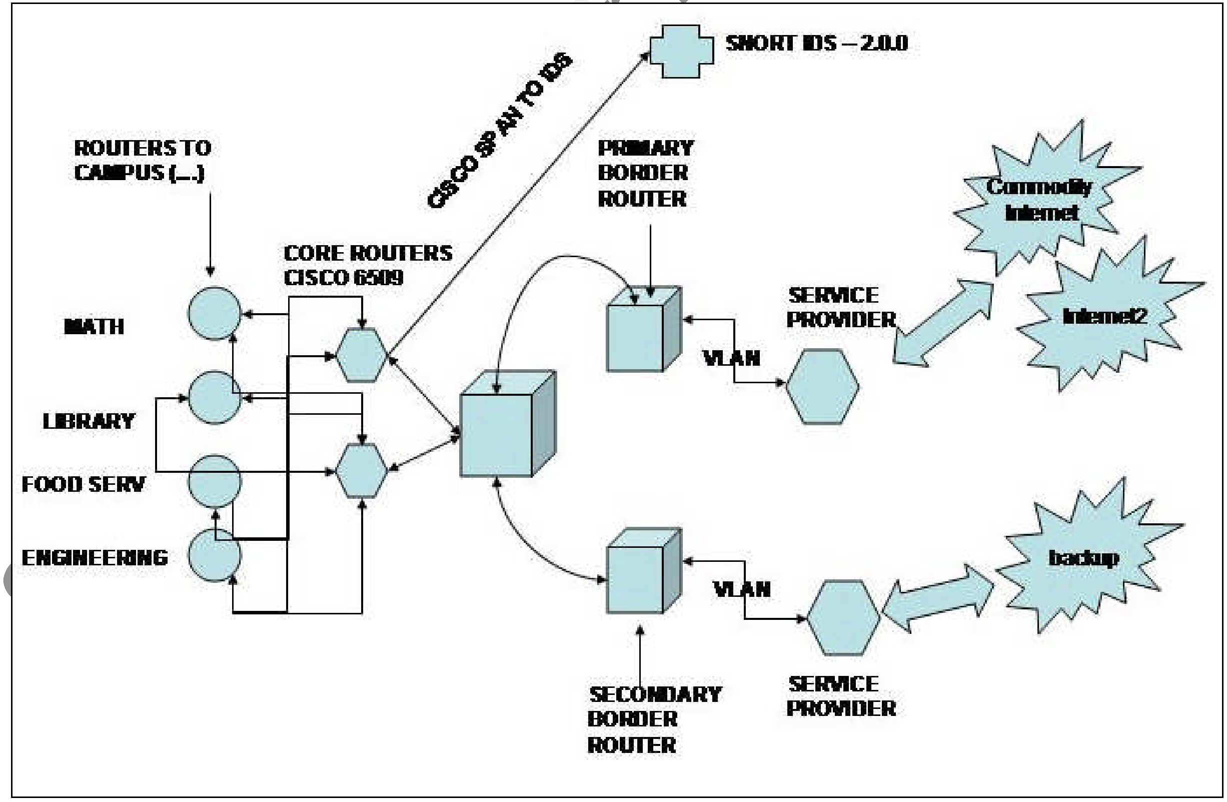
Which RFC obsoletes RFC 1072?
a) RFC 1323
b) RFC 1045
c) RFC 1185
d) RFC 1205

Answer a) RFC 1323 - TCP Extensions for High Performance (http://www.faqs.org/rfcs/rfc1323.html).

DETECT THREE WEB-IIS UNICODE DIRECTORY TRAVERSAL ATTEMPTS
Posted to intrusions@incidents.org Tue, 22 Jul 2003 21:03:10 -0500

1. Source of Trace:

This trace was obtained from data gathered from my employer's network. The network is a large university campus class B size (/16) network. It was (and still is) sub-netted (or super-netted) into all sizes of networks (/19 - /30). The IDS sensor was placed on a span representative of commodity traffic passing thru the border. See graphic below:



2. Detect was generated by:

The IDS sensor was a Dual Xeon 2.4Ghz IBM power station running RedHat AE 2.1. Several packages were downloaded from their respective sources and compiled to complete my combination IDS and analysis workstation. Alerts were NOT logged to the snort database but logged to a tcpdump capture file for offline analysis. The build was Snort 2.0 build 72. Data was captured from 04/28/2003-08:41 until 04/29/2003-00:20. The alert was:

```
[**] [1:1945:1] WEB-IIS unicode directory traversal attempt [**]
[Classification: Web Application Attack] [Priority: 1]
04/28-18:03:26.586592 80.126.131.196:42917 -> 129.74.XXX.XXX:80
TCP TTL:110 TOS:0x0 ID:21692 IpLen:20 DgmLen:147 DF
***AP*** Seq: 0x67E1777C  Ack: 0x56D00161  Win: 0x4470  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0884]

[**] [1:982:6] WEB-IIS unicode directory traversal attempt [**]
[Classification: Web Application Attack] [Priority: 1]
04/28-18:03:49.261373 80.126.131.196:43108 -> 129.74.XXX.XXX:80
TCP TTL:110 TOS:0x0 ID:30542 IpLen:20 DgmLen:180 DF
***AP*** Seq: 0x68E54B8F  Ack: 0x573A280F  Win: 0x4470  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0884]
```

The snort rule which matched this is from web-iis.rules:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS unicode
directory traversal attempt"; flow:to_server,established; content:"/..%255c.."; nocase;
classtype:web-application-attack; reference:cve,CVE-2000-0884; sid:1945; rev:1;)
```

The rule triggered on the matching content of '%255c'. The packets which caused this alert were extracted utilizing the following command:

```
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 dst host 129.74.XXX.XXX
and src host 80.126.131.196  | grep 18:03:49
18:03:49.261373 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 194: 80.126.131.196.43108 >
129.74.XXX.XXX.http: P [tcp sum ok] 0:140(140) ack 1 win 17520 (DF) (ttl 110, id 30542, len
180)
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 dst host 129.74.XXX.XXX
and src host 80.126.131.196  | grep 18:03:26
18:03:26.586592 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 161: 80.126.131.196.42917 >
129.74.XXX.XXX.http: P [tcp sum ok] 0:107(107) ack 1 win 17520 (DF) (ttl 110, id 21692, len
147)
```

There were more packets directed to this destination host than shown above. These packets will be shown later under attack mechanism in this 3rd detect.

3. Probability the source address was spoofed:

Given the nature of the attack, this is unlikely. This is an attempt to exploit an IIS web server machine. We may not witness the initial SYN of the 3 way handshake, but there is traffic between the hosts, and we note sequence numbers increasing as the conversation continues.

```
[root@localhost caps]# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 dst host
129.74.250.88 and src host 80.126.131.196   | wc -l
     62
```

<SNIP>

```
18:01:36.320008 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 157: 80.126.131.196.42029 >
129.74.XXX.XXX.http: P [tcp sum ok] 1667308644:1667308747(103) ack 1431287454 win
 17520 (DF) (ttl 110, id 43370, len 143)
18:01:40.906178 0:0:c:7:ac:1 0:7:d:e6:67:fc ip 158: 80.126.131.196.42131 > 129
.74.XXX.XXX.http: P [bad tcp cksum 59bf!] 1432826627:1432826731(104) ack 167447
1929 win 64136 [tos 0x10]  (ttl 240, id 0, len 144, bad cksum 0!)
18:01:43.904784 0:0:c:7:ac:1 0:7:d:e6:67:fc ip 152: 80.126.131.196.42165 > 129
.74.XXX.XXX.http: P [bad tcp cksum ec14!] 1433589989:1433590087(98) ack 1677102
736 win 64142 [tos 0x10]  (ttl 240, id 0, len 138, bad cksum 0!)
18:01:45.405537 0:e0:f9:c0:a8:0 0:0:c:7:ac:3 ip 164: 80.126.131.196.42192 > 12
9.74.XXX.XXX.http: P [tcp sum ok] 1679607671:1679607781(110) ack 1434330264 win
 17520 (DF) (ttl 110, id 46732, len 150)
```

The attacking host is resolvable and appears to be a DSL customer. http://www.ripe.net provides the following (truncated) information.

```
% This is the RIPE Whois server.

inetnum:      80.126.0.0 - 80.127.255.255
netname:      NL-XS4ALL-20011011
descr:        PROVIDER
country:      NL
admin-c:      CB127
tech-c:       CB127
tech-c:       OD45
status:       ALLOCATED PA
mnt-by:       RIPE-NCC-HM-MNT
mnt-lower:    XS4ALL-MNT
mnt-routes:   XS4ALL-MNT
changed:      hostmaster@ripe.net 20011011
source:       RIPE

route:        80.126.0.0/15
descr:        XS4ALL networking
mnt-by:       XS4ALL-MNT
changed:      erik@xs4all.net 20011011
source:       RIPE

address:      The Netherlands
mnt-by:       XS4ALL-MNT
changed:      cor@xs4all.nl 19980503
source:       RIPE
```

4. Description of attack:

According to http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0884 and http://securityfocus.com/bid/1806 this attack is an example of an ‘input validation error attack'. The attack targets Microsoft IIS servers which are not utilizing the UrlScan Security Tool and are not patched from Microsoft. Unicode is defined as an attempt to provide a unique number for every character, no matter what the platform, no matter what the program, no matter what the language[18].

Additional searching for 'WEB-IIS unicode directory traversal' revealed a paper from PhiRo-In[19] which highlights several Unicode syntaxes to use in order to attempt exploitation of vulnerable servers. A short sample of the syntaxes from the paper is listed below and subsequently available for snort to pattern match. Presented are the 13 different sets of syntax from the paper *"Exploit the IIS hole using the Echo Style"*, with 1 examples each:

The /SCRIPTS/ syntaxes :
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+
The /MSADC/ syntaxes :
/msadc/..%%35%63..%%35%63..%%35%63..%%35%63winnt/system32/cmd.exe?/c+
The /VTI_BIN/ syntaxes :
/_vti_bin/..%255c..%255c..%255c..%255c..%255c../winnt/system32/cmd.exe?/c+
The /PBSERVER/ syntaxes :
/PBServer/..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+
The /RPC/ syntaxes :
/Rpc/..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+
The /CGI-BIN/ syntaxes :
/cgi-bin/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+
The /IISADMPWD/ syntaxes :
/iisadmpwd/..%c1%1c..%c1%1c..%c1%1c..%c1%1c..%c1%1c../winnt/system32/cmd.exe?/c+
The /SAMPLES/ syntaxes :
/samples/..%c1%1c..%c1%1c..%c1%1c..%c1%1c..%c1%1c../winnt/system32/cmd.exe?/c+
The /_VTI_CNF/ syntaxes :
/_vti_cnf/..%c1%1c..%c1%1c..%c1%1c..%c1%1c..%c1%1c../winnt/system32/cmd.exe?/c+
The /ADSAMPLES/ syntaxes :
/adsamples/..%c1%1c..%c1%1c..%c1%1c..%c1%1c..%c1%1c../winnt/system32/cmd.exe?/c+
The /SRCHADMIN/ syntaxes :
/srchadmin/..%c1%1c..%c1%1c..%c1%1c..%c1%1c..%c1%1c../winnt/system32/cmd.exe?/c+
The /WWWROOT/ syntaxes :
/WWWROOT/..%e0%80%af..%e0%80%af..%e0%80%af..%e0%80%af..%e0%80%af..%e0%80%af..%e0%80%af..%e0%80%af/winnt/system32/cmd.exe?/c+
The /_MEM_BIN/ syntaxes :
/_mem_bin/..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+

By examining the above syntaxes it is hoped to demonstrate to the read the vast amount of different Unicode syntaxes available to be passed, and the variety of syntaxes to be matched in order for this to be recognized. The CVE-2000-0884 reference from http://securityfocus.com/bid/1806/discussion/ provides additional details regarding what happens when any of these Unicode syntaxes are sent to a vulnerable IIS server. An IIS server which is configured to run as privileged service in the operating system is potentially vulnerable because:

> The IUSR_machinename account is a member of the Everyone and Users groups by default, therefore, any file on the same logical drive as any web-accessible file that is accessible to these groups can be deleted, modified, or executed. Successful exploitation would yield the same privileges as a user who could successfully log onto the system to a remote user possessing no credentials whatsoever.

It is of supreme importance to utilize tools similar to the IISLockd Wizard and UrlScan tools from Microsoft if you are running an IIS 4.0 or 5.0 server. Utilizing the wizard to create an alternative user and lock down NTFS permissions on directories served by the web server is a positive step towards preventing exploitation.

5. Attack mechanism:

This attack works by completing a TCP connection to a Microsoft operating system based host running Microsoft IIS version 4.0 or 5.0 and utilizing Unicode to pass particular types of malformed URLs back to the server. Once the connection is complete, malformed URLs like above in section 4 are executed. These requests are processed using (in the default case of an unmodified IIS server) using the security of the IUSR_machinename account. Often this account is able to access files in folders containing programs such as cmd.exe. This OS specific command is a shell which provides the user the ability to execute nefarious commands.

6. Correlations:

This IP address was queried at Dshield.org and did not appear in the Dshield.org database. Submission to mynetwatchman.com returned the following results:

```
Incident Id Source IP  Provider Domain Agent Count  Event Count  Incident Status ISP
Resolution Comments
30430668 80.126.131.196 xs4all.nl 2 2 Closed No Recent Activity
29293286 80.126.131.196 xs4all.nl 6 382 Closed Provider Acknowledged
```

The event http://www.mynetwatchman.com/LID.asp?IID=30430668 referenced a MS-SQL Spida Worm event and event http://www.mynetwatchman.com/LID.asp?IID=29293286 referenced several probes to destination networks following the 129.x.x.x pattern on April 28th, 2003. This correlates to behavior observed on our 129.74.x.x network the same day.

7. Evidence of active targeting:

This source IP of 80.126.131.196 was the source host in 3312 packets out of 493387 packets captured in the snort.log.1051560674 from 04/28/2003-08:41 until 04/29/2003-00:20 representing only .67% of the data.

```
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 src host 80.126.131.196  | wc -l
   3312
```

But this source IP was responsible for http traffic to 43 different hosts and ftp traffic to 215 hosts.

```
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 src host 80.126.131.196  | awk {'print $8'} |
grep http | sort | uniq | wc -l
     43
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 src host 80.126.131.196  | awk {'print $8'} |
grep ftp | sort | uniq | wc -l
    215
```

I conclude this source host was spraying all 129.74 subnets searching for an exploitable host.

8. Severity:

The calculation for severity is from http://www.giac.org/GCIA_assignment.php: severity = (criticality + lethality) – (system countermeasures + network countermeasures) with each value ranked on a scale from 1 (lowest) to 5 (highest).

In this specific case the server is highly critical to the operation of a specific unit making the criticality value 5. The lethality is valued at a 2.5. The nature of this exploit is to gain control of the target by compromise. I submit that the target machine was not compromised by this exploit since the 12 captured packets returned contained data suggesting 403 errors were returned as follows:

```
# /usr/sbin/tcpdump -n -e  -vv -r snort.log.1051560674 src host 129.74.XXX.XXX and dst host
80.126.131.196 | wc -l
12
<SNIP>
18:02:30.078888 0:7:d:e6:67:fc 0:0:c:7:ac:1 ip 198: 129.74.XXX.XXX.http >
80.126.131.196.42463: P [tcp sum ok] 1442901310:1442901454(144) ack 1705964871 win
64116 (DF) (ttl 126, id 28453, len 184)
0x0000   4500 00b8 6f25 4000 7e06 3d35 814a fa58        E...o%@.~.=5.J.X
0x0010   507e 83c4 0050 a5df 5600 ed3e 65ae f547        P~...P..V..>e..G
0x0020   5018 fa74 2b9f 0000 4854 5450 2f31 2e31        P..t+...HTTP/1.1
0x0030   2034 3033 2041 6363 6573 7320 466f 7262        .403.Access.Forb
0x0040   6964 6465 6e0d 0a53 6572 7665 723a 204d        idden..Server:.M
0x0050   6963 726f 736f 6674 2d49 4953 2f35 2e30        icrosoft-IIS/5.0
```

The system countermeasures are valued at 2 since it is believed at the time the target was running some form of UrlScan, but its MS patch level was unknown. The Internet is available to machines on the network. Router ACLs could be configured if necessary to quell any malevolent traffic from the border routers, but are not in place and not proactive but reactive. I will estimate this at a 1 for network countermeasures. Therefore severity = (5 + 2.5) – (2+1) = 4.5 Severe!

9. Defensive recommendation:

Defensive recommendations with respect to the host are to insure that the latest versions of operating system patches are installed. Usage of IISLockd along with the latest version of UrlScan should be used. If possible a host based firewall should be installed blocking HTTP traffic to only specific networks if the business purpose of the target permits. At first notice of such patterns in the target's W3SVC and UrlScan logs along with correlation to Snort IDS logs & IP flow information, border router ACLs should be configured to block 80.126.131.196

10. Multiple choice test question:

Which of the following is the recommended risk mitigation strategy for running Microsoft IIS servers?

a) Run Apache on Windows 2000 Server in the future.
b) Lock down access control on folders that are part of the operating system.
c) Utilize the IIS Lockdown Wizard version 2.1 from www.microsoft.com and be sure to properly configure UrlScan for your location.
d) Apply a template from http://www.cisecurity.org/bench_win2000.html in the Security Configuration Manager via the Microsoft Management Console (MMC) snap-in which reflects the reflects the documents developed by various security authoritative resources (like the SANS Institute).
e) Visit Microsoft Update often and apply any patches.

Answer:

I submit a combination of answers b and c and d. Rather than visit http://v4.windowsupdate.microsoft.com/en/default.asp , I suggest utilizing the Microsoft Baseline Security Analyzer v1.1.1 which according to Microsoft.com superior tool for applying patches and best security practices for risk mitigation.

PART THREE - ANALYZE THIS

For this final part of the practical assignment, the logs from an unknown University were downloaded from http://www.incidents.org/logs. Below are lists of the five consecutive days' of data files selected:

Log Files	***Scan Files***	***Out of Spec Files***
alert.030711	scans.030711	OOS_Report_2003_07_11_27931
alert.030712	scans.030712	OOS_Report_2003_07_12_20109
alert.030713	scans.030713	OOS_Report_2003_07_13_9896
alert.030714	scans.030714	OOS_Report_2003_07_14_3882
alert.030715	scans.030715	OOS_Report_2003_07_15_23381

After downloading all the files from the website, a Perl script csv.pl was utilized to delimit the files for importing into a FoxPro database. This Perl script was utilized in Brian Cahoon's[20] practical and was created by Tod Beardsley[21] and used in his practical assignment (Thank you!). All three files were imported into Visual FoxPro 8 and ad hoc SQL statements were executed against the data.

INITIAL RESULTS

FILE DATE	***NUMBER OF ALERTS***	***PERCENT OF TOTAL***	***NUMBER OF SCANS***	***PERCENT OF TOTAL***
July 11 2003	134021	21.5%	1922317	20.0%
July 12 2003	213025	34.1%	1868520	19.4%
July 13 2003	86854	13.9%	1953613	20.3%
July 14 2003	78603	12.6%	1632673	17.0%
July 15 2003	67070	10.7%	1516769	15.8%
July 16 2003	44532	7.1%	730928	7.6%
		100.0%		100.0%
TOTAL EVENTS	624379		9624835	

It is not possible to determine the exact network architecture, but certain characteristics are discernable. Processing of the data was initiated by examining the alert files. The total alerts for the five day period are summarized below in this table. Each alert in this table is briefly described and any relevant data discovered is highlighted in the following paragraphs.

	DESCRIPTION OF ALERT	***COUNT***	***PERCENTAGE***
1	CS WEBSERVER - external web traffic	160128	25.6%
2	High port 65535 tcp - possible Red Worm – tra	135012	21.6%
3	SMB Name Wildcard	78485	12.6%
4	spp_http_decode: IIS Unicode attack detected	72217	11.6%
5	MY.NET.30.4 activity	57470	9.2%
6	SYN-FIN scan!	36255	5.8%
7	EXPLOIT x86 NOOP	33907	5.4%
8	spp_http_decode: CGI Null Byte attack detected	11099	1.8%
9	MY.NET.30.3 activity	9085	1.5%
10	Queso fingerprint	7943	1.3%
11	Null scan!	3507	0.6%
12	Possible trojan server activity	3444	0.6%
13	connect to 515 from inside	3325	0.5%
14	TCP SRC and DST outside network	1920	0.3%
15	High port 65535 udp - possible Red Worm -	1848	0.3%
16	connect to 515 from outside	1757	0.3%
17	IDS552/web-iis_IIS ISAPI Overflow ida nosize	1662	0.3%
18	IDS552/web-iis_IIS ISAPI Overflow ida INTERN	863	0.1%
19	NMAP TCP ping!	861	0.1%

20	Incomplete Packet Fragments Discarded	815	0.1%
21	NIMDA - Attempt to execute cmd from campus	422	0.1%
22	scan (Externally-based)	261	0.0%
23	SNMP public access	252	0.0%
24	SUNRPC highport access!	211	0.0%
25	External RPC call	199	0.0%
26	[UMBC NIDS IRC Alert] IRC user /kill detected	162	0.0%
27	SMB C access	153	0.0%
28	Notify Brian B. 3.54 tcp	94	0.0%
29	Notify Brian B. 3.56 tcp	89	0.0%
30	FTP passwd attempt	83	0.0%
31	EXPLOIT x86 stealth noop	83	0.0%
32	TFTP - Internal UDP connection to external tftpr	79	0.0%
33	CS WEBSERVER - external ftp traffic	74	0.0%
34	EXPLOIT x86 setuid 0	57	0.0%
35	Tiny Fragments - Possible Hostile Activity	53	0.0%
36	EXPLOIT x86 setgid 0	52	0.0%
37	RFB - Possible WinVNC - 010708-1	52	0.0%
38	NETBIOS NT NULL session	51	0.0%
39	DDOS shaft client to handler	51	0.0%
40	MYPARTY - Possible My Party infection	50	0.0%
41	FTP DoS ftpd globbing	37	0.0%
42	TFTP - External TCP connection to internal tftpr	22	0.0%
43	TFTP - Internal TCP connection to external tftpr	22	0.0%
44	Probable NMAP fingerprint attempt	21	0.0%
45	[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC	20	0.0%
46	EXPLOIT NTPDX buffer overflow	19	0.0%
47	Attempted Sun RPC high port access	17	0.0%
48	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC	16	0.0%
49	External FTP to HelpDesk MY.NET.70.49	13	0.0%
50	External FTP to HelpDesk MY.NET.70.50	12	0.0%
51	External FTP to HelpDesk MY.NET.53.29	8	0.0%
52	IRC evil - running XDCC	7	0.0%
53	ICMP SRC and DST outside network	6	0.0%
54	DDOS mstream handler to client	5	0.0%
55	Back Orifice	4	0.0%
56	TFTP - External UDP connection to internal tftp server	4	0.0%
57	Traffic from port 53 to port 123	3	0.0%
58	EXPLOIT x86 NOPS	3	0.0%
59	DDOS mstream client to handler	2	0.0%
60	NIMDA - Attempt to execute root from campus host	2	0.0%
61	EXPLOIT FTP passwd retrieval retr path	1	0.0%

62	[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.	1	0.0%
63	[UMBC NIDS IRC Alert] K\:line'd user detected	1	0.0%
64	Bugbear@MM virus in SMTP	1	0.0%

ALERT EVENTS OF INTEREST

1 CS WEBSERVER – external web traffic 160128 25.6%

Events related to CS WEBSERVER all have a common destination IP of MY.NET.100.165 and destination port of 80. 25601 distinct IP addresses generated alerts for MY.NET.100.165. MY.NET.100.165 was the source of 24 alerts related to Trojan (*Possible Trojan server activity*) and Worm (*High port 65535 tcp - possible Red Worm – tr*) activity. Given that 25.6% of alerts were attributed to web traffic, it is important to maintain the highest security standards on this web server with respect to operating system patches and binary patches. The priority of this machine may be high since it appears to be the destination of much port 80 traffic. The percentage of alert events itself is not alarming, but the contents of the packets flowing to the web server may be alarming as we continue below with discussion.

2 High port 65535 tcp - possible Red Worm – traffic 135012 21.6%

Events related to Red Worm traffic involve 133 distinct source IP addresses of which 39 have some sort of MY.NET designation. Code Red[22] is exploiting and propagating to machines on MY.NET subnets. Code Red is also an alert for 164 distinct destination IP addresses of which 37 has some sort of MY.NET designation. This implies that MY.NET machines are actively participating in spreading the worm because they are infected. Data below from the alert table shows activity seems to be exploding on the 12th of July.

DATE	***ALERT DESCRIPTION***	***COUNT***
11	High port 65535 tcp - possible Red Worm - tr	121
12	High port 65535 tcp - possible Red Worm - tr	132903
13	High port 65535 tcp - possible Red Worm - tr	193
14	High port 65535 tcp - possible Red Worm - tr	444
15	High port 65535 tcp - possible Red Worm - tr	342
16	High port 65535 tcp - possible Red Worm - tr	1009

Of particular interest on July 12th are these IP address listed below. Given the count of events for July 12th being 132903, one must assume that University network performance must have been affected by this activity.

DAY	***DESCRIPTION***	***SOURCE IP***	***DESTINATION IP***	***TOTAL EVENTS***

12	High port 65535 tcp	MY.NET.82.36	24.84.205.243	78661
12	High port 65535 tcp	24.84.205.243	MY.NET.82.36	53747
12	High port 65535 tcp	MY.NET.99.51	66.208.20.15	235
12	High port 65535 tcp	66.208.20.15	MY.NET.99.51	154

The machine MY.NET.82.36 and MY.NET.99.51 require cleansing from Code Red and additional security measures to insure this volume of activity is averted. In all 39 internal MY.NET machines were sources for this alert and 37 were destinations for this alert event. According to other practical assignments[23] port 65535 is the backdoor entry point, and should be blocked inbound at the border.

3 SMB Name Wildcard 78485 12.6%

I believe this alert reflects an attempt to access a default share, either C$ or ADMIN$ or $IPC. All alerts destination ports were the NETBIOS name service on port 137. There is a CERT® Vulnerability Note VN-2000-03 which suggests blocking NetBIOS services at the network perimeter. A query of the alert table shows that 1085 distinct external IP addresses have triggered alerts on 1383 distinct MY.NET hosts. All destination IP addresses were probed on destination port 137. It is critical to turn off ports 137,138 and 139 at the border router.

10367 events are from a source IP of 169.254.X.X with 130 MY.NET addresses as targets. This range of IP addresses (from 169.254.0.1 through 169.254.255.254) is reserved by the Internet Assigned Numbers Authority (IANA). This is a result of Windows 2000 using Automatic Private IP Addressing (APIPA) to automate Internet Protocol (IP) configuration of network connections. If a DHCP server is not reached or leased configuration fails, the computer uses APIPA to automatically configure TCP/IP[24]. This is incorrect network behavior and should be rectified by University staff.

4 spp_http_decode: IIS Unicode attack detected 72217 11.6%

According to several postings retrieved from google.com searching, this may be a false alarm. Closer attention must be paid to the actual payload in the packets. These alerts could also be evidence of some worm activity. According to snort.org documentation the alert is evidence an attempt was made to use a unicode encoded representation of a "\" in a URL request. An attacker would be able to access files and directories outside the web root of a vulnerable Internet Information Services (IIS) server. This URL which contains the documentation is located at:http://www.snort.org/snort-db/sid.html?sid=983. I queried the table for events with a source IP beginning with MY.NET and I returned 70696 records. There were 959 distinct alerts where the source IP address was not a MY.NET machine and the destination was a MY.NET machine. This represents 200 distinct MY.NET IP addresses. This is certainly an amazing amount of IIS exploitation occurring on the University network.

5 MY.NET.30.4 activity 57470 9.2%

MY.NET.30.4 appears to be a web server with a high level of activity from several distinct specific source IP addresses. This table illustrates the top 5 source IP addresses. The two primary destination ports are port 80 (commonly HTTP) with 30078 events and port 514 (commonly shell or syslog) with 25623 events. This is about 96% of the traffic, with almost 38% of the traffic from 68.54.93.211 to port 514 (22200 events with destination port 514).

This is suspicious since the traffic between MY.NET.30.4 and 68.54.93.211 shows the source IP's source port increasing with each event (port 1091 ->33494). This suggests scan activity for reconnaissance. Searching through the OOS logs did not have any hits for MY.NET.30.4 or 68.54.93.211. The severity of this event depends on the importance and security level of the machine(s) at MY.NET.30.4. A sampling of the alert data reveals these external IP addresses responsible for generating the activity.

SOURCE IP	***COUNT EVENTS***
68.54.93.211	22311
151.196.21.230	1588
141.149.36.60	770
66.196.72.49	596
66.196.72.56	584
68.170.69.138	573
24.104.7.38	557

6 SYN-FIN scan! 36255 5.8%

This may be a SYN-FIN fragments scan to bypass firewalls. Two specific source IP addresses seem to have generated 36219 (99%) alert events: 142.26.120.7, 20538 events and 195.5.55.32, 15681 events. The alert "SYN-FIN scan!" was generated against 23363 distinct MY.NET IP addresses. 36219 events were from a source port of 21. 36209 events were directed to a destination of port 21. There is an issue with a Microsoft Windows 2000 firewall product passing SYN-FIN packets and it is detailed in http://www.securityfocus.com/bid/4521/discussion/. This activity has been noted by others[25], and it is noted that these scans may be an attempt at exploitation. A list of FTP exploits is available from ISS: http://www.iss.net/security_center/advice/Exploits/Services/FTP/default.htm. The activity is heavily noted in OOS logs and University staff need to immediately address this malevolent behavior.

7 EXPLOIT x86 NOOP 33907 5.4%

I conclude that this alert was triggered by the Snort SHELLCODE x86 NOOP rule (SID 648)[26]. This signature pattern matches consecutive NOOP instructions. According to many sources it is common for buffer overflow code to contain a large sequence of NOOP instructions as it increases the odds of successful execution of the useful shell code[27]. From the alert data, the top 3 destination ports were: port 80 – 29471 events, port 119 – 3905 events and port 166 – 343 events. Often it is suggested that these could be false positives triggered by downloading binary data.

Querying the data reveals 29471 events with a destination port of 80 that were generated by three primary source IP addresses. Other student practical assignments[28] further suggest the potential for exploitation of a target system. But upon further inspection the 3905 port 119 events suggest attempts at transferring binary data, i.e. news from 131.118.254.130 (news.ums.edu). Therefore, University system administration staff should investigate this as a configuration issue and not necessarily a denial of service attack or intrusion event. These 3 IP addresses should be blocked at the border.

SOURCE IP	***COUNT EVENTS***	***PERCENTAGE***
172.176.163.241	10814	37%
217.88.160.45	6615	22%
172.180.87.233	6555	22%

8 spp_http_decode: CGI Null Byte attack detected 11099 1.8%

This alert is not triggered by a rule, but with the http_decode pre-processor. A discussion of the initial CGI Null Byte announcement may be found in Issue 55 of Phrack [29] where 'Rain Forest Puppy' ruminates over Perl allowing NULL characters in variables as data. By hiding commands behind null bytes '%00' and backslash '\' characters[30] the sender of such data would hope to view or modify content source in directories on web servers.

106 distinct MY.NET IP address generated alert events. There were 154 distinct destination IP addresses of which 7 were MY.NET IP addresses. This indicates there may be a significant issue with internal machines on several MY.NET subnets. Since 11072 alert events are due to hosts on the MY.NET network, it is imperative to have some forensic analysis performed. The above conclusions related to null byte attacks is supported by other student practical assignments[31]. Further investigation of these IP addresses below is warranted!

SOURCE IP	***COUNT EVENTS***	***PERCENTAGE***
MY.NET.152.19	1668	5%
MY.NET.97.159	1518	4%

MY.NET.152.251	769	7%
MY.NET.152.252	713	6%
MY.NET.97.58	560	5%

9 MY.NET.30.3 activity 9085 1.5%

Out of the 9085 alert events these distinct source IP addresses generated most of the events which need to be investigated further.

SOURCE IP	***COUNT EVENTS***	***PERCENTAGE***
68.55.52.234	3182	35%
141.149.36.60	1412	16%
68.81.2.19	906	10%
68.33.25.138	606	7%
68.55.250.229	365	4%
68.55.226.150	323	4%
68.55.63.234	255	3%
68.54.90.123	210	2%

Drilling down by destination port reveals a possible explanation for this behavior observed on the University network. MY.NET.30.3 may be a Novell machine since port 524 appears to be very popular[32] with the Novell products. Novell uses port 524 when running in pure IP mode for NCP requests. NCP is the protocol for transmitting information between a NetWare server and its clients. The detailed data below highlights the traffic to the destination port.

SOURCE IP	***DESTINATION PORT***	***COUNT OF EVENTS***
68.55.52.234	**301**	2552
141.149.36.60	524	1412
68.81.2.19	524	906
68.55.52.234	**524**	630
68.33.25.138	524	606
68.55.250.229	524	365
68.55.226.150	524	323
68.55.63.234	524	255
68.54.90.123	524	210
67.249.226.75	524	201
68.50.106.78	524	179
68.55.144.24	524	162

The 68.55.x.x network seems to be a popular origin for this traffic. Given the ARIN information, this is puzzling why a cable modem user(?) may be contacting a Novell Server. Depending on the location of the University in question, I

speculate this may be a home user’s configured machine trying to contact the assumed Novell server (MY.NET.30.3) over IP.

```
# ARIN WHOIS database, last updated 2003-07-30 19:15
# Enter ? for additional hints on searching ARIN's WHOIS database.
[whois.arin.net]


CustName:   Comcast Cable Communications, Inc.
StateProv:  NJ
Country:    US
RegDate:    2003-03-19
Updated:    2003-03-19
NetRange:   68.55.0.0 - 68.55.255.255
CIDR:       68.55.0.0/16
Parent:     NET-68-32-0-0-1
NetType:    Reassigned
Comment:    NONE
RegDate:    2003-03-19
Updated:    2003-03-19
# ARIN WHOIS database, last updated 2003-07-30 19:15
```

Out of spec logs only show 1 other event with MY.NET.30.3 being a targeted destination by a SYN-FIN scan:

```
07/12-16:25:27.006262 195.5.55.32:21 -> MY.NET.30.3:21
TCP TTL:22 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x484FFE76  Ack: 0x6B63C5C1  Win: 0x404  TcpLen: 20
```

More information is required before additional defensive measures should be taken.

10 Queso fingerprint 7943 1.3%

This alert seems to target a wide range of MY.NET hosts. 91 distinct MY.NET hosts were targeted. This alert suggests and attempt at (OS) fingerprinting is occurring. TCP/IP stack information[33] from the remote operating system is queried by Queso. These 5 destination IP addresses received more than 50% of the Queso traffic: MY.NET.25.71 - 1124, MY.NET.25.70 – 1103, MY.NET.25.73 – 1080, MY.NET.25.69 – 1035, and MY.NET.25.72 -1010. If any of these machines are valued ‘5’ relative to the criticality measure for the severity calculation, this scan may signal the onset of additional attacks & probes. They should be secured immediately by the University in question. These two ports: port 25 – 5842 events and port 80 – 1169 events were favorite destination ports for the targeted machines. Targeted MY.NET machines should have (if installed) their mail binaries and web server binaries patched to patched to the latest vendor recommendations to avoid the risk of having the services (if present) exploited.

The following alerts from the alert file represent less than one percent each of the total alerts. These will be discussed in less detail since the above alerts compromise more than 80 percent of total alerts and their volume demands additional attention.

11 Null scan! 3507 0.6%

This alert triggers on TCP packets sent with no flags set. This could be potentially bad once traffic payload is examined. Suggestion: Block source IP's, Investigate traffic.

12 Possible Trojan server activity 3444 0.6%

It could be one of many types of unauthorized programs. Depending on MY.NET machine purpose and OS, compromise could be damaging. For more information the University security professionals or network engineering staff should reference the SANS reading room (UNIX in particular) document: http://www.sans.org/y2k/DDoS.htm

13 connect to 515 from inside 3325 0.5%

Why are MY.NET machines printing to external IP's? Port 515 is presumably the LPD printer daemon, which is exploitable. Specifically for Solaris CVE-2001-0353[34] and other OSes[35]. It is suggested that the University block port 515 in and out at border.

14 TCP SRC and DST outside network 1920 0.3%

This is bad traffic. Check routers for incorrect parameters. 192.168 private addresses are also seen on the University network. A suggestion is to have University network engineering track down hosts and identify users. Investigate sources of private IP addresses.

15 High port 65535 udp - possible Red Worm – traffic 1848 0.3%

This is another variant of the Code Red worm on UDP. Standard procedures for University risk mitigation include operating system patches, and installation of IIS tools to prevent exploitation. Queries reveal MY.NET IP addresses as the source of alerts. Ports 625 and 655 are the two destination ports for the alerts. Suggestion: Isolate and clean infected MY.NET hosts.

16 connect to 515 from outside 1757 0.3%

There are three source IP addresses responsible for generating these alerts, of which 131.118.229.7 has generated 1602 events targeting MY.NET.24.15. This is direct targeting and depending on the configuration and purpose of MY.NET.24.15, the host may be in danger. The other source IP addresses of 211.22.200.245 and 81.0.145.118 together generated alerts on 148 distinct

MY.NET destination IP addresses. This may be more of a probing type event rather a brute force event.

17	IDS552/web-iis_IIS ISAPI Overflow ida nosize	1662	0.3%
18	IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize	863	0.1%

These alerts are similar to that observed in another student practical[36] which suggests this alert triggered on CERT® Incident Note IN-2001-09 Code Red II[37]. No source IP addresses appear for this event appear to originate from MY.NET hosts, therefore it may not have exploited any University MY.NET hosts. The University should patch and rebuild any infected hosts if this exploit attempt succeeds.

The remaining alerts 19-64 consist of a variety of potentially damaging traffic which should not be allowed to roam freely on the University network, but often does. For example, alerts 32, 42, 43, and 56 are all some manifestation of TFTP server accesses either internal or external. The tftp protocol is ripe with exploits, and should be limited on the campus network as a whole.

Alerts 23, 24, 25 are all examples of protocols which should be configured correctly: SNMP, and RPC. SNMP[38] is excellent when properly implemented for network and infrastructure management purposes, but improperly engineered solutions become a liability. SNMP can be used for reconnaissance[39] or denial of service attacks[40] or system compromise[41]. RPC vulnerabilities are present in many operating systems and are being actively exploited today as can be seen in CERT® Advisory CA-2003-19[42]. The University Security Directorate (if present) should consider researching best practices for securing machines across campus. An excellent reading is from SANS FAQs is “IDS Evasion and Denial of Service Using RPC Design Flaws” by Joseph (Randy) Taylor[43].

Alerts 26, 52, 62, and 63 are all some manifestation of Internet Relay Chat mischief such as killing IRC daemons or types of flooding IRC attacks. A good starting place for reading is http://johoho.eggheads.org/eggdrop/attacks.htm. Often compromised machines are running IRC servers for intruders. Hacked machines at our own University have been found running IRC servers.

The University in question needs to address the top ten problems at the very least before staff can consider spending resources investigating events that comprise less than 5% of the total alerts generated for this time period.

TOP TEN ALERT EXTERNAL SOURCE ADDRESSES

This table below represents the top 10 non MY.NET source address from the alert table. Where possible Whois information from http://ws.arin.net/cgi-bin/whois.pl is truncated to display the following parameters if available:

OrgName, OrgID, City, StateProv, Country, NetRange, CIDR, NetType, Comments, RegDate and Updated date.

RANK	***SOURCE IP***	***COUNT***
1	24.84.205.243	53750
2	68.54.93.211	22311
3	142.26.120.7	20538
4	195.5.55.32	15681
5	172.176.163.241	10817
6	169.254.45.176	10364
7	217.88.160.45	6615
8	172.180.87.233	6571
9	193.41.146.24	5270
10	131.118.254.130	3755

1) IP **24.84.205.243** in 53750 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    Shaw Communications Inc.
OrgID:      SHAWC
City:       Calgary
StateProv:  AB
Country:    CA
NetRange:   24.80.0.0 - 24.87.255.255
CIDR:       24.80.0.0/13
NetType:    Direct Allocation
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2001-07-12
Updated:    2003-06-20
```

This IP address was the source of the alert *"High port 65535 tcp - possible Red Worm – tr"* and targeted MY.NET.82.36. It is recommended that this machine be examined for any sign of compromise. If not already present, the target machine should be at the latest patch level, with maintained and updated Antivirus software, configured with the IIS Lockdown Tool 2.1, and some sort of host based file integrity package similar to TripWire[44].

2) IP **68.54.93.211** in 22311 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
Comcast Cable Communications, Inc. JUMPSTART-1 (NET-68-32-0-0-1)
                      68.32.0.0 - 68.63.255.255
Comcast Cable Communications, Inc. BALTIMORE-A-4 (NET-68-54-80-0-1)
                      68.54.80.0 - 68.54.95.255
```

This IP address was a large contributor to alert event number 5 – MY.NET.30.4 and this IP address may be blocked at the border to halt the flow of traffic to MY.NET.30.4. The University's actions depend on the importance of the target machine utilizing the severity equation from above: severity = (criticality + lethality) – (system countermeasures + network countermeasures. Further information could not be gleaned from OOS or SCAN logs.

3) IP **142.26.120.7** in 20538 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    British Columbia Systems Corporation
OrgID:      BCSC
City:       Victoria
StateProv:  BC
Country:    CA
NetRange:   142.26.0.0 - 142.26.255.255
CIDR:       142.26.0.0/16
Parent:     NET-142-0-0-0-0
NetType:    Direct Assignment
RegDate:    1991-05-13
Updated:    1998-09-16
```

This IP address was noted in the 6th alert SYN-FIN scan. There are 20701 scan events from this source IP address directed at destination port 21. The query results from the scan table indicate that 142.26.120.7 generated scan events against 20568 distinct 130.85.x.x hosts. This is an indication that the intent is to find an FTP server for potentially illicit behavior. The University should reference the SANS paper *"FTP and the Warez Scene"* by Shelli Crocker[45] which provides a starting point for discussion.

4) IP **195.5.55.32** in 15681 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    RIPE Network Coordination Centre
OrgID:      RIPE
City:       Amsterdam
Country:    NL
NetRange:   195.0.0.0 - 195.255.255.255
CIDR:       195.0.0.0/8
NetName:    RIPE-CBLK3
NetHandle:  NET-195-0-0-0-1
NetType:    Allocated to RIPE NCC
Comment:    These addresses have been further assigned to users in
Comment:    the RIPE NCC region. Contact information can be found in
Comment:    the RIPE database at http://www.ripe.net/whois
RegDate:    1996-03-25
Updated:    2003-04-25
```

This IP address was also noted in the 6th alert "SYN-FIN scan". There are 15723 scan events from this source IP address directed at destination port 21. The query results from the scan table indicate that 195.5.55.32 also generated scan events against 15723 distinct 130.85.x.x hosts. OOS files were filled with corresponding events for example:

```
07/12-16:23:01.741346 195.5.55.32:21 -> MY.NET.1.135:21TCP TTL:22 TOS:0x0 ID:39426
IpLen:20 DgmLen:40 ******SF Seq: 0x5F16F9B1  Ack: 0x3E903C6  Win: 0x404  TcpLen: 20
```

We note the TCP source and destination ports are 21, the TOS is zero, and the ID is 39425 with both SYN and FIN flags set. Including the window size of 1028 (0x404) a makes it very likely that the attacker is using Synscan[46] or a tool build around its source[47]. Excellent information discussing this is available from Terry Bidwell on the SANS site[48].

5) IP **172.176.163.241** in 10817 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    America Online
OrgID:      AOL
City:       Dulles
StateProv:  VA
Country:    US
NetRange:   172.128.0.0 - 172.191.255.255
CIDR:       172.128.0.0/10
NetHandle:  NET-172-128-0-0-1
Parent:     NET-172-0-0-0-0
NetType:    Direct Allocation
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2000-03-24
Updated:    2002-08-09
```

This IP address was responsible for 10814 alert events in relation to event seven *“EXPLOIT x86 NOOP”* and the IP address is not seen in the OOS files or scans files. Since approximately 18 MY.NET hosts were the target of this alert, the University should take action to ascertain the level of the target hosts, and tackle any outstanding issues.

6) IP **169.254.45.176** in 10364 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    Internet Assigned Numbers Authority
OrgID:      IANA
City:       Marina del Rey
StateProv:  CA
Country:    US
NetRange:   169.254.0.0 - 169.254.255.255
CIDR:       169.254.0.0/16
NetHandle:  NET-169-254-0-0-1
Parent:     NET-169-0-0-0-0
NetType:    IANA Special Use
Comment:    Please see RFC 3330 for additional information
```

This is to be expected since according to the RFC 3330[49]:

169.254.0.0/16 - This is the "link local" block. It is allocated for communication between hosts on a single link. Hosts obtain these addresses by auto-configuration, such as when a DHCP server may not be found.

There was no information for this IP address returned from the OOS files, and scan files. The IP was event three in reference to the *“SMB Name Wildcard”* in the alert files. Microsoft hosts assign this range automatically when attempting to auto configure the network settings. This is called Automatic Private IP Addressing (APIPA) and indicates that the host is most likely looking for a WINS server, hence the port 137 traffic. It seems the host is attempting to have a NetBIOS name mapped to an IP address[50]. This host’s network jack should be disabled and the host removed from the network to be properly configured.

7) IP **217.88.160.45** in 6615 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    RIPE Network Coordination Centre
OrgID:      RIPE
City:       Amsterdam
Country:    NL
NetRange:   217.0.0.0 - 217.255.255.255
CIDR:       217.0.0.0/8
NetHandle:  NET-217-0-0-0-1
NetType:    Allocated to RIPE NCC
Comment:    These addresses have been further assigned to users in
Comment:    the RIPE NCC region. Contact information can be found in
Comment:    the RIPE database at http://www.ripe.net/whois
RegDate:    2000-06-05
Updated:    2003-04-25
```

This IP address was responsible for 6615 alert events in relation to event seven *“EXPLOIT x86 NOOP”* and the IP address is not seen in the OOS files or scans files. Since approximately 18 MY.NET hosts were the target of this alert, the University should take action to ascertain the level of the target hosts, and tackle any outstanding issues.

8) IP **172.180.87.233** in 6571 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl

```
OrgName:    America Online
OrgID:      AOL
City:       Dulles
StateProv:  VA
Country:    US
NetRange:   172.128.0.0 - 172.191.255.255
CIDR:       172.128.0.0/10
NetName:    AOL-172BLK
NetType:    Direct Allocation
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2000-03-24
Updated:    2002-08-09
```

The IP address 172.180.87.233 appears in 58002 records from the scans files imported into the database, but it does not appear in any OOS files. The host generated this many scan events because it appears to be attempting a SYN to destination port 80 to 24178 distinct hosts on the 130.85 network. It seems to be some reconnaissance for web server information which may reveal exploitable systems. It is recommended that the University configure ANY server to reveal as little information possible via: disabling welcome banners, removing modules that reveal configuration information (Apache mod_info[51]) and disabling servlets such as Snoop[52] if enabled in the default installation.

9) IP **193.41.146.24** in 5270 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    RIPE Network Coordination Centre
OrgID:      RIPE
City:       Amsterdam
Country:    NL
NetRange:   193.0.0.0 - 193.255.255.255
CIDR:       193.0.0.0/8
NetType:    Allocated to RIPE NCC
Comment:    These addresses have been further assigned to users in
Comment:    the RIPE NCC region. Contact information can be found in
Comment:    the RIPE database at http://www.ripe.net/whois
RegDate:    1992-08-12
Updated:    2003-04-25
```

Since there is no data in the OOS logs for 193.41.146.24, the alert log must suffice. The IP address 193.41.146.24 beings generating alerts on July 12th at timestamp 05:56:24.596859 to destination IP MY.NET.100.165 of description "*CS WEBSERVER – external web traffic".* We are able to ascertain that 4511 more alerts indicating traffic between 193.41.146.24 and MY.NET.100.165 where generated up to timestamp 12:42:42.038227 on July 12th. At timestamp 21:35:09.463489 on July 12th MY.NET.100.165 begins generating alerts of description "*High port 65535 tcp - possible Red Worm – tr"* and *"Possible trojan server activity"* to external IP addresses. The host has been infected. July 12th as shown previously in alert number one was an explosive day for bad web traffic.

10) IP **131.118.254.130** in 3755 alert events yields the following Whois information from http://ws.arin.net/cgi-bin/whois.pl:

```
OrgName:    University of Maryland
OrgID:      UNIVER-270
City:       Adelphi
StateProv:  MD
Country:    US
NetRange:   131.118.0.0 - 131.118.255.255
CIDR:       131.118.0.0/16
NetType:    Direct Assignment
RegDate:    1988-11-15
```

Updated: 1998-11-24

Since this source IP address seems to be attempting to contact MY.NET.24.8 on destination port 119. It seems that 131.118.254.130 resolves to FDQN of news.ums.edu. It is possible that the University System of Maryland's news server was mistakenly attempting to contact MY.NET.24.8. The OOS log files do not yield any information on this traffic. Therefore it is plausible to conclude that alert event seven EXPLOIT x86 NOOP is actually an attempt to transfer binary data (i.e. news).

TOP TEN ALERT TARGET ADDRESSES

In this section the initial framework for the link graph begins to crystallize. Each event will be discussed briefly as most of the targeted addresses below. Only those IP address not previously highlighted will be discussed below.

EVENT	***SOURCE IP***	***COUNT***
1	MY.NET.100.165	160427
2	24.84.205.243	78661
3	MY.NET.30.4	57441
4	MY.NET.82.36	53752
5	210.192.111.73	14611
6	211.147.7.47	9695
7	MY.NET.30.3	9090
8	MY.NET.137.7	5232
9	MY.NET.24.8	4002
10	MY.NET.86.19	3132

1) IP **MY.NET.100.165** was previously highlighted and appears to be a web server which has been compromised and subsequently has become infected.

2) IP **24.84.205.243** appears in addition to being was the source of the alert *"High port 65535 tcp - possible Red Worm – tr"* targeting MY.NET.82.36, but also is a target itself.

3) IP **MY.NET.30.4** was previously highlighted and appears to be the object of unusual activity.

4) IP **MY.NET.82.36** was previously highlighted and appears to offer port 80 which was targeted as MY.NET.100.165.

5) IP **210.192.111.73** was targeted in 14611 alert events by MY.NET.198.172 with a description of "*spp_http_decode: IIS Unicode attack detected*". Information from http://ws.arin.net/cgi-bin/whois.pl reveals:

inetnum: 210.192.96.0 - 210.192.127.255

```
netname:     CHINANETCENTER
descr:       ChinaNetCenter Ltd.
descr:       Internet Service Provider
descr:       China
country:     CN
admin-c:     XW53-AP
tech-c:      KJ17-AP
mnt-by:      APNIC-HM
mnt-lower:   MAINT-CHINANETCENTER
remarks:     Transfer from AUNET-AP
changed:     hm-change@apnic.net 20020827
status:      ALLOCATED PORTABLE
source:      APNIC
```

It is unfortunate that the University's machine is targeting other systems. It is unknown what the University's liability would be should there be any damages associated with an attack. It is imperative that the University formulate aggressive information security policies to mitigate risk. This MY.NET.198.172 needs to be further investigated as to why it is generating this traffic to a target in China.

6) IP **211.147.7.47** was targeted by 11 distinct MY.NET hosts generating alerts with a description of *"spp_http_decode: IIS Unicode attack detected*". Information from http://ws.arin.net/cgi-bin/whois.pl reveals:

```
inetnum:     211.147.0.0 - 211.147.7.255
netname:     DYNEGY-COMMUNICATION
descr:       DYNEGY-COMMUNICATION
descr:       CO.LTD
descr:       BEIJING
country:     CN
admin-c:     PP40-AP
tech-c:      SD76-AP
mnt-by:      MAINT-CNNIC-AP
changed:     hui_zh@sina.com 20011112
status:      ALLOCATED PORTABLE
source:      APNIC
```

Once again, it is unfortunate that the University's machine is targeting other systems. This coordinated attack by 11 distinct MY.NET host machines to a machine purportedly in China is not a good sign of the state of the University's network and host infrastructure.

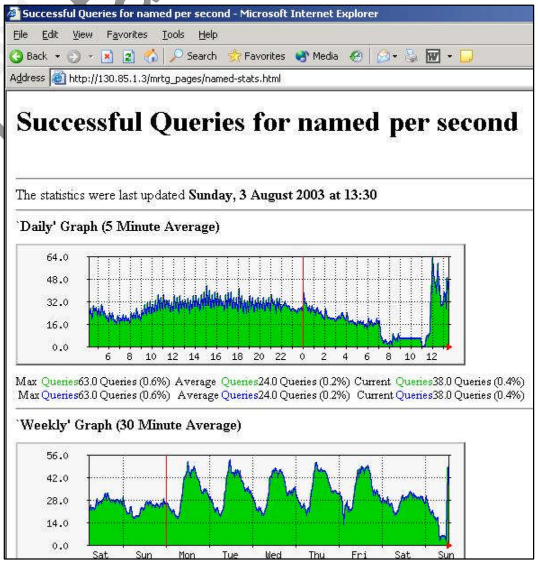
TOP TEN SCANS AND SCAN DESTINATION PORTS

In this section the scan logs were all imported into a FoxPro table for further analysis using SQL statements to extract relevant information. Each scan event will be discussed in relation to the destination port of the scan.

EVENT	***COUNT OF EVENTS***	***SCAN DESCRIPTION***	***SCAN DESTINATION PORT***
1	3597877	UDP scan (Externally-based)	53
2	1969017	UDP scan (Externally-based)	137
3	1154965	SYN scan (Externally-based)	80
4	825919	SYN scan (Externally-based)	1214
5	238297	UDP scan (Externally-based)	6257
6	192807	FIN scan (Externally-based)	1214
7	181265	SYN scan (Externally-based)	21
8	154071	SYN scan (Externally-based)	1730
9	149513	SYN scan (Externally-based)	3389
10	134493	SYN scan (Externally-based)	445

1) 3597877 UDP scan events were detected to destination port 53. It is common to see many scan events from DNS servers labeled as such and subsequently be false alarms. For example in the snort.conf configuration file it is possible to uncomment *“portscan-ignorehosts”* to ignore TCP SYN and UDP "scans" from DNS. The source IP address of 130.85.1.3 appears to be responsible for the majority of this activity with 3112927 UDP scan events to destination port 53, followed by 130.85.1.4 with 470152 UDP scan events.

Directing a web browser to http://130.85.1.3 provides a web page with information related to named performance. If the University is going to track performance tuning information with MRTG and provide web access to these results, the site should have access controls placed upon it. There are numerous vulnerabilities associated with DNS (CA-2002-31)[53] and this information would provide an attacker with information as to how highly utilized the target is (criticality to infrastructure) and effectiveness of such an attack.


Successful Queries for named per second - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
64.0
48.0
32.0
16.0
0.0
6 8 10 12 14 16 18 20 22 0 2 4 6 8 10 12
56.0
42.0
28.0
14.0
0.0
Sat Sun Mon Tue Wed Thu Fri Sat Sun

http://130.85.1.3/mrtg_pages/named-stats.html

Successful Queries for named per second

The statistics were last updated **Sunday, 3 August 2003 at 13:30**

`Daily' Graph (5 Minute Average)

Max Queries63.0 Queries (0.6%) Average Queries24.0 Queries (0.2%) Current Queries38.0 Queries (0.4%)
Max Queries63.0 Queries (0.6%) Average Queries24.0 Queries (0.2%) Current Queries38.0 Queries (0.4%)

`Weekly' Graph (30 Minute Average)

2) 1969017 UDP scan events were detected to destination port 137. The top ten source IP addresses are all hosts in 130.85.97.x subnet and account for 1781996 of the UDP scan events targeted at port 137. DNS lookups of several IP addresses indicate that machines are part of some dial up pool, for example the FQDN of 130.85.97.49 is ppp-049.dialup.umbc.edu thereby making it plausible that these are Windows based machines attempting to resolve hostnames via NetBIOS.

Without additional correlating evidence from OOS log files and no other detailed packet data, the scans could be indicative of some possible NetBIOS exploits. The SANS FAQ[54] detailing port 137 scans does not rule out worm activity (network.vbs). The University security directorate should recommend guidelines akin to CERT® [55] Coordination Center documents which recommend Windows users block inbound and outbound traffic to TCP and UDP ports 135, 137, 139 and UDP port 138.

3) 1154965 SYN scans directed at destination port 80 initially may not be of concern, but queries of the scan logs indicate ominous behavior. Specifically source IP address 172.80.87.223 generated 57999 scan alerts. The same source IP was involved in 6615 alert events related to *"EXPLOIT x86 NOOP"*. The target of these scans, if vulnerable to a SYN flood attack, can be overwhelmed and a denial of that web service would occur. In this particular case, source IP 172.80.87.223 was performing reconnaissance.
No overwhelming amount of SYN flagged packets over a small time period were issued against a particular host. Nonetheless the University should employ some strategy to guard against the possibility of a SYN flood by incorporating the defensive recommendations below.

4) 825919 SYN scans directed at destination port 1214 are of concern for two reasons. Port 1214 is the service destination for several peer to peer file sharing applications. KAZAA, Morpheous, and Grokster utilize TCP and UDP protocols on this port[56]. Primarily, University information security professionals should formulate a policy because worms, exploits, and 'Spyware' are prevalent with P2P. Finally, P2P applications open the University and its community to litigation based on the Digital Millennium Copyright Act. The University is encouraged to investigate bandwidth monitoring/packet shaping technologies to curb this behavior and mitigate risk.

5) 238297 UDP scans directed at destination port 6257 are a sign of additional P2P activity caused by the WinMX application which TCP 6699 and UDP 6257 ports by default to establish connections with other users on the WinMX Peer Network[57].

6) 192807 FIN scans directed at destination port 1214 was detailed above in event number four.

7) 181265 SYN scans directed at destination port 21 is of concern. The table below lists the top 5 source IP addresses of this scan event directed to port 21.

	SOURCE IP ADDRESS	***COUNT OF EVENTS***	***PERCENTAGE OF EVENTS***
1	213.39.155.17	48742	27%
2	217.224.251.17	47496	26%

3	128.186.55.170	34124	19%
4	213.23.163.11	25963	14%
5	217.219.130.91	17403	10%

Unfortunately there is no data from the OOS logs to augment the information available in the scan logs. The alert event logs do provide information which suggests this activity is unwelcome in nature. Alert description such as "*Notify Brian B. 3.56 tcp*", "*MY.NET.30.4 activity*", "*External FTP to HelpDesk MY.NET.53.29*", "*External FTP to HelpDesk MY.NET.70.49*", "*External FTP to HelpDesk MY.NET.70.50*" *and* "*CS WEBSERVER - external ftp traffic*" imply that FTP services are not permissible for resources external to the University. If possible these IP address should be blocked at the border and the system administrators be contacted to ascertain the nature of the contact.

8) 154071 SYN scans directed at destination port 1730 is an indication of network game play which may consume University network bandwidth. According to cotse.com port search TCP and UDP port 1730 is used for roketz[58]. The external IP addresses involved in these scans appeared to be searching for roketz based servers. The University needs to formulate some policy for the treatment of illegitimate traffic not related to research or any sanctioned endeavor supported by the University. An example of acceptable traffic would be video streaming which may require tremendous bandwidth. If the University was streaming commencement ceremonies to satellite campuses, it would most likely be acceptable. LAN/WAN gaming is most likely not officially authorized.

9) 149513 SYN scans directed at destination port 3389 should raise the eye of any University network services staff member who is familiar with Microsoft technologies. The primary source IP triggering this event is 68.163.94.35. Output from http://ws.arin.net/cgi-bin/whois.pl results in the following information:

```
Verizon Internet Services VIS-68-160 (NET-68-160-0-0-1)
                          68.160.0.0 - 68.163.255.255
Seniors Coalition VZ-SNRSCLTN-1 (NET-68-163-94-32-1)
                          68.163.94.32 - 68.163.94.63
```

There was no information available from OOS logs to substantiate the reconnaissance scan from 68.163.94.35. This port is the default Windows Terminal Services port for client connections. According to Microsoft security bulletin MS01-040[59] there is a potential to affect server performance and potentially stop the target machine from responding altogether. It is unknown if this is legitimate traffic from some offsite faculty or staff member. Terminal Services access should be integrated into a University wide infrastructure design which includes VPN access.

10) 134493 SYN scans directed at destination port 445 are another opportunity for the University to mitigate risk. This port is now used by Windows 2000 based operating systems for running Server Messaging Block protocol over TCP[60].

Inbound port 445 should be blocked at the University's border routers given the known vulnerabilities:

CAN-2002-0597[61] LANMAN service on Microsoft Windows 2000 allows remote attackers to cause a denial of service (CPU/memory exhaustion) via a stream of malformed data to microsoft-ds port 445.
CAN-2002-0283[62] Windows XP with port 445 open allows remote attackers to cause a denial of service (CPU consumption) via a flood of TCP SYN packets containing possibly malformed data.

The above scanning activity should highlight the need for the University to pursue an aggressive information security policy as recommended in the defensive recommendations section of the practical assignment.

OOS TOP TEN DESTINATION PORTS

OOS logs were referenced whenever a specific IP address was present in the scans logs or alert logs. All log data available facilitates the creation of a strategy for the University to mitigate risk and properly tune computing resources to further its mission. More than 5 external IP addresses have been researched in previous alert and scan analysis discussions. The criterion for additional data mining in the OOS logs was chosen to be the top ten destination ports. The table below lists additional ports which deserve closer scrutiny and may provide additional insight into the overall network traffic transiting the University networks.

#	***DESTINATION PORT***	***COUNT OF EVENTS***
1	21	23799
2	25	9553
3	80	1732
4	110	617
5	113	265
6	3456	183
7	4662	141
8	81	51
9	8080	43
10	6881	41

The OOS logs indicate additional popular common ports that may have not been mentioned in scan log analysis and alert log analysis must be included in the overall University information security recommendations. These ports will be discussed below.

1) Destination port 21 – previously discussed & referenced from OOS logs.

2) Destination port 25 – SMTP is an important protocol to consider with respect to information security. Email is a very popular service for the University community and querying the OOS logs reveals the 30 distinct MY.NET IP addresses were targeted on destination port 25. An excellent source of

information is the student practical from Stephanie Alarcon[63]. The following MY.NET hosts from the OOS logs seem to be functioning as mail hosts:

SOURCE IP ADDRESS	***COUNT EVENTS***
MY.NET.25.73:25	1753
MY.NET.25.70:25	1739
MY.NET.25.71:25	1704
MY.NET.25.69:25	1696
MY.NET.25.72:25	1632

The activities surrounding these hosts should be investigated thoroughly. Active monitoring of mail logs in addition to other operating system logs is necessary in order to discern if any of this traffic is truly nefarious (SPAM).

4) Destination port 110 – POP3 represents a risk for the University community. A search via http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=POP3 returned 63 potential matching candidates for various exploits and denial of service issues. POP3 accepts clear text connections by default on TCP port 110. Passing clear text passwords over the network is not acceptable, and the University should investigate the alternatives such as secure IMAP or secure POP3 on TCP port 993 for imaps or TCP port 995 for pop3s.

5) Destination port 113 – Authd represents a potential component of the mail service infrastructure. In particular, Sendmail attempts to utilize[64] authd to verify the connecting sender's or recipient's address. Inspection of these OOS logs indicates behavior associated with the transfer of mail.

6) Destination port 3456 – It may appear that this is the VAT port TCP for the X11-based audio teleconferencing tool or the 'Terror Trojan'[65]. Examination of the OOS logs clarifies this traffic as Kazaa as seen from a sample packet below:

```
07/13-00:40:30.781566 148.63.120.213:36917 -> MY.NET.69.217:3456
TCP TTL:115 TOS:0x0 ID:20635 IpLen:20 DgmLen:448 DF
****P*** Seq: 0x90BC0E0A  Ack: 0x0  Win: 0x2000  TcpLen: 20
47 45 54 20 2F 2E 68 61 73 68 3D 63 62 30 39 34  GET /.hash=cb094
38 35 33 32 30 32 64 32 39 32 61 32 33 65 33 36  853202d292a23e36
32 61 62 64 34 32 62 32 30 61 39 64 31 65 31 66  2abd42b20a9d1e1f
30 32 63 20 48 54 54 50 2F 31 2E 31 0D 0A 48 6F  02c HTTP/1.1..Ho
73 74 3A 20 31 33 30 2E 38 35 2E 36 39 2E 32 31  st: MY.NET.69.21
37 3A 33 34 35 36 0D 0A 55 73 65 72 41 67 65 6E  7:3456..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 63 6F 6E 66 75 73 65 64 63  rname: confusedc
68 69 63 6B 65 6E 0D 0A 58 2D 4B 61 7A 61 61 2D  hicken..X-Kazaa-
4E 65 74 77 6F 72 6B 3A 20 4B 61 5A 61 41 0D 0A  Network: KaZaA..
58 2D 4B 61 7A 61 61 2D 49 50 3A 20 31 39 32 2E  X-Kazaa-IP: 192.
31 36 38 2E 30 2E 37 3A 31 30 38 30 0D 0A 58 2D  168.0.7:1080..X-
4B 61 7A 61 61 2D 53 75 70 65 72 6E 6F 64 65 49  Kazaa-SupernodeI
```

```
50 3A 20 32 34 2E 31 38 37 2E 32 30 31 2E 31 34  P: 24.187.201.14
32 3A 33 36 31 38 0D 0A 52 61 6E 67 65 3A 20 62  2:3618..Range: b
79 74 65 73 3D 31 32 37 31 30 33 36 35 35 2D 31  ytes=127103655-1
32 37 31 34 31 38 38 37 0D 0A 43 6F 6E 6E 65 63  27141887..Connec
74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 58 2D 4B  tion: close..X-K
61 7A 61 61 2D 58 66 65 72 49 64 3A 20 31 34 39  azaa-XferId: 149
37 36 37 35 39 0D 0A 58 2D 4B 61 7A 61 61 2D 58  76759..X-Kazaa-X
66 65 72 55 69 64 3A 20 62 75 6C 71 54 6B 65 4A  ferUid: bulqTkeJ
4A 6E 36 64 4A 58 31 64 56 46 6C 66 65 70 34 4D  Jn6dJX1dVFlfep4M
55 47 4B 57 41 72 32 41 44 33 53 69 76 72 6E 65  UGKWAr2AD3Sivrne
50 62 77 3D 0D 0A 0D 0A                          Pbw=....
```

Once again, the network bandwidth is being utilized for questionable purposes. If not already incorporated into the University's acceptable use policy, a policy should be drafted and adopted to address utilization of P2P file sharing programs to exchange copyrighted materials on resident student networks.

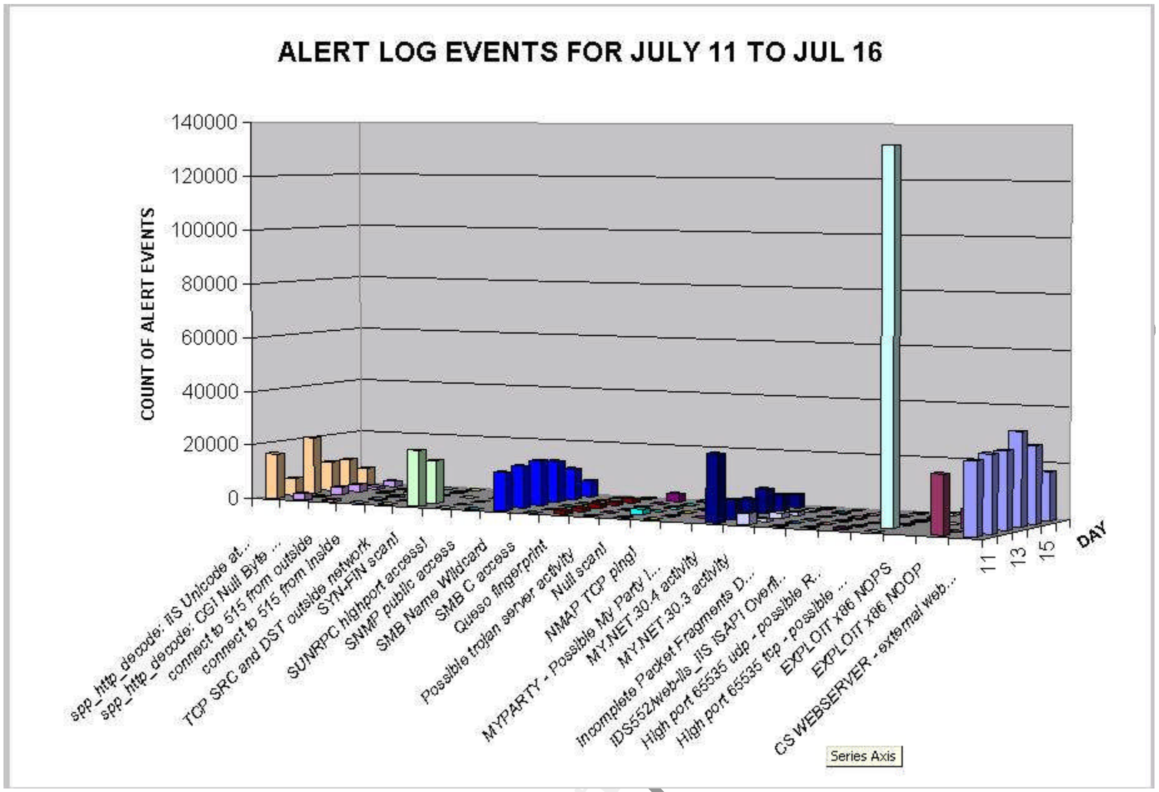
DEFENSIVE RECOMMENDATIONS TO THE UNIVERSITY

This University has some severe issues related to network security, but none of the issues that plague this University are insurmountable. Since it is not uncommon for higher education institutions to have similar problems, it is suggested that the University participate in Educause. In particular it is recommended that the University in question participate in any conferences or seminars related to information security issues in higher education institutions.

After pouring over the almost 550 megabytes of log data for the 5 days in question, it is imperative that the University address these key issues:

1) The rampant worm and virus activity that most likely plagues faculty, staff and student workstations.
2) University-wide issues related to copyright infringement over P2P networks.
3) Adequate hardening of basic infrastructure services responsible for facilitating the educational process of the University's customers: the students.

Below is a graphic which further highlights the situation facing the University in question.



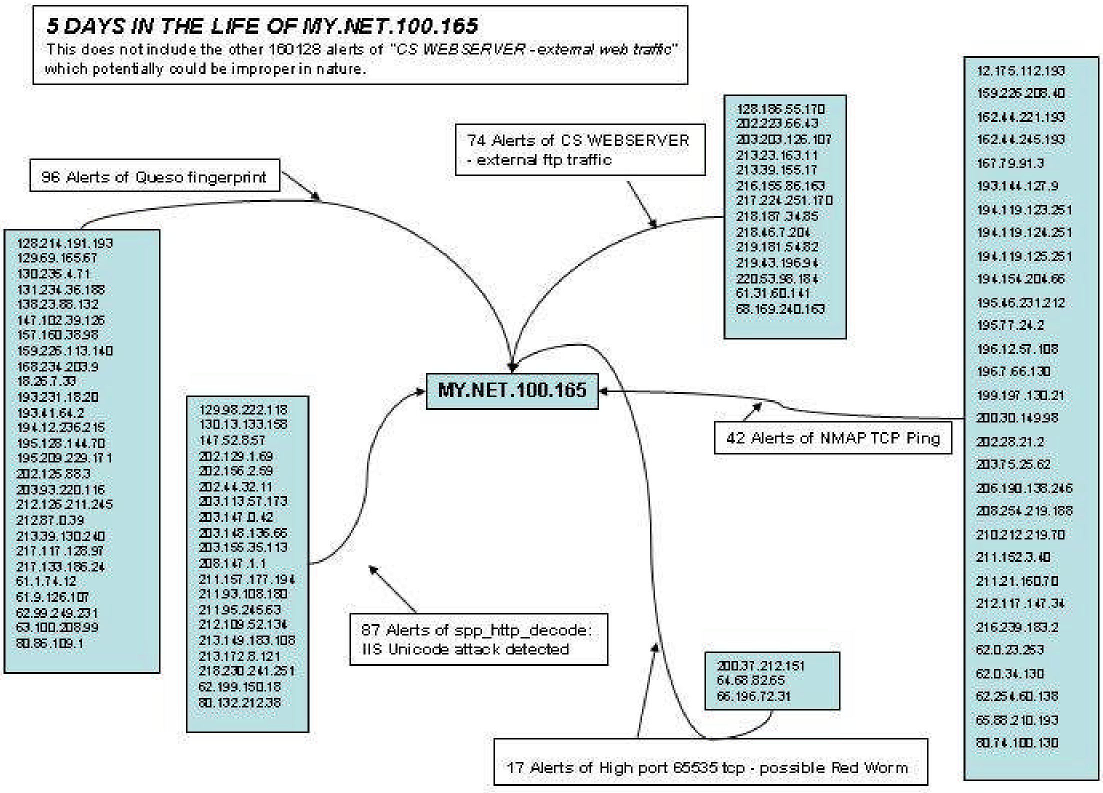
The graphic above is a sample of the data from the alert logs. This is not all of the alerts for the sample time period of July 11th to July 16th, but approximately 23 alerts plotted versus date. The objective of this graph is to visually demonstrate that in addition to the occasional spikes in questionable network traffic, there is a steady undercurrent of activity which pervades the network infrastructure. University information technology staff must realize that their networks are exposed daily to a large spectrum of harmful traffic.

It is not feasible to suggest that the University simply 'stand up' a firewall device or devices (in a load balanced configuration) at the boarder to handle all traffic. Universities in general are somewhat unique in that they are a mix of various computing models. On some subnets workstations and servers have free unfettered access to the Internet much like an Internet service provider. There are also highly secured network subnets with machines that store information of the highest sensitivity such as human resources data, student account information, and critical financial data. Incorporated into this mixture are the semi-secured critical infrastructure services utilized campus-wide such as email, LDAP, Kerberos, web services, and possibly 'shared' file space (AFS). I label these as semi-secured since there is still the potential for abuse due to the nature of the service. You must trust that not all students are interested in 'bombing' the email server, but simply wish to utilize the service for academic pursuits.

It is also not plausible to suggest that each business unit or college must be responsible for its own defense, thereby creating multiple island-like domains of highly secure systems, while stranding some users (faculty/staff/students) in a sea of insecurity. Without complete details of the sample University's infrastructure, the following precautions should be taken by University staff to mitigate risk. These suggestions are listed below in an arbitrary order since implementation costs depend on too many variables unknown to myself at this time.

- Initially it is suggested that some statement be included where possible on all University server machines that discloses to users that they are using University services and their activities may be subject to monitoring. University critical infrastructure resources should be subjected to the commonly known principles of least privilege and compartmentalizing of information[66].
- The University should acquire site volume licensing for personal virus protection software as well as desktop firewall software for faculty, staff and students.
- Utilizing host based firewalls for servers is recommended in situations where only where configuration is relatively simple and standardized, any marginal increase in security is offset by the potential for misconfiguration and endless tweaks of rule sets to accommodate infrastructure changes.
- The university should focus on technologies from current hardware and software vendors that facilitate basic security practices such as the implementation of a patch management tools, and tools for building pre-configured operating systems from images already deemed acceptable by the information security department.
- The University should investigate some technology and policy to curb the P2P activity which exposes the University to potential litigation for copyright infringement.
- If central computing services are incorporated into a datacenter facility, the University should consider a comprehensive design including firewall devices, VPN technologies and bandwidth management.
- Implementation of a distributed set of Intrusion Detection Systems to be staffed by a team of dedicated individuals. My experience in higher education has shown me that the role of administrator, analyst and engineer are often duties given to one individual. Intrusion detection is not a part time job given the volume of traffic at this University.

In order to demonstrate the need to prevent, detect and halt malicious network activity, I chose to examine the top talker centered on the destination address of MY.NET.100.165. Examination of all alerts across all machines on MY.NET subnets provides an excellent picture of the level of security for University, but it is necessary to focus on a specific machine to understand how the above defensive recommendations will affect the operational stability of campus wide services.



Each of these events that occurred over the 5 day period attempted to gather information for reconnaissance purposes or the events were simply outright attacks against My.NET.100.165. If I may assume that MY.NET.100.165 is a Windows 2000 Server machine running IIS 5, has an operating system which is current on its patch level, and is configured with the UrlScan Security tool there is a possibility that the Code Red Worm and Unicode attacks will not succeed.
If the machine is running Tripwire to function as a host based intrusion detection system alerting University staff with daily reports of file and directory changes, then there is a possibility that any external ftp traffic which succeeds in uploading files will be discovered. If a host based firewall or router ACLs are configured to block traffic based on business rules (no external traffic), then the machine will be unreachable by those with least privilege. If there are adequate resources dedicated to the practice of intrusion detection at the University in question, these resources will function as a feedback mechanism continually improving and refining the infrastructure.

APPENDIX - CODE

csv.pl

```
# Name: csv.pl
# Reads in a Snort -A Fast style alert log which for some
# reason wasn't generated as CSV, and make it as such.
#
# Usage: csv.pl infile [outfile]
unless ($ARGV[0]) {
print "Need an input file!\n";
        die "(Hint: go to http://www.research.umbc.edu/~andy and get one)\n";
        }
unless ($ARGV[1]) {
        $outfile = "$ARGV[0].csv";
} else { $outfile = "$ARGV[1]";
}
open(INFILE,"$ARGV[0]") || die "Can't open $ARGV[0] for reading!\n";
open(OUTFILE,">$outfile") || die "Can't open $ARGV[1] for writing!\n";

        print "Transforming $ARGV[0] into $outfile.\n";
                print "Just a moment.";
        @calendar=qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec);

                while (<INFILE>) {
                        next unless /(\w{1,3}\.){2}(\d{1,3}\.\d{1,3})/; # Skip lines missing IPv4 IPs.
                        next if /spp_portscan/; # Skip portscan notifications.
                        chomp;
                                if (/ \[\*\*\] /) { # Alert report.
                ($date_and_time,$alert,$src_and_dst) = split(/\s+\[\*\*\]\s/);
                ($date,$time) = split(/-/,$date_and_time);
                ($month_number,$day) = split(/\//,$date);
                $month = $calendar[$month_number-1];
                ($src,$dst) = split(/\s-\>\s/,$src_and_dst);
                ($src_ip,$src_port) = split(/:/,$src);
                ($dst_ip,$dst_port) = split(/:/,$dst); $snort_entry="ALERT" ;

                        } else { # Scan report.
                                ($month,$day,$time,$src,$arrow,$dst,$alert,$flags) = split;
                        undef $arrow;
                        ($src_ip,$src_port) = split(/:/,$src);
                        $alert = "$alert scan (Internally-based)" if $src_ip =~ /^MY\.NET/;
                        $alert = "$alert scan (Externally-based)" unless $src_ip =~ /^MY\.NET/;
                        ($dst_ip,$dst_port) = split(/:/,$dst); $snort_entry="SCAN" ;
}

        print OUTFILE "$snort_entry,";
        print OUTFILE "$month,$day,$time,$alert,";
        print OUTFILE "$src_ip,";
        print OUTFILE "$src_port" if $src_port;
        print OUTFILE "None" unless $src_port;
        print OUTFILE ",";
        print OUTFILE "$dst_ip";
        print OUTFILE ",";
        print OUTFILE "$dst_port" if $dst_port;
        print OUTFILE "," if $flags;
        print OUTFILE "None," unless $dst_port;
        print OUTFILE "$flags" if $flags;
        print OUTFILE "\n";

                $happydots++;
                        print "." if $happydots % 100 == 0; # if $happydots == 100;
                        print "Just a moment." if $happydots % 46600 == 0;
        }
```

REFERENCES

Note: If the endnotes section of the practical is not considered to be adequate documentation, I respectfully submit a complete listing of endnotes as my references below. The reasoning for both of these sections being included is per the MLA Handbook for Writers of Research Papers, 5th ed., pages 270-285.

Gibaldi, Joseph. MLA Handbook for Writers of Research Papers. 5th ed. New York: Modern Language Association of America, 1999.

Sequeira, Dinesh. “Intrusion Prevention Systems: Security’s Silver Bullet?”. URL: http://www.bcr.com/bcrmag/2003/03/p36.asp

The MITRE Corporation, “Common Vulnerabilities and Exposures List “. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0236

ITSX BV. “Title: Solaris SNMP to DMI mapper daemon vulnerability”. March 2001. URL: http://www.itsx.com/snmpXdmid.html

Entercept Security Technologies. “SEA SNMP - Buffer Overflow and Format String Vulnerabilities in Sun Solaris”. URL: http://www.entercept.com/dr/snmp

Anonymous. “Definition: Buffer Overflow”. URL: http://commons.somewhere.com/buzz/2000/Definition.Buffer.Overfl.html

Noordergraaf, Alex. “JumpStart Technology: Effective Use in the Solaris Operating Environment”.October 2001. URL:http://www.sun.com/presents/discussions/disc-112701/

Sun Microsystems. “Solaris Security Toolkit (JASS) “. URL: http://wwws.sun.com/software/security/jass/

Sun Microsystems. URL: http://sunsolve.sun.com/pub-cgi/show.pl?target=patches/patch-access

Caldera International. “Introduction to the desktop management interface“. URL: http://uw713doc.caldera.com/en/SDK_dmi/Introduction_to_the_desktop_management_interface.html

Caldera International. “The role of the DMI Service Provider”. URL: http://uw713doc.caldera.com/en/SDK_dmi/The_role_of_the_DMI_Service_Provide.html

Sun Microsystems. “Using SNMP With DMI”. Sun Product Documentation. URL: http://docs.sun.com/db/doc/806-2905/6jc3duo48?a=view.

Caldera International. URL: http://uw713doc.caldera.com/en/man/html.3dmi/DmiComponentAdded.3dmi.html

Reed, Darren. “IP Filter”. URL: http://coombs.anu.edu.au/~avalon/ip-filter.html

Zalewski, Michal & Stearns, William. “p0f - passive OS fingerprinting”. URL: http://www.stearns.org/p0f/p0f.fp

Microsoft Corporation. “Windows 2000 QoS Packet Scheduler Sends Packets with Wrong Checksum on Network Adapters That Enable Hardware Checksum”. Microsoft Knowledge Base Article. URL: http://support.microsoft.com/default.aspx?scid=kb;EN-US;270923

Jacobson, V & Braden, R. “TCP Extensions for Long-Delay Paths”. October 1988. URL: http://www.faqs.org/rfcs/rfc1072.html

West, M. McCann, S. “TCP/IP Field Behavior”. Internet-Draft. March 2003. URL: http://www.ietf.org/internet-drafts/draft-ietf-rohc-tcp-field-behavior-02.txt

Unicode Inc. “What is Unicode”.URL: http://www.unicode.org/standard/WhatIsUnicode.html

PhiRo-In. “Exploit the IIS hole using the Echo Style”. URL: http://www.rootshell.be/~doxical/download/docs/win/Exploit-unicode.rtf

Cahoon, Brian. Analyst 0602 SANS GCIA Practical. January 2003. URL: http://www.giac.org/practical/GCIA/Brian_Cahoon.pdf

Beardsley, Tod. Analyst 0525 SANS GCIA Practical. May 2002. URL: http://www.giac.org/practical/Tod_Beardsley_GCIA.doc

CERT Coordination Center. “CERT® Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL”. January 2002. URL: http://www.cert.org/advisories/CA-2001-19.html

Reiter, Michael. “Analysis of the Adoreworm/CodeRed”. 2001. URL:http://www.giac.org/practical/Michael_Reiter_GCFW.zip

PSI.” Microsoft Windows 2000 TCP/IP Configuration Guide”. URL: http://www.support.psi.com/support/common/inet-serv/tcpip/2000/win2k_tcpip.html

Posting. URL: http://www.incidents.org/archives/intrusions/msg01325.html

Sourcefire, INC. “Snort Signature Database”. URL: http://www.snort.org/snort-db/sid.html?sid=648

Securityfocus. FOCUS-IDS Archive. April 2002. URL: http://www.securityfocus.com/archive/96/266455/2002-04-08/2002-04-14/0

Orebaugh, Angela. Analyst 0498 SANS GCIA Practical. March 2002. URL: www.giac.org/practical/Angela_Orebaugh_GCIA.doc

rain.forest.puppy.”Perl CGI problems “. URL:http://www.wiretrip.net/rfp/txt/phrack55.txt

Neohapsis Archives. November 2000. URL: http://archives.neohapsis.com/archives/snort/2000-11/0248.html

McCabe, Mike. Analyst 0577 SANS GCIA Practical. November 2002. URL: http://www.giac.org/practical/GCIA/Mike_McCabe_GCIA.doc

Novell. “Protocols and Ports used by NetWare 5 IP”. June 2001. URL: http://www.novell.com/coolsolutions/netware/features/a_ports_nw5_nw.html

Fyodor. “Remote OS detection via TCP/IP Stack FingerPrinting”. October 1998. URL:http://www.insecure.org/nmap/nmap-fingerprinting-article.txt

ICAT. http://icat.nist.gov/icat.cfm?cvename=CAN-2001-0353

SecurityFocus. “Multiple Vendor LPRng User-Supplied Format String Vulnerability“. URL: http://www.securityfocus.com/bid/1712/discussion/

Yuen, Rick Winkey. Analyst 0435 SANS GCIA Practical. October 2001. URL: http://www.giac.org/practical/Rick_Yuen_GCIA.doc.

CERT Coordination Center. “Code Red II: Another Worm Exploiting Buffer Overflow In IIS Indexing Service DLL “. August 2001. URL: http://www.cert.org/incident_notes/IN-2001-09.html

Harrington D., Presuhn R., Wijnen B. “An Architecture for Describing SNMP Management Frameworks”. January 1998. URL: http://www.faqs.org/rfcs/rfc2271.html

Romananski, James.” Default SNMP Community Strings Set to "public and "private"” from “Using SNMP for Reconnaissance “. URL:http://www.sans.org/resources/idfaq/snmp.php

SecurityFocus. “Microsoft Windows 2000 SNMP Printer Query Denial of Service Vulnerability”. October 2002. URL: http://www.securityfocus.com/bid/6030

SecurityFocus. “hIRIX SNMP Daemon Buffer Overflow Vulnerability” . URL: http://www.securityfocus.com/bid/4421/discussion/

CERT Coordination Center . “CERT® Advisory CA-2003-19 Exploitation of Vulnerabilities in Microsoft RPC Interface”. July 2003. URL: http://www.cert.org/advisories/CA-2003-19.html

Taylor, Randy.” IDS Evasion and Denial of Service Using RPC Design Flaws”. URL:http://www.sans.org/resources/idfaq/rpc_evas.php

Tripwire Inc. URL: http://www.tripwire.com/

Crocker, Shelly.”FTP and the Warez Scene”. URL: http://www.sans.org/rr/papers/60/978.pdf.

Smith, Donald.” Mscan, Sscan and Synscan - The evolution of worm - enabling vulnerability scanners that span two Millenniums.” URL: http://www.whitehats.ca/main/publications/external_pubs/scanner_fingerprints/scanner_fingerprints.html

Thibault, David. Analyst 0287 SANS GIAC Practical. November 2000. URL: www.giac.org/practical/David_Thibault_GCIA.html

Global Incident Analysis Center. November 2000. URL: http://www.sans.org/y2k/111600.htm

IANA. “Special-Use IPv4 Addresses”. September 2002. URL: http://www.faqs.org/rfcs/rfc3330.html

Neohapsis Archives. Posting. September 2000. URL: http://archives.neohapsis.com/archives/incidents/2000-09/0208.html.

Apache Software Foundation.” Apache Module mod_info”. Apache HTTP Server Documentation Project. http://httpd.apache.org/docs-2.0/mod/mod_info.html

SecurityFocus. “Apache Tomcat Snoop Servlet Information Disclosure Vulnerability”. July 2000. URL: http://www.securityfocus.com/bid/1532

CERT® Coordination Center. “CERT® Advisory CA-2002-31 Multiple Vulnerabilities in BIND”. November 2002. URL: http://www.cert.org/advisories/CA-2002-31.html

Alexander, Bruce. “Intrusion Detection FAQ Port 137 Scan”. URL: http://www.sans.org/resources/idfaq/port_137.php

CERT® Coordination Center . “Windows NT Configuration Guidelines”. URL: http://www.cert.org/tech_tips/win_configuration_guidelines.html

Internet Storm Center. Port Reports. URL: http://isc.incidents.org/port_details.html?port=1214

URL: http://winmx.2038.net/winmx/fr-blocked.html

BlueMoon Software. URL: http://www.bluemoon.ee/history/roketz/

Microsoft Corporation.” Invalid RDP Data Can Cause Memory Leak in Terminal Services”. July 2001. URL: www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-040.asp

Vidstrom, Arnie.” The use of TCP port 445 in Windows 2000”. URL: http://ntsecurity.nu/papers/port445/

The MITRE Corporation, “Common Vulnerabilities and Exposures List. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0597

The MITRE Corporation, “Common Vulnerabilities and Exposures List. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0283

Alarcon, Stephanie. Analyst 0438 SANS GCIH Practical. May 2003. http://www.giac.org/practical/GCIH/stephanie_alarcon_GCIH.pdf.

Dshield.org. Distributed Intrusion Detection System. URL: http://www.dshield.org/ports/port113.php

URL: http://www.megasecurity.org/trojans/t/terrortrojan/Terrortrojan1.0.html

Zimmerman, Scott. “Secure Infrastructure Design.”.URL: http:/www.cert.org/archive/pdf/Secure_Infrastructure_Design.pdf

TABLE OF ENDNOTES

[1] Sequeira, Dinesh. “Intrusion Prevention Systems: Security’s Silver Bullet?”. URL: http://www.bcr.com/bcrmag/2003/03/p36.asp

[2] The MITRE Corporation, “Common Vulnerabilities and Exposures List “. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0236

[3] ITSX BV. “Title: Solaris SNMP to DMI mapper daemon vulnerability”. March 2001. URL: http://www.itsx.com/snmpXdmid.html

[4] Entercept Security Technologies. “SEA SNMP - Buffer Overflow and Format String Vulnerabilities in Sun Solaris”. URL: http://www.entercept.com/dr/snmp

[5] Anonymous. “Definition: Buffer Overflow”. URL: http://commons.somewhere.com/buzz/2000/Definition.Buffer.Overfl.html

[6] Noordergraaf, Alex. “JumpStart Technology: Effective Use in the Solaris Operating Environment”.October 2001. URL:http://www.sun.com/presents/discussions/disc-112701/

[7] Sun Microsystems. “Solaris Security Toolkit (JASS) “. URL: http://wwws.sun.com/software/security/jass/

[8] Sun Microsystems. URL: http://sunsolve.sun.com/pub-cgi/show.pl?target=patches/patch-access

[9] Caldera International. “Introduction to the desktop management interface“. URL: http://uw713doc.caldera.com/en/SDK_dmi/Introduction_to_the_desktop_management_interface.html

[10] Caldera International. “The role of the DMI Service Provider”. URL: http://uw713doc.caldera.com/en/SDK_dmi/The_role_of_the_DMI_Service_Provide.html

[11] Sun Microsystems. “Using SNMP With DMI”. Sun Product Documentation. URL: http://docs.sun.com/db/doc/806-2905/6jc3duo48?a=view.

[12] Caldera International. URL: http://uw713doc.caldera.com/en/man/html.3dmi/DmiComponentAdded.3dmi.html

[13] Reed, Darren. “IP Filter”. URL: http://coombs.anu.edu.au/~avalon/ip-filter.html

[14] Zalewski, Michal & Stearns, William. “p0f - passive OS fingerprinting”. URL: http://www.stearns.org/p0f/p0f.fp

[15] Microsoft Corporation. “Windows 2000 QoS Packet Scheduler Sends Packets with Wrong Checksum on Network Adapters That Enable Hardware Checksum”. Microsoft Knowledge Base Article. URL: http://support.microsoft.com/default.aspx?scid=kb;EN-US;270923

[16] Jacobson, V & Braden, R. “TCP Extensions for Long-Delay Paths”. October 1988. URL: http://www.faqs.org/rfcs/rfc1072.html

[17] West, M. McCann, S. “TCP/IP Field Behavior”. Internet-Draft. March 2003. URL: http://www.ietf.org/internet-drafts/draft-ietf-rohc-tcp-field-behavior-02.txt

[18] Unicode Inc. “What is Unicode”.URL: http://www.unicode.org/standard/WhatIsUnicode.html

[19] PhiRo-In. “Exploit the IIS hole using the Echo Style”. URL: http://www.rootshell.be/~doxical/download/docs/win/Exploit-unicode.rtf

[20] Cahoon, Brian. Analyst 0602 SANS GCIA Practical. January 2003. URL: http://www.giac.org/practical/GCIA/Brian_Cahoon.pdf

[21] Beardsley, Tod. Analyst 0525 SANS GCIA Practical. May 2002. URL: http://www.giac.org/practical/Tod_Beardsley_GCIA.doc

[22] CERT Coordination Center. “CERT® Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL”. January 2002. URL: http://www.cert.org/advisories/CA-2001-19.html

[23] Reiter, Michael. “Analysis of the Adoreworm/CodeRed”. 2001. URL:http://www.giac.org/practical/Michael_Reiter_GCFW.zip

[24] PSI.” Microsoft Windows 2000 TCP/IP Configuration Guide”. URL: http://www.support.psi.com/support/common/inet-serv/tcpip/2000/win2k_tcpip.html

[25] Posting. URL: http://www.incidents.org/archives/intrusions/msg01325.html

[26] Sourcefire, INC. “Snort Signature Database”. URL: http://www.snort.org/snort-db/sid.html?sid=648

[27] Securityfocus. FOCUS-IDS Archive. April 2002. URL: http://www.securityfocus.com/archive/96/266455/2002-04-08/2002-04-14/0

[28] Orebaugh, Angela. Analyst 0498 SANS GCIA Practical. March 2002. URL: www.giac.org/practical/Angela_Orebaugh_GCIA.doc

[29] rain.forest.puppy.”Perl CGI problems “. URL:http://www.wiretrip.net/rfp/txt/phrack55.txt

[30] Neohapsis Archives. November 2000. URL: http://archives.neohapsis.com/archives/snort/2000-11/0248.html

[31]McCabe, Mike. Analyst 0577 SANS GCIA Practical. November 2002. URL: http://www.giac.org/practical/GCIA/Mike_McCabe_GCIA.doc

[32] Novell. “Protocols and Ports used by NetWare 5 IP”. June 2001. URL: http://www.novell.com/coolsolutions/netware/features/a_ports_nw5_nw.html

[33] Fyodor. “Remote OS detection via TCP/IP Stack FingerPrinting”. October 1998. URL:http://www.insecure.org/nmap/nmap-fingerprinting-article.txt

[34] ICAT. http://icat.nist.gov/icat.cfm?cvename=CAN-2001-0353

[35] SecurityFocus. “Multiple Vendor LPRng User-Supplied Format String Vulnerability“. URL: http://www.securityfocus.com/bid/1712/discussion/

[36] Yuen, Rick Winkey. Analyst 0435 SANS GCIA Practical. October 2001. URL: http://www.giac.org/practical/Rick_Yuen_GCIA.doc.

[37] CERT Coordination Center. “Code Red II: Another Worm Exploiting Buffer Overflow In IIS Indexing Service DLL “. August 2001. URL: http://www.cert.org/incident_notes/IN-2001-09.html

[38] Harrington D., Presuhn R., Wijnen B. “An Architecture for Describing SNMP Management Frameworks”. January 1998. URL: http://www.faqs.org/rfcs/rfc2271.html

[39] Romananski, James.” Default SNMP Community Strings Set to "public and "private"" from “Using SNMP for Reconnaissance “. URL:http://www.sans.org/resources/idfaq/snmp.php

[40] SecurityFocus. “Microsoft Windows 2000 SNMP Printer Query Denial of Service Vulnerability”. October 2002. URL: http://www.securityfocus.com/bid/6030

[41] SecurityFocus. “hIRIX SNMP Daemon Buffer Overflow Vulnerability” . URL: http://www.securityfocus.com/bid/4421/discussion/

[42] CERT Coordination Center . “CERT® Advisory CA-2003-19 Exploitation of Vulnerabilities in Microsoft RPC Interface”. July 2003. URL: http://www.cert.org/advisories/CA-2003-19.html

[43] Taylor, Randy.” IDS Evasion and Denial of Service Using RPC Design Flaws”. URL:http://www.sans.org/resources/idfaq/rpc_evas.php

[44] Tripwire Inc. URL: http://www.tripwire.com/

[45] Crocker, Shelly.”FTP and the Warez Scene”. URL: http://www.sans.org/rr/papers/60/978.pdf.

[46] Smith, Donald.” Mscan, Sscan and Synscan - The evolution of worm - enabling vulnerability scanners that span two Millenniums.” URL:http://www.whitehats.ca/main/publications/external_pubs/scanner_fingerprints/scanner_fingerprints.html

[47] Thibault, David. Analyst 0287 SANS GIAC Practical. November 2000. URL: www.giac.org/practical/David_Thibault_GCIA.html

[48] Global Incident Analysis Center. November 2000. URL: http://www.sans.org/y2k/111600.htm

[49] IANA. “Special-Use IPv4 Addresses”. September 2002. URL: http://www.faqs.org/rfcs/rfc3330.html

[50] Neohapsis Archives. Posting. September 2000. URL: http://archives.neohapsis.com/archives/incidents/2000-09/0208.html.

[51] Apache Software Foundation.” Apache Module mod_info”. Apache HTTP Server Documentation Project. http://httpd.apache.org/docs-2.0/mod/mod_info.html

[52] SecurityFocus. “Apache Tomcat Snoop Servlet Information Disclosure Vulnerability”. July 2000. URL: http://www.securityfocus.com/bid/1532

[53] CERT® Coordination Center. “CERT® Advisory CA-2002-31 Multiple Vulnerabilities in BIND”. November 2002. URL: http://www.cert.org/advisories/CA-2002-31.html

[54] Alexander, Bruce. “Intrusion Detection FAQ Port 137 Scan”. URL: http://www.sans.org/resources/idfaq/port_137.php

[55] CERT® Coordination Center . “Windows NT Configuration Guidelines”. URL: http://www.cert.org/tech_tips/win_configuration_guidelines.html

[56] Internet Storm Center. Port Reports. URL: http://isc.incidents.org/port_details.html?port=1214

[57]URL: http://winmx.2038.net/winmx/fr-blocked.html

[58] BlueMoon Software. URL: http://www.bluemoon.ee/history/roketz/

[59] Microsoft Corporation.” Invalid RDP Data Can Cause Memory Leak in Terminal Services”. July 2001. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-040.asp

[60] Vidstrom, Arnie.” The use of TCP port 445 in Windows 2000”. URL: http://ntsecurity.nu/papers/port445/

[61] The MITRE Corporation, “Common Vulnerabilities and Exposures List. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0597

[62] The MITRE Corporation, “Common Vulnerabilities and Exposures List. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0283

[63] Alarcon, Stephanie. Analyst 0438 SANS GCIH Practical. May 2003. http://www.giac.org/practical/GCIH/stephanie_alarcon_GCIH.pdf.

[64] Dshield.org. Distributed Intrusion Detection System. URL: http://www.dshield.org/ports/port113.php

[65] URL: http://www.megasecurity.org/trojans/t/terrortrojan/Terrortrojan1.0.html

[66] Zimmerman, Scott. “Secure Infrastructure Design.”.URL: http:/www.cert.org/archive/pdf/Secure_Infrastructure_Design.pdf