Table of Contents

Section I – Exploit Analysis

Pwdump3

In 1997, Jeremy Allison released Pwdump – a utility that would extract the password hashes from a Windows NT system for offline cracking. This program was revolutionary for the time and contributed to Microsoft's release of Service Pack 3 for NT. Service Pack 3 provided syskey, which rendered this attack useless until Todd Sabin released Pwdump2 in 1998. Although pwdump2 had to be run locally, it remained effective when combined with password cracking tools like l0phtcrack. In February 2001, e-business technology, Inc. released Pwdump3, which operates across the network, with or without syskey enabled.

How is this a threat? Daniel Marvin stated it well in his article:

"This is the vulnerability: access to one resource allows access to a second resource. Now, how does the access of the first machine lead to access of the second machine? Pay attention here, this material will get your manager's attention in a hurry! If your enterprise is normal and uses a common password for the Local Administrator account then any employee sitting at an NT workstation could own your CEO's access in less than a week."

Pwdump3 a powerful program that, like other tools, is a tool of intent. If you mix ill intent with bad practice, someone is going to be in a lot of trouble. What would happen if someone within your organization used this tool with malicious intent?

The Attack

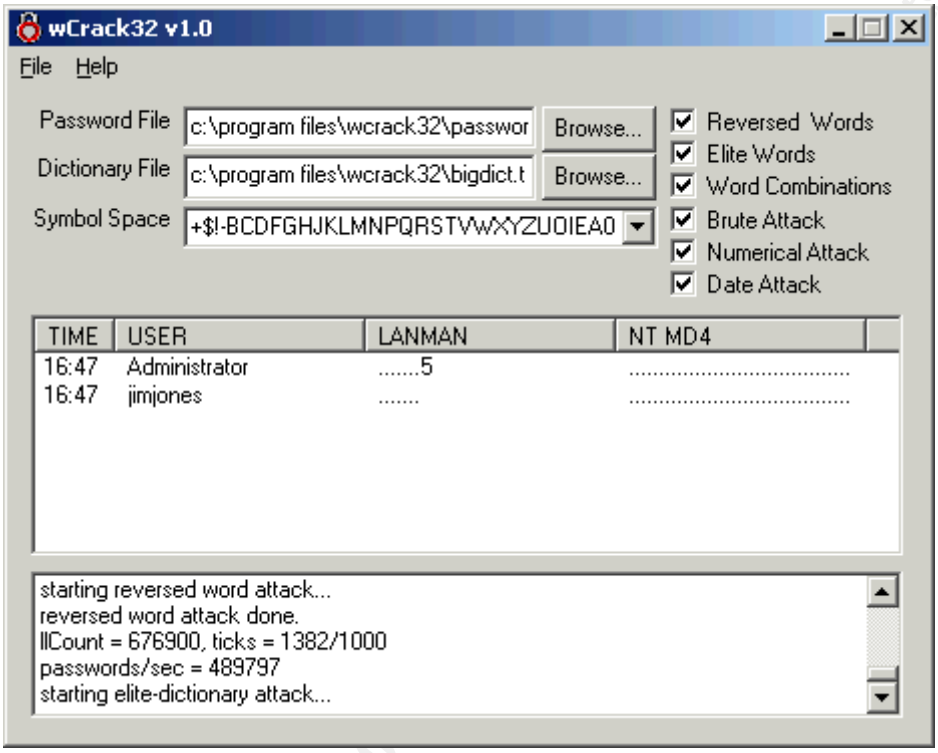
Say that our developer, John, was recently passed-over for promotion and has administrative rights on a few test machines that stage web development initiatives. He grabs pwdump3 and gets to work:

```
C:\pwdump3e>pwdump3e
Usage: PWDUMP3E machineName [outputFile] [userName]

C:\pwdump3e>pwdump3e testbox passwordlist johnquik
                Please enter the password >******
                Completed.

C:\pwdump3e>type passwordlist
Administrator:500: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3A:::
Guest:501:NO PASSWORD********************:NO PASSWORD********************:::
TsInternetUser:1000: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3:::
IUSR_TESTBOX:1001: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3:::
IWAM_TESTBOX:1002: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3:::
johnquik:1111: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3A:::
jimjones:1111: DC20C34715CC6CAC8B18DAF08C09EA3A: DC20C34715CC6CAC8B18DAF08C09EA3A:::
TESTBOX$:1008:NO PASSWORD********************:DC20C34715CC6CAC8B18DAF08C09EA3A:::
                                                                                Sanitized
```

Note that there are two interesting accounts here, the local administrator account and jimjones. Most organizations use the same local administrator account for every machine in the organization. Jim Jones is a database administrator in another department. John decides to get cracking; he uses crack32, because it's quick, free, and has a GUI.



Within an hour, he has the administrator password w0b3gon5, and the jimjones password "scleROSIS." What else does John have access to? He notices that most of the boxes have the same administrator password, and that jimjones domain account password is the same as the local account that was cracked.

Using Sysinternals pstools, John does whatever he wants while remaining extremely quiet. There may also be some local accounts that are very similar to domain admin accounts used by system technicians (i.e. local accounts with identical passwords). In most environments, "the game is over". John can simply peruse until he finds and exploits an account that yields a domain administrator password dump.

Countermeasures

To deal with this sort of attack, the first line of defense should be good local password account policies, including generated, unique passwords for each local administrator account. Auditing should also be centralized and monitored, and security personnel should be immediately notified by network intrusion detection of possible misuse of Pwdump3.

The readme that comes with Pwdump3e states:

"Remote access to a machine is accomplished by running the hash extraction program as a service, because Windows NT/2000 allows services to be installed and started remotely. Pwdump3e first connects to the ADMIN$ share and copies the service executable files there. It then requests the Service Control Manager to install and then run the service program. The extracted hash information is temporarily stored in the remote machine's registry. Pwdump3e remotely connects to the registry to read the stored data. Cleanup consists of removing the registry data, un-installing the service, and deleting the executable files from the remote machine."

To get a clearer understanding, look at the heart of the code. It (1) instantiates the service manager, (2) creates the service, (3) passes arguments, and (4) starts the service. Note that there are several unique things to key on, but that pwservice occurs in two different operations.

```
    // establish the service on remote machine
1   hscm = OpenSCManager( machineName, NULL, SC_MANAGER_CREATE_SERVICE );
    if( !hscm )
    {
       sprintf( errMsg, "Failed to open SCM\n" );
       throw errMsg;
    }

2    hsvc = CreateService( hscm, "pwservice", "PW Dumper", SERVICE_ALL_ACCESS,
                      SERVICE_WIN32_OWN_PROCESS, SERVICE_DEMAND_START, SERVICE_ERROR_IGNORE,
                      "pwservice.exe", NULL, NULL, NULL, NULL, NULL );
    if( !hsvc )
    {
       hsvc = OpenService( hscm, "pwservice", SERVICE_ALL_ACCESS  );
       if( !hsvc )
       {
          sprintf( errMsg, "Failed to create service\n" );
          throw errMsg;
       }
    }

    // parameter for service
3   const char* varg[1];
    varg[0] = keyName;

    // run service
4   if( !StartService( hsvc, 1, varg ) )
       fprintf( stderr, "Service failed: %d\n", GetLastError() );
```

Can we find the "Dumper" keyword in a network trace?

```
windump -X -r pwdump3e_1.tcpdump | grep Dumper
0x00d0   0a00 0000 5057 2044 756d 7065 7200 1339        ....PW.Dumper...
```

The dumper keyword occurred once making it ideal to flag as questionable activity. In order to narrow the scope further, we should look at the entire datagram.

```
11:13:40.515668 x.x.x.x.1035 > y.y.y.y.139: P 113683:113939(256) ack 149268 win
>>> NBT Packet
NBT Session Packet
Flags=0x0
Length=252 (0xfc)

SMB PACKET: SMBtrans (REQUEST)

0x0000   4500 0128 04b3 4000 4006 84e9 ac10 ac10        E..(..@.@.......
0x0010   ac10 ac02 040b 008b f43b 068d 2d96 883f        .........;..-..?
0x0020   5018 fa6f d3ac 0000 0000 00fc ff53 4d42        P..o.........SMB
0x0030   2500 0000 0018 07c8 0000 b766 6418 d53e        %..........fd..>
0x0040   c581 0000 0708 8404 0110 4144 1000 00a8        ..........AD....
0x0050   0000 0000 0400 0000 0000 0000 0000 0000        ................
0x0060   0054 00a8 0054 0002 0026 0008 80b9 0000        .T...T...&......
0x0070   5c00 5000 4900 5000 4500 5c00 0000 ff00        \.P.I.P.E.\.....
0x0080   0500 0003 1000 0000 a800 0000 0200 0000        ................
0x0090   9000 0000 0000 1800 0000 0000 500b e38e        ............P...
0x00a0   47d1 d511 ab10 0010 b568 f2ab 0a00 0000        G........h......
0x00b0   0000 0000 0a00 0000 7077 7365 7276 6963        ........pwservic
0x00c0   6500 4860 2cb2 4000 0a00 0000 0000 0000        e.H`,.@.........
0x00d0   0a00 0000 5057 2044 756d 7065 7200 0000          ....PW.Dumper...
0x00e0   ff01 0f00 1000 0000 0300 0000 0000 0000        ................
0x00f0   0e00 0000 0000 0000 0e00 0000 7077 7365        ............pwse
0x0100   7276 6963 652e 6578 6500 0000 0000 0000          rvice.exe.......
0x0110   0000 0000 0000 0000 0000 0000 0000 0000        ................
0x0120   0000 0000 0000 0000                            ........
```

In addition to the word “dumper”, note that the pwservice.exe keyword that occurs twice. This is inline with the code that we looked at before indicating that a successful remote password dump has been performed. This is definitely something that our IDS should be looking for.

If we wanted to come up with a snort rule to look for this type of activity, we would start with the most basic of characteristics. Those characteristics are the source port and any flags and PW Dumper. Pwservice occurs twice in this packet, as well as in other packets. However, adding it as a keyword could possibly lower false positives.

Snort Rule

alert tcp any any -> any 139 (msg:"PWDump3 activity"; flags: A+; content: "Dumper"; content: "pwservice";)

Reference

http://www.ebiz-tech.com/html/pwdump.html
http://www.entmag.com/archive/1997/june11/061129.asp
http://www.sans.org/infosecFAQ/authentic/insecurity.htm
http://www.sans.org/infosecFAQ/authentic/pass_protect.htm
http://www.securityfocus.com/infocus/1353
http://www.webspan.net/~tas/pwdump2/

Section II - Detects

Nimda Worm

Snort Logs

```
[**] [110:4:1] spp_unidecode: Invalid Unicode String detected [**]
09/25-17:24:40.022957 1.2.3.4:3633 -> x.x.x.x:80
TCP TTL:120 TOS:0x0 ID:3233 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x88ED44D1  Ack: 0x38B58B1F  Win: 0x4470  TcpLen: 20

---------------------------------------- snip ----------------------------------------

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
09/25-17:24:40.458528 1.2.3.4:3657 -> x.x.x.x:80
TCP TTL:120 TOS:0x0 ID:3297 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x890119FE  Ack: 0x39153CEC  Win: 0x4470  TcpLen: 20

[**] [1:974:3] WEB-IIS .... access [**]
[Classification: Web Application Attack] [Priority: 1]
09/25-17:24:40.655870 1.2.3.4:3670 -> x.x.x.x:80
TCP TTL:120 TOS:0x0 ID:3326 IpLen:20 DgmLen:137 DF
***AP*** Seq: 0x890C6FDD  Ack: 0x38B4C772  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
09/25-17:24:40.895471 1.2.3.4:3680 -> x.x.x.x:80
TCP TTL:120 TOS:0x0 ID:3358 IpLen:20 DgmLen:138 DF
***AP*** Seq: 0x89141421  Ack: 0x38B8107F  Win: 0x4470  TcpLen: 20
```

Apache Logs

```
"GET /scripts/root.exe?/c+dir HTTP/1.0" 403 301 "-" "-"
"GET /MSADC/root.exe?/c+dir HTTP/1.0" 403 299 "-" "-"
"GET /c/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 309 "-" "-"
"GET /d/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 309 "-" "-"
"GET /scripts/..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 323 "-" "-"
"GET /_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir –snip--
"GET /_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir –snip--
"GET/msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%c1%1c../winnt/system32/cmd.exe?/c+dir
"GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 322 "-" "-"
"GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 318 "-" "-"
"GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 322 "-" "-"
"GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 322 "-" "-"
"GET /scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 302 "-" "-"
"GET /scripts/..%%35c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 302 "-" "-"
"GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 323 "-" "-"
"GET /scripts/..%252f../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 403 323 "-" "-"
```

Source

This signature was detected on my cable modem connection.

Generated by
This detect was generated by Snort using the default rule set against a raw tcpdump file.

Spoof Probability
This detect is probably not spoofed because the pattern is identical to a compromised host that is seeking to infect other hosts. This pattern is repetitive and constant from these hosts.

Description
This detect is the Nimda worm attempting to propagate to vulnerable Microsoft IIS machines. Nimda tries over 15 different requests that utilize the "IIS/PWS Extended UNICODE Directory Traversal Vulnerability", the "IIS/PWS Escaped Character Decoding Command Execution Vulnerability", and code left behind by the Code Red II and Sadmind worms. When a host is infected, the payload is uploaded to the host with the TFTP command. The name of the file transferred is admin.dll (Nimda is "admin" backwards). The worm then adds a web server mime type (eml) that is propagated to Internet Explorer web browsers that request pages from that server. If they are Windows NT machines, they too begin the process of looking for vulnerable web servers. Nimda also propagates by email and by infecting executables in open network shares.

The Attack
The trace above shows the web server propagation method of Nimda, so that is what will be discussed. The worm actually exploits two major vulnerabilities in Microsoft Internet Information Server.

The first is the IIS/PWS Extended UNICODE Directory Traversal Vulnerability, or simply put, the UNICODE vulnerability. The UNICODE vulnerability was discovered in October of 2000 by an anonymous poster to a PacketStorm forum. Microsoft posted the MS00-078 bulletin on October 17, 2000. This exploit makes it possible to perform a ../ or ..\ if extended UNICODE characters are used, such as:

http://target/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir
http://target/scripts/..%c0%9v../winnt/system32/cmd.exe?/c+dir
http://target/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
http://target/scripts/..%c0%qf../winnt/system32/cmd.exe?/c+dir

The ramifications of this were monumental. This is effectively a remote command shell, allowing TFTP uploads and any system level actions to be taken against the server.

The second form of propagation utilized the IIS/PWS Escaped Character Decoding Command Execution Vulnerability. This vulnerability was discovered on May 15, 2001. In response, Microsoft released Security Bulletin MS01-026. Security Focus states:

"If a malformed filename is submitted and circumvents the initial security check, the undocumented procedure will decode the malformed request, possibly allowing the execution of arbitrary commands."

This is another remote command shell via URL requests.

Finally the Nimda worm uses root.exe, left over from the Code Red II and Sadmind worms to propagate. Root.exe is a copy of cmd.exe, placed in the /scripts directory, that may not have been properly removed from a previously infected web server.

Correlations
The correlations to this activity are extensive. The Cooperative Association for Internet Data Analysis posted analysis of Nimda activity in relation to Internet "HTTP" requests. They noted the following:

"By 17:00 PDT (00:00 GMT) on September 19th, we had observed ***450,000*** *unique IP addresses attempting to spread the Nimda worm. The discrepancy between the number of hosts infected at any given time and this number of unique IP addresses initially is caused by the removal of many pools of infected hosts from the Internet. Some organizations chose to remove themselves voluntarily to protect their machines. Some ISPs disconnected customers who were found to be spreading the worm, while others blocked traffic to or from port 80.* ***Finally, some locations were compromised so severely that the infected hosts saturated their links to the rest of the Internet, thereby reducing the ability of the infected hosts to spread the worm.*** *This saturation also may have overwhelmed BGP keepalive messages, causing withdrawal of routes. Information about disinfection and prevention of Nimda was released around 16:30 PDT (23:30 GMT)."* [emphasis by author]

In terms of reach, Nimda touched everything that is on the Internet, and most internal networks. Nimda activity is on the decline, according to the Cooperative Association for Internet Data Analysis, but it has not completely subsided.

Targeting
This is activity was not actively targeted. Nimda will seek out machines with the same second octet 50% of the time. It will then target machines sharing the first octet 25% of the time, as well as random machines 25% of the time.

Severity
- Criticality: 0. This is just my personal web server
- Lethality: 0. This is a lethal exploit, only not against apache.
- Sys Countermeasures: 0. The system is not vulnerable to this attack.
- Net Countermeasures: 2. IDS in place.
- Severity 0. (0+0) – (0+2) = -2

Defense
Apart from stringent perimeter defense, recommendations are to make sure that systems are updated with the latest service packs and hotfixes. Also, utilize tools like URLScan

2.0 from Microsoft, which can greatly reduce the amount of damage that can be done by similar attacks. It does this by validating inbound requests against configuration options. If a URL contains "cmd," then it will not be processed. It allows the option to deny everything that is not explicitly allowed, such as default.asp and index.htm.

Question

Large numbers of UNICODE alerts on a network are usually indicative of:

a) a host(s) infected with the Code Red worm.
b) a host(s) infected with the Sadmind worm.
c) a host(s) infected with the .net worm.
d) a host(s) infected with the Nimda worm.

Reference

http://www.antivirus.com/vinfo/virusencyclo/default5.asp?VName=PE_NIMDA.A&VSect=T
http://securityresponse.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html
http://www.microsoft.com/technet/security/bulletin/ms00-078.asp?frame=true
http://www.microsoft.com/technet/security/bulletin/MS01-020.asp?frame=true
http://www.securityfocus.com/bid/1806
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0884
http://www.securityfocus.com/bid/1101
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0258
http://www.caida.org/dynamic/analysis/security/nimda/

Sam Access

SMB sam file 2001-10-15 10:27:57 2001-10-15 15:58:35 CUSTOM 192.168.5.1 192.168.8.9 NETBIOSNAME

Source
The source of this alert was my employer's network.

Generated by
This activity was generated by an enterprise ISS Black Ice management console. Black Ice is installed and runs unknown to the end user.

Spoof Probability
This activity is not likely to be spoofed because the alert was detected on the client side, going outbound.

Description
The Security Accounts Manager (SAM) contains the username and password pairs for all of the local users. This type of attack is devastating if the intent is to use one local administrative account to compromise another, eventually leading to an account that has domain administrative privileges.

The Attack
This attack consists of utilizing tools like Pwdump3 by e-business technology, Inc. (discussed earlier). The attacker is attempting to get the user account database so that a password-cracking program can be executed against it. Regarding this type of attack, ISS states:

"Never run file sharing on a 'Domain Controller', because only Domain Controllers have the full SAM database, which should be protected at all costs."

In this case, the machine was not a domain controller, yet it was a production file server. Ultimately, the attacker may now crack the password database, exposing the local administrator account password. If this password were consistent with other local administrator accounts, it would have to be changed on all of the affected machines.

Correlations
There have been no correlations to this attack.

Targeting
This was probably a targeted attack because it was an isolated attack against one host.

Severity
- Criticality: 2. This was a production fileserver

- Lethality: 5. This is a lethal exploit.
- Sys Countermeasures: 2. The system was running ISS Black Ice.
- Net Countermeasures: 0. No network countermeasures.
- Severity 0. (2+4) – (2+0) = 4

Defense

This type of attack is difficult to defend against. One form of defense would be to disable the administrative shares (c$, admin$, etc.). In order for this attack to work, Pwdump3 has to copy files over to the target, install a service that runs as system, and dump the passwords. If the administrative shares are not there, this process cannot be completed. NT password hashing and password vulnerabilities are listed in "SANS Top 20 Internet Vulnerabilities." Be sure to enforce strong password and auditing policies. It would be a good idea to crack your own passwords occasionally, to make sure that the policies are actually being adhered to.

Question

In terms of network security, remote compromise of an NT SAM is devastating because:

a) the entire machine will have to be rebuilt.
b) the system time will not be synchronized with the network
c) the local administrative accounts are typically the same on all machines
d) the SAM contains the administrative console for remote registry changes.

Reference

http://www.ebiz-tech.com/html/pwdump.html
http://www.entmag.com/archive/1997/june11/061129.asp
http://www.sans.org/infosecFAQ/authentic/insecurity.htm
http://www.sans.org/infosecFAQ/authentic/pass_protect.htm
http://www.securityfocus.com/infocus/1353
http://www.webspan.net/~tas/pwdump2/

SNMP Host Scanning

SNMP backdoor	2001-11-03 18:56:16	2001-11-03 18:56:16	CUSTOM	10.10.10.10	208.167.172.65	MISC
SNMP backdoor	2001-11-03 18:56:16	2001-11-03 18:56:16	CUSTOM	10.10.10.10	208.167.172.66	MISC
SNMP backdoor	2001-11-03 18:56:22	2001-11-03 18:56:22	CUSTOM	10.10.10.10	208.167.172.90	MISC
SNMP backdoor	2001-11-03 18:56:25	2001-11-03 18:56:25	CUSTOM	10.10.10.10	208.167.172.127	MISC
		--Snipped--				
SNMP backdoor	2001-11-07 15:05:53	2001-11-07 15:05:53	CUSTOM	10.10.10.10	194.200.187.15	MISC
SNMP backdoor	2001-11-07 15:05:53	2001-11-07 15:05:53	CUSTOM	10.10.10.10	194.200.187.14	MISC
SNMP backdoor	2001-11-07 15:05:54	2001-11-07 15:05:54	CUSTOM	10.10.10.10	194.200.187.46	MISC
SNMP backdoor	2001-11-07 15:06:01	2001-11-07 15:06:01	CUSTOM	10.10.10.10	194.200.187.249	MISC
SNMP backdoor	2001-11-07 15:06:01	2001-11-07 15:06:01	CUSTOM	10.10.10.10	194.200.187.253	MISC
SNMP backdoor	2001-11-07 15:06:09	2001-11-07 15:06:09	CUSTOM	10.10.10.10	194.200.187.224	MISC
SNMP backdoor	2001-11-07 15:06:11	2001-11-07 15:06:11	CUSTOM	10.10.10.10	194.200.187.254	MISC
		--Snipped--				
SNMP backdoor	2001-11-07 16:05:34	2001-11-07 16:05:34	CUSTOM	10.10.10.10	65.89.31.184	MISC
SNMP backdoor	2001-11-07 16:05:34	2001-11-07 16:05:34	CUSTOM	10.10.10.10	65.89.31.106	MISC
SNMP backdoor	2001-11-07 16:05:49	2001-11-07 16:05:49	CUSTOM	10.10.10.10	65.89.31.141	MISC
SNMP backdoor	2001-11-07 16:05:50	2001-11-07 16:05:50	CUSTOM	10.10.10.10	65.89.31.233	MISC
		--Snipped--				
SNMP backdoor	2001-11-08 10:58:57	2001-11-08 10:58:57	CUSTOM	10.10.10.10	64.36.46.33	MISC
SNMP backdoor	2001-11-08 10:58:58	2001-11-08 10:58:58	CUSTOM	10.10.10.10	64.36.46.62	MISC
SNMP backdoor	2001-11-08 10:59:00	2001-11-08 10:59:00	CUSTOM	10.10.10.10	64.36.46.63	MISC
SNMP backdoor	2001-11-08 10:59:05	2001-11-08 10:59:05	CUSTOM	10.10.10.10	64.36.46.193	MISC

Source
The source of this alert was my employer's network.

Generated by
This activity was generated by an enterprise ISS Black Ice management console. Black Ice is installed and runs unknown to the end user.

Spoof Probability
This activity is not likely to be spoofed because the alert was detected on the client side, going outbound.

Description
This is a network probe for a default SNMP (Simple Network Management Protocol) community string. It is also listed in "SANS Top 20 Most Critical Internet Threats". The following is a quote taken from sans.org:

"The Simple Network Management Protocol (SNMP) is widely used by network administrators to monitor and administer all types of network-connected devices ranging from routers to printers to computers. SNMP uses an unencrypted "community string" as its only authentication mechanism. Lack of encryption is bad enough, but the default community string used by the vast majority of SNMP devices is "public", with a few "clever" network equipment vendors changing the string to "private". Attackers can use this vulnerability in SNMP to reconfigure or shut down devices remotely. Sniffed

SNMP traffic can reveal a great deal about the structure of your network, as well as the systems and devices attached to it. Intruders use such information to pick targets and plan attacks."

The Attack

This attack mechanism is to scan a range of hosts for insecure community strings. In this case, the attacker is using the "private" community string, hoping to illicit responses from poorly configured hosts. This is also a full TCP/IP connection, and not a stealthy scan. The scan times are fairly quick, almost three per second. From this, and the randomness of the targets, this is probably an automated tool.

SNMP is used to monitor the status of and control network devices. It is very prominent, and often times uses the default community strings of "public" and/or "private". Community strings are basically the password that controls access to the device. This type of scanning is basically checking for hosts that have default passwords for full access, which makes it more than just a scan. The following hosts were scanned:

Cable & Wireless USA (NETBLK-CW-10BLK) CW-10BLK 208.128.0.0 - 208.175.255.255
WESTERN ASSET MANAGEMENT (NETBLK-CW-208-167-172) CW-208-167-172
208.167.172.0 - 208.167.172.63
Foxlink International Inc. (NETBLK-CW-208-167-172-64) CW-208-167-172-64
208.167.172.64 - 208.167.172.127

inetnum: 194.200.187.0 - 194.200.187.15
netname: RABCAPITD1-1-1
descr: **Rab Capital Ltd**
country: GB
admin-c: CDM31-RIPE
tech-c: CDM31-RIPE
status: ASSIGNED PA
mnt-by: AS1849-MNT
changed: jons@uk.uu.net 19990824
source: RIPE

Broadwing Communications, Inc. (NETBLK-BROADWING-2BLK) BROADWING-2BLK
65.88.0.0 - 65.91.255.255
Broadwing_Communications_Internal_Hayward (NETBLK-BRW-GLUEHAYWARD8931) BRW-GLUEHAYWARD8931
65.89.31.0 - 65.89.31.255

PM Realty Group, L.P. (NETBLK-78034) 78034 64.36.46.32 - 64.36.46.63

The scans are spread out over several days, and in short bursts. This does not appear to be a full-time scanner, or someone that is trying to be overly evasive.

Correlations

The incidents.org Handlers Diary (for November 19th, 2001) had an analysis of recent SNMP activity. The analysis described this type of scanner a *static machine that only scanned once,* due to the fact that there are no other occurrences from this source. This host was not listed in the incident.org database.

Targeting
This attack was probably not targeted. It appears that someone may have been testing a new piece of software.

Severity
- Criticality: 2. This attack is targeting core network devices.
- Lethality: 3. This is listed as a SANS Top 20 vulnerability.
- Sys Countermeasures: 2. The system was running ISS Black Ice.
- Net Countermeasures: 0. No network countermeasures.
- Severity 0. (2+3) – (2+0) = 3

Defense
The best defensive action to take against attacks like this is to disable SNMP if you are not using it. If you do need SNMP, be sure to have strong community strings. SNMP should also be blocked at the perimeter, both inbound and outbound. Organizations should also police their own networks and take appropriate disciplinary action against employees that scan any network for common vulnerabilities.

Question
The SNMP service is a risk primarily because:
a) It is ideally the most secure form of monitoring remote devices
b) It is considered by some to be virtually impenetrable
c) It is typically configured with default community strings, which control access
d) It is prone to service failure

Reference
http://www.sans.org/top20.htm
http://icat.nist.gov/icat.cfm?cvename=CAN-1999-0517
http://icat.nist.gov/icat.cfm?cvename=CAN-1999-0516
http://icat.nist.gov/icat.cfm?cvename=CAN-1999-0254
http://icat.nist.gov/icat.cfm?cvename=CAN-1999-0186

Trolling for SSH

```
Nov 17 23:31:02 206.251.11.242:22 -> xyz.xyz.xyz.137:22 SYN ******S*
Nov 17 23:31:02 206.251.11.242:22 -> xyz.xyz.xyz.138:22 SYN ******S*
Nov 17 23:31:02 206.251.11.242:22 -> xyz.xyz.xyz.139:22 SYN ******S*
Nov 18 00:22:53 206.251.11.242:22 -> xyz.xyz.xyz.137:22 SYN ******S*
Nov 18 00:22:53 206.251.11.242:22 -> xyz.xyz.xyz.139:22 SYN ******S*
Nov 18 00:22:53 206.251.11.242:22 -> xyz.xyz.xyz.138:22 SYN ******S*
Nov 18 00:22:53 206.251.11.242:22 -> xyz.xyz.xyz.153:22 SYN ******S*
Nov 18 01:20:40 206.251.11.242:22 -> xyz.xyz.xyz.137:22 SYN ******S*
Nov 18 01:20:40 206.251.11.242:22 -> xyz.xyz.xyz.138:22 SYN ******S*
Nov 18 01:20:40 206.251.11.242:22 -> xyz.xyz.xyz.139:22 SYN ******S*
Nov 18 02:12:12 206.251.11.242:22 -> xyz.xyz.xyz.137:22 SYN ******S*
Nov 18 02:12:12 206.251.11.242:22 -> xyz.xyz.xyz.139:22 SYN ******S*
Nov 18 02:12:12 206.251.11.242:22 -> xyz.xyz.xyz.138:22 SYN ******S*
Nov 18 02:12:12 206.251.11.242:22 -> xyz.xyz.xyz.153:22 SYN ******S*
```

Source
The source of this alert was my employer's network.

Generated by
This event was generated by Snort running on a perimeter bastion host.

Spoof Probability
This event is probably not spoofed because it correlates to other similar events from the same IP. Although this address is not likely a spoofed address, it was probably a compromised host.

Description
Called the "SSH CRC-32 Compensation Attack Detector Vulnerability", Security Focus describes this attack as:

"Secure Shell, or SSH, is an encrypted remote access protocol. SSH or code based on SSH is used by many systems all over the world and in a wide variety of commercial applications. An integer-overflow bug in the CRC32 compensation attack detection code may allow remote attackers to write values to arbitrary locations in memory."

This attack was initially believed to be too difficult because the attacker would have to have extensive knowledge of the processes running on the target machine. Security Focus posted an update to their original post that stated:

"There have been reports suggesting that this may be occuring. Since early September, independent, reliable sources have confirmed that this vulnerability is being exploited by attackers on the Internet. Security Focus does not currently have the exploit code being used, however this record will be updated if and when it becomes available."

The Attack
This is not an attack, but it is a scan for vulnerable hosts to attack. However, there is

exploit code in the wild. Teso developed exploit code, but did not make their code public. They did, however, make a public statement regarding their code on November 8th stating, there were much more sophisticated exploits available on IRC. Teso let David A. Dittrich analyze their exploit. In his analysis, Mr. Dittrich noted that the exploit works by brute-forcing the memory address of the target, opening a rogue command shell on a high numbered port.

In this alert, the host is from the same address, and same destination and source ports. The times are inline with the other detects posted to incidents.org.

Correlations
Patrick Nolan was the first to notice this source address as being a bad guy. He posted a message to incidents.org regarding a scan the he detected on November 17th. Laurie Zirkle then posted with similar activity from the previous day and the same host. John Sage then posted on November 18th:

"As I said in my previous post, while cross-checking the OS at Netcraft and by nmap, toward the end of my snort capture of the nmap scan 206.251.11.242 is suddenly making persistent attempts to connect to my firewall's tcp:22 ssh, which of course is locked down.."

The activity on our network was detected beginning on the 17th, and proceeding through the 18th four different times using the same pattern that Larie Zirkle observed.

Targeting
This activity was targeted because there were four different occurrences. It seems that one time was not enough.

Severity
- Criticality: 0. This host is just a test machine.
- Lethality: 4. This is a root level compromise.
- Sys Countermeasures: 3. Layered SSH defense - Unpatched.
- Net Countermeasures: 2. Intrusion Detection.
- Severity 0. (0+4) – (4+2) = -1

Defense
Defensive recommendations are to layer defense. Patches are important, but in this case the overall security of the host was the saving factor. At the time of this attack, the machine was running a vulnerable version of SSH. However, the machine enforces the principle of least access in determining which hosts can access it and how.

Question
A host has layered defense when:
a) A system adheres to the OSI model

b) A system has multiple login prompts
c) A system has security failover mechanisms
d) A system has many different security analysts

Reference

http://xforce.iss.net/alerts/advise100.php
http://razor.bindview.com/publish/advisories/adv_ssh1crc.html
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=info&id=2347
http://www.incidents.org/archives/intrusions/msg02489.html
http://www.incidents.org/archives/intrusions/msg02500.html
http://staff.washington.edu/dittrich/misc/ssh-analysis.txt

Trolling for Subseven

```
Nov 27 20:42:19 SCREEN 566703: *May 14 10:50:09: LOGP: list 100 denied tcp 207.14.233.114(3797) -> z.y.x.18(27374), 1 packet
Nov 27 20:42:21 SCREEN 566704: *May 14 10:50:10: LOGP: list 100 denied tcp 207.14.233.114(3800) -> z.y.x.21(27374), 1 packet
Nov 27 20:42:22 SCREEN 566705: *May 14 10:50:11: LOGP: list 100 denied tcp 207.14.233.114(3803) -> z.y.x.24(27374), 1 packet
Nov 27 20:42:23 SCREEN 566706: *May 14 10:50:13: LOGP: list 100 denied tcp 207.14.233.114(3807) -> z.y.x.28(27374), 1 packet
Nov 27 20:42:25 SCREEN 566707: *May 14 10:50:14: LOGP: list 100 denied tcp 207.14.233.114(3810) -> z.y.x.31(27374), 1 packet
Nov 27 20:43:29 SCREEN 566709: *May 14 10:51:19: LOGP: list 100 denied tcp 207.14.233.114(3973) -> z.y.x.193(27374), 1 packet
Nov 27 20:43:31 SCREEN 566711: *May 14 10:51:20: LOGP: list 100 denied tcp 207.14.233.114(3976) -> z.y.x.196(27374), 1 packet
Nov 27 20:43:32 SCREEN 566712: *May 14 10:51:21: LOGP: list 100 denied tcp 207.14.233.114(3800) -> z.y.x.21(27374), 3 packets
Nov 27 20:43:34 SCREEN 566713: *May 14 10:51:23: LOGP: list 100 denied tcp 207.14.233.114(3805) -> z.y.x.26(27374), 3 packets
Nov 27 20:43:35 SCREEN 566714: *May 14 10:51:24: LOGP: list 100 denied tcp 207.14.233.114(3808) -> z.y.x.29(27374), 3 packets
Nov 27 20:43:36 SCREEN 566715: *May 14 10:51:25: LOGP: list 100 denied tcp 207.14.233.114(3972) -> z.y.x.192(27374), 1 packet
Nov 27 20:43:38 SCREEN 566716: *May 14 10:51:27: LOGP: list 100 denied tcp 207.14.233.114(3976) -> z.y.x.196(27374), 1 packet
Nov 27 20:43:39 SCREEN 566717: *May 14 10:51:28: LOGP: list 100 denied tcp 207.14.233.114(3981) -> z.y.x.201(27374), 1 packet
Nov 27 20:43:40 SCREEN 566718: *May 14 10:51:29: LOGP: list 100 denied tcp 207.14.233.114(3984) -> z.y.x.204(27374), 1 packet
Nov 27 20:43:40 SCREEN 566719: *May 14 10:51:30: LOGP: list 100 denied tcp 207.14.233.114(3986) -> z.y.x.206(27374), 1 packet
Nov 27 20:43:43 SCREEN 566720: *May 14 10:51:32: LOGP: list 100 denied tcp 207.14.233.114(3988) -> z.y.x.208(27374), 1 packet
Nov 27 20:43:44 SCREEN 566721: *May 14 10:51:33: LOGP: list 100 denied tcp 207.14.233.114(3990) -> z.y.x.210(27374), 1 packet
Nov 27 20:43:45 SCREEN 566722: *May 14 10:51:34: LOGP: list 100 denied tcp 207.14.233.114(4012) -> z.y.x.232(27374), 1 packet
Nov 27 20:43:46 SCREEN 566723: *May 14 10:51:35: LOGP: list 100 denied tcp 207.14.233.114(3992) -> z.y.x.212(27374), 1 packet
Nov 27 20:43:47 SCREEN 566724: *May 14 10:51:37: LOGP: list 100 denied tcp 207.14.233.114(4018) -> z.y.x.238(27374), 1 packet
Nov 27 20:43:49 SCREEN 566725: *May 14 10:51:38: LOGP: list 100 denied tcp 207.14.233.114(4021) -> z.y.x.241(27374), 1 packet
Nov 27 20:43:50 SCREEN 566726: *May 14 10:51:39: LOGP: list 100 denied tcp 207.14.233.114(3972) -> z.y.x.192(27374), 1 packet
Nov 27 20:43:51 SCREEN 566727: *May 14 10:51:41: LOGP: list 100 denied tcp 207.14.233.114(4012) -> z.y.x.232(27374), 1 packet
Nov 27 20:43:52 SCREEN 566728: *May 14 10:51:42: LOGP: list 100 denied tcp 207.14.233.114(4016) -> z.y.x.236(27374), 1 packet
Nov 27 20:43:53 SCREEN 566729: *May 14 10:51:43: LOGP: list 100 denied tcp 207.14.233.114(4033) -> z.y.x.253(27374), 1 packet
Nov 27 20:43:55 SCREEN 566730: *May 14 10:51:44: LOGP: list 100 denied tcp 207.14.233.114(4029) -> z.y.x.249(27374), 1 packet
Nov 27 20:43:56 SCREEN 566731: *May 14 10:51:45: LOGP: list 100 denied tcp 207.14.233.114(3986) -> z.y.x.206(27374), 1 packet
Nov 27 20:43:57 SCREEN 566732: *May 14 10:51:46: LOGP: list 100 denied tcp 207.14.233.114(4034) -> z.y.x.254(27374), 1 packet
Nov 27 20:43:58 SCREEN 566733: *May 14 10:51:47: LOGP: list 100 denied tcp 207.14.233.114(4022) -> z.y.x.242(27374), 1 packet
Nov 27 20:43:59 SCREEN 566734: *May 14 10:51:48: LOGP: list 100 denied tcp 207.14.233.114(3995) -> z.y.x.215(27374), 1 packet
Nov 27 20:44:00 SCREEN 566735: *May 14 10:51:50: LOGP: list 100 denied tcp 207.14.233.114(4028) -> z.y.x.248(27374), 1 packet
Nov 27 20:44:02 SCREEN 566736: *May 14 10:51:51: LOGP: list 100 denied tcp 207.14.233.114(4031) -> z.y.x.251(27374), 1 packet
Nov 27 20:44:03 SCREEN 566737: *May 14 10:51:52: LOGP: list 100 denied tcp 207.14.233.114(4034) -> z.y.x.254(27374), 1 packet
Nov 27 20:44:05 SCREEN 566738: *May 14 10:51:55: LOGP: list 100 denied tcp 207.14.233.114(4010) -> z.y.x.230(27374), 1 packet
Nov 27 20:44:07 SCREEN 566739: *May 14 10:51:56: LOGP: list 100 denied tcp 207.14.233.114(4014) -> z.y.x.234(27374), 1 packet
Nov 27 20:44:08 SCREEN 566740: *May 14 10:51:57: LOGP: list 100 denied tcp 207.14.233.114(4017) -> z.y.x.237(27374), 1 packet
Nov 27 20:44:09 SCREEN 566741: *May 14 10:51:59: LOGP: list 100 denied tcp 207.14.233.114(4020) -> z.y.x.240(27374), 1 packet
Nov 27 20:44:11 SCREEN 566742: *May 14 10:52:00: LOGP: list 100 denied tcp 207.14.233.114(4023) -> z.y.x.243(27374), 1 packet
Nov 27 20:44:12 SCREEN 566743: *May 14 10:52:01: LOGP: list 100 denied tcp 207.14.233.114(4027) -> z.y.x.247(27374), 1 packet
Nov 27 20:44:14 SCREEN 566744: *May 14 10:52:03: LOGP: list 100 denied tcp 207.14.233.114(4031) -> z.y.x.251(27374), 1 packet
Nov 27 20:44:15 SCREEN 566745: *May 14 10:52:04: LOGP: list 100 denied tcp 207.14.233.114(4034) -> z.y.x.254(27374), 1 packet
```

Note: The router clock is wrong

Source
The event was generated on a small company's screen router.

Generated by
A Cisco screen router logging to a remote syslog generated this event.

Spoof Probability
This event is probably not spoofed. This is a simple SYN-scan that is attempting to illicit SYN-ACK packets from the target. This is also known as a half open scan. Most spoof scanning that occurs now uses the idle scan in the latest version of Nmap. It would have a typical service port as its source port that remained static. For example:

Attacker:80 -> target:27374
Attacker:80 -> target2:27374

Description
This attacker is trolling for subseven trojans. Subseven is known as a remote administration trojan. This is because of this type of malware allows total control of the victim. The attacker can do anything from read and write to the registry and filesystem, to reboot the computer and eject the cdrom. Some of its other features include:

- AIM, ICQ, MSN and Yahoo spy
- Control mouse
- Get cached passwords
- Packet sniffer

This is an extremely powerful tool that has a huge underground following. It is the most widely distributed backdoor/trojan on the Internet. It is highly configurable. There are several ways that the attackers can notify themselves when someone has inadvertently stepped into their trap.

The Attack
This attacker is doing a syn-scan over a class-c network in search of default subseven installs. The scanner resolves to:

SprintLink (NETBLK-SPRINT-W2) SPRINT-W2 207.12.0.0 - 207.15.255.255
Sprint Mid-Atlantic Telecom (NETBLK-SPRINT-CF0EE0-1) SPRINT-CF0EE0-1
207.14.224.0 - 207.14.239.255
Entersoft (NETBLK-ENTERSOFT-COM-BLK1) ENTERSOFT-COM-BLK1
207.14.233.0 - 207.14.238.255

Interestingly, this is the only scan to come through this network for the entire day of logs that were analyzed. Everything else was port 80/tcp worm traffic. This scan was very obvious in the logs. The randomization pattern seems too tight, almost like this class-C network was scanned alone. However, there is very little variation in the scanners ephemeral port, indicating that this may not be a very busy scanner. Also, is a relatively slow scan, approximately one packet per second. Overall, this seems to be a somewhat slow, focused scan: not too fast, not too slow, just wrong. This scan almost seems deliberate. Maybe this is a beginner's scan.

Correlations
Incidents.org recently noted increased scanning to this port on November 12th. Although our scanner is not one of the top scanners listed, with this type of activity he will be soon.

Targeting
This activity does suggest mild targeting, based on the slowness of the scan, and the tightness of the randomization pattern.

Severity

- Criticality: 1. This is the screen router in front of the company firewall.
- Lethality: 0. This is a simple host scan in search of trojans.
- Sys Countermeasures: 3. This is a router, and there is one NT machine total.
- Net Countermeasures: 2. Good ACL's
- Severity 0. (1+0) – (3+2) = -4

Defense

Defensive recommendations are to use good ACL's on screen routers to control this type of activity similar to the router that was targeted in this detect.

Question

Although it can be changed, the default port for the subseven backdoor is:

a) 12345/tcp
b) 37337/tcp
c) 27374/tcp
d) 65478/tcp

Section III - Analyze This

Overview

My University submitted nearly 100,000 network alert events to be analyzed. This does not include the port scans and out-of-spec logs that were submitted for analysis. This information consists of fifteen log files that start on the 24th of October and proceed through the 28th of October. This information reveals that there is nefarious activity on the network, as well as extremely questionable activity.

Top Targets					
Host	Dest	Src	Avg	Spread	Total
MY.NET.253.114	2914	2	583.2	2912	2916
MY.NET.235.110	2558	29	517.4	2529	2587
MY.NET.110.139	2318	9	465.4	2309	2327
MY.NET.140.9	1789	3	358.4	1786	1792
MY.NET.100.165	1616	50	333.2	1566	1666
MY.NET.219.50	1312	1	262.6	1311	1313
MY.NET.97.25	239	41	56	198	280
MY.NET.153.154	236	48	56.8	188	284
MY.NET.225.98	188	839	205.4	651	1027
MY.NET.253.114	2914	2	583.2	2912	2916

The top alert destinations are all within My University's subnet. The sources, average, spread, and total alerts per host are provided as well. This should clearly delineate the most active hosts. 253.114 is the most active alert destination logging almost 3000 events over a five day period. It averaged almost 400 events a day for the observed period. This is not the only factor for determining activity. For instance, 225.98 is actually more of an alert source than destination, averaging over 205 alerts per day. Overall activity is determined by a hosts destination count as well as its source count. Very often, this can indicate compromised or malicious hosts that have fallen to the dark side. For example, 225.98 recorded the following alerts:

- As a source 839 instances of ICMP Echo Request Nmap or HPING2
- As a destination 188 instances of *ICMP Destination Unreachable (Communication Administratively Prohibited)*

This machine seems to be trying to get somewhere it is not supposed to be.

Top Sources					
Host	Dest	Src	Avg	Spread	Total
MY.NET.208.246	107	8345	1690.4	-8238	8452
61.134.9.88	34	2319	470.6	-2285	2353
MY.NET.14.1	70	1616	337.2	-1546	1686
211.90.176.59	5	1334	267.8	-1329	1339
MY.NET.225.98	188	839	205.4	-651	1027
MY.NET.207.22	149	625	154.8	-476	774
MY.NET.205.46	94	429	104.6	-335	523
MY.NET.153.111	1	412	82.6	-411	413
130.205.92.178	1	397	79.6	-396	398
MY.NET.208.246	107	8345	1690.4	-8238	8452

The top source addresses are listed above. Notice that there are more addresses from My University's subnet than from the Internet. If we apply the source to destination principle to 14.1, we see a Cisco router that is receiving a lot of attention from various hosts:

```
10/25-08:10:50.849343  [**] ICMP Destination Unreachable (Administratively Prohibited) [**] MY.NET.14.1 -> MY.NET.218.130
10/25-08:10:53.174641  [**] ICMP traceroute  [**] MY.NET.98.131 -> MY.NET.14.1
10/25-08:11:39.459627  [**] ICMP traceroute  [**] MY.NET.218.198 -> MY.NET.14.1
10/25-08:11:55.380096  [**] ICMP Destination Unreachable (Administratively Prohibited) [**] MY.NET.14.1 -> MY.NET.234.50
10/25-08:12:04.584497  [**] ICMP Destination Unreachable (Administratively Prohibited) [**] MY.NET.14.1 -> MY.NET.98.148
10/25-08:12:28.436966  [**] ICMP Destination Unreachable (Administratively Prohibited) [**] MY.NET.14.1 -> MY.NET.209.110
```

The host correlation report is provided at the end of this document. This information was generated with a custom Perl script that is provided at the end of this report. This should help My University quickly identify the top talkers now, and in the future.

Alert Activity

Alert Description	Count
MISC Large UDP Packet	20571
Tiny Fragments - Possible Hostile Activity	10308
UDP SRC and DST outside network	8387
ICMP Echo Request speedera	8345
spp_http_decode: IIS Unicode attack detected	5893
Watchlist 000220 IL-ISDNNET-990517	5417
INFO MSN IM Chat data	3711
WEB-MISC Attempt to execute cmd	3628

The top alerts are shown above. The preceding top destination and top source address statistics to not count the first two top alerts shown above; this was done for clarity. The following are descriptions for each of the top alerts.

Misc. Large UDP Packet

Large UDP packets are typically one form of denial of service or another. Some machines can't properly handle excessive amounts of large UDP packets and freeze. In this case however, the packets are very big, and require fragmentation to be sent across the wire. This further supports the idea of denial of service. The reassembly time of excessive amounts of large UDP packets is exhaustive. To further support the denial of service theory, the originating hosts are from Asia, or at least appear to be. Here is a sample of the activity:

```
10/28-17:57:42.544731  [**] MISC Large UDP Packet [**] 61.134.9.88:28822 -> MY.NET.110.139:42519
10/28-17:57:42.951614  [**] Incomplete Packet Fragments Discarded [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:45.346528  [**] Incomplete Packet Fragments Discarded [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:46.345593  [**] MISC Large UDP Packet [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:46.544504  [**] MISC Large UDP Packet [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:46.749524  [**] MISC Large UDP Packet [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:49.853617  [**] MISC Large UDP Packet [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:50.044971  [**] MISC Large UDP Packet [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:57:52.843252  [**] Incomplete Packet Fragments Discarded [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:58:05.756262  [**] Incomplete Packet Fragments Discarded [**] 61.134.9.88:0 -> MY.NET.110.139:0
10/28-17:58:06.047001  [**] Incomplete Packet Fragments Discarded [**] 61.134.9.88:0 -> MY.NET.110.139:0
```

One interesting observation is the source and destination ports. They appear to be port 0 designations, but this is not the case. The first packet in a fragment train contains the port designations, yet the following packets do not.

This particular stream originates from 61.134.9.88 which resolves to:

```
inetnum:    61.134.3.0 - 61.134.20.95
netname:    SNXIAN
descr:      XI'AN DATA BUREAU
country:    CN
admin-c:    WWN1-AP
tech-c:     WWN1-AP
mnt-by:     MAINT-CHINANET-SHAANXI
mnt-lower:  MAINT-CN-SNXIAN
changed:    ipadm@public.xa.sn.cn 20010427
source:     APNIC
```

The top five other addresses resolve to the following two IP blocks:

```
inetnum:    61.153.17.0 - 61.153.17.255
netname:    NINGBO-ZHILAN-NET
descr:      NINGBO TELECOMMUNICATION CORPORATION ,ZHILAN APPLICATION SERVICE PROVIDER
descr:      Ningbo, Zhejiang Province
country:    CN
admin-c:    CZ61-AP
tech-c:     CZ61-AP
mnt-by:     MAINT-CHINANET-ZJ
changed:    master@dcb.hz.zj.cn 20010512
source:     APNIC
```

Atlantech Online, Inc. (NETBLK-AOI1999B)
1010 Wayne Avenue, Suite 630

Silver Spring, MD 20910

US

Netname: AOI1999B
Netblock: 209.190.192.0 - 209.190.255.255
Maintainer: ATON

Coordinator:
Center, Network Operations (EF105-ARIN) noc@atlantech.net
301-589-3060 (FAX) 301-593-9897

Domain System inverse mapping provided by:

DNS1.ATLANTECH.NET 209.183.205.35
DNS2.ATLANTECH.NET 209.183.192.65

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 22-May-2000.
Database last updated on 27-Nov-2001 19:55:19 EDT.

This attack may or may not have originated from this host. Most of the source addresses, however, are from Asia.

Reference
http://www.cert.org/advisories/CA-1996-01.html

Tiny Fragments – Possible Hostile activity
Fragmented packets are typically anomalous traffic used to evade IDS hosts and packet filtering devices. Fragments are not common and no commercial devices fragment packets to smaller than 256 bytes. This type of traffic is suspect.

There are two sources, and two destinations involved.

Source	**# Alerts (sig)**	**# Alerts (total)**
MY.NET.8.1	10306	10306
MY.NET.70.11	2	2
Destinations	**# Alerts (sig)**	**# Alerts (total)**
MY.NET.16.42	10306	10310
203.106.219.181	2	2

MY.NET.8.1 is very busy speaking exclusively to 16.42. MY.NET.70.11 is speaking exclusively to 203.106.219.181. It is obvious that 16.42 is under attack. Fragrouter or frel is probably listening on 8.1. Fragrouter and frel are programs that act as fragmentation routers to perform malicious activity.

To see if it is being used for this purpose, there will need to be a route added that uses 8.1 as a gateway. For example:

```
route add MY.NET.16.42 MASK 255.255.255.255 MY.NET.8.1
ping MY.NET.16.42
```

If the ping is successful, then 8.1 is acting as a fragmentation router.

Reference

http://www.sans.org/infosecFAQ/threats/frag_attacks.htm

UDP SRC and DST outside network

At first glance, one would assume that someone is doing some very obvious spoof scanning. The vast majority of the traffic flagged as having an external source and destination port is actually multicast traffic. However, there are some interesting events sprinkled here and there throughout the trace. For example:

```
10/26-14:54:21.960506  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 193.31.36.38:137
10/26-15:02:44.335967  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 82.103.189.240:137
10/26-15:02:51.893375  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 186.108.53.32:137
10/26-15:07:54.478873  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.93.103.240:137
10/26-15:15:47.748216  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 59.115.250.110:137
10/26-15:41:50.346983  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 124.185.54.137:137
10/26-15:47:35.485811  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.48.192.104:137
10/26-15:48:04.065116  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.106.208.207:137
10/26-15:57:46.609883  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 63.80.204.66:137
10/26-16:09:56.133590  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.29.141.77:137
10/26-16:19:35.663726  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.99.182.18:137
10/26-16:21:16.298404  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.112.69.189:137
10/26-16:27:25.462238  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.28.28.234:137
10/26-16:27:48.182005  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.154.155.30:137
10/26-16:36:29.433618  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.136.248.113:137
10/26-16:36:31.146728  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.136.248.113:137
10/26-16:58:30.452880  [**] UDP SRC and DST outside network [**] 169.254.171.77:137 -> 169.54.14.151:137
```

This traffic seems harmless enough, but close examination reveals that this appears to be a low and slow scan potentially targeting machines on the 169.* network. The first thing to notice is the source address. 169.254 is the net block that windows machines default to when they cannot reach a DHCP server. It appears that there is a host on My University's

network that is spewing out connections on a misconfigured interface. Further investigation reveals that this is probably the result of malware known as the network.vbs worm.

ICMP Echo Request speedera

On December 5, 2000, Joe Stewart released the following analysis of the Speedera pings that were being reported to various mailing lists. Initially, everyone thought that this was a distributed ping flood by Speedera, but Joe reported the following:

> The true source of the pings is Speedera.net's "Global Traffic Management" system. It isn't a random or sequential sweep of the net; the pings only occur when you make a DNS lookup request for one of their load-balanced cache customers' websites They then use the latency results of the distributed pings to return the IP address of the cache with the fastest route to you.

Closer examination of these traces indicate that the pings are originating from MY.NET and going to an @Home address at a rate of approximately five or six packets per second.

```
10/25-18:30:57.924100  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.182129  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.304077  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.488952  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.694001  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.734109  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:58.998986  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:59.163978  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:59.508780  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:59.538814  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:30:59.953731  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:00.305187  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:01.053560  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:01.096593  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:01.718450  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.173417  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.223322  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.308363  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.393301  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.643277  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:02.753226  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:03.193224  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
10/25-18:31:03.998047  [**] ICMP Echo Request speedera [**] MY.NET.208.246 -> 65.6.219.12
```

Regardless of what is intended to look like, this appears to be a denial of service ping flood that is designed to look like a Speedera ping. Speedera pings would appear more distributed and would not be sustained.

Reference

http://www.sans.org/y2k/121100-1200.htm

spp_http_decode: IIS Unicode attack detected - CVE-2000-0884

Published on Oct. 17, 2000 the UNICODE directory traversal has proven extremely devastating. The exploit is a malformed URL request that enables the sender to execute commands on a Microsoft web server. This vulnerability has been used in several worms such as Code Blue, and Nimda. An example would look like this:

http://target/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir
http://target/scripts/..%c0%9v../winnt/system32/cmd.exe?/c+dir

IIS parses this URL incorrectly and returns the executed "dir" command. The implications of this are enormous. This is a command prompt from anywhere. You could upload code with the TFTP command, or make a backup file of the SAM for offline cracking. The possibilities are endless. There were almost six thousand UNICODE attempts over five days. This is mostly remnant trojan activity, although there are some targeted attacks that utilize this exploit.

Reference
http://www.securityfocus.com/bid/1806

Watchlist 000220 IL-ISDNNET-990517
This is a long name for Gnutella traffic. Gnutella is a peer-to-peer technology that allows people to share files. It is most known for its distributed nature versus that of Napster. The alerts confirm this:

```
10/24-09:54:01.450502  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:54:07.558648  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:54:19.583844  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:54:49.956384  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:54:54.974474  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:55:11.955654  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:55:39.546664  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:55:41.544059  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:56:16.191841  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:56:20.540192  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:56:41.091717  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
10/24-09:56:42.965331  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.43.225:56237 -> MY.NET.234.198:6346
```

Notice that the destination port is that of a Gnutella host. Unfortunately, this traffic is common, yet is probably against the universities acceptable use policy.

Reference
http://www.sans.org/y2k/gnutella.htm

INFO MSN IM Chat data
This is Microsoft instant messenger traffic. Here is a sample:

```
10/25-00:33:01.806611  [**] INFO MSN IM Chat data [**] MY.NET.97.183:1525 -> 64.4.12.163:1863
10/25-00:33:50.371291  [**] INFO MSN IM Chat data [**] MY.NET.97.15:3909 -> 64.4.12.165:1863
10/25-00:34:05.633997  [**] INFO MSN IM Chat data [**] MY.NET.98.179:2453 -> 64.4.12.185:1863
10/25-00:34:12.509430  [**] INFO MSN IM Chat data [**] MY.NET.97.30:4142 -> 64.4.12.168:1863
```

Port1863/tcp is consistent with the MSN Instant Messenger client. This is probably safe to ignore unless university policy discourages chat clients.

WEB-MISC Attempt to execute cmd
This is the Code Red II worm trying to infect another host. It does this by copying cmd.exe from the system32 directory of a Windows NT/2000 machine to the scripts directory of the web server as root.exe. This allows anyone to come along and execute commands against the web server remotely (similar to the preceding worm discussion).

Most of the patterns within the logs files analyzed seem to be random in nature. Here is a sample:

```
10/25-00:50:27.456750  [**] WEB-MISC Attempt to execute cmd [**] 203.203.52.159:1659 -> MY.NET.218.198:80
10/25-00:50:33.846184  [**] WEB-MISC Attempt to execute cmd [**] 211.111.44.169:2141 -> MY.NET.204.197:80
10/25-00:50:37.483756  [**] WEB-MISC Attempt to execute cmd [**] 202.99.231.188:4056 -> MY.NET.243.103:80
10/25-00:51:01.841423  [**] WEB-MISC Attempt to execute cmd [**] 202.103.64.102:3793 -> MY.NET.220.160:80
10/25-00:51:19.471125  [**] WEB-MISC Attempt to execute cmd [**] 202.223.125.237:3260 -> MY.NET.215.200:80
10/25-00:51:53.004383  [**] WEB-MISC Attempt to execute cmd [**] 210.102.80.152:4941 -> MY.NET.241.253:80
10/25-00:51:56.556887  [**] WEB-MISC Attempt to execute cmd [**] 130.166.107.62:1103 -> MY.NET.240.13:80
10/25-00:52:20.702048  [**] WEB-MISC Attempt to execute cmd [**] 130.228.39.69:1376 -> MY.NET.229.83:80
```

This is consistent with the Code Red II worm.

Events of interest

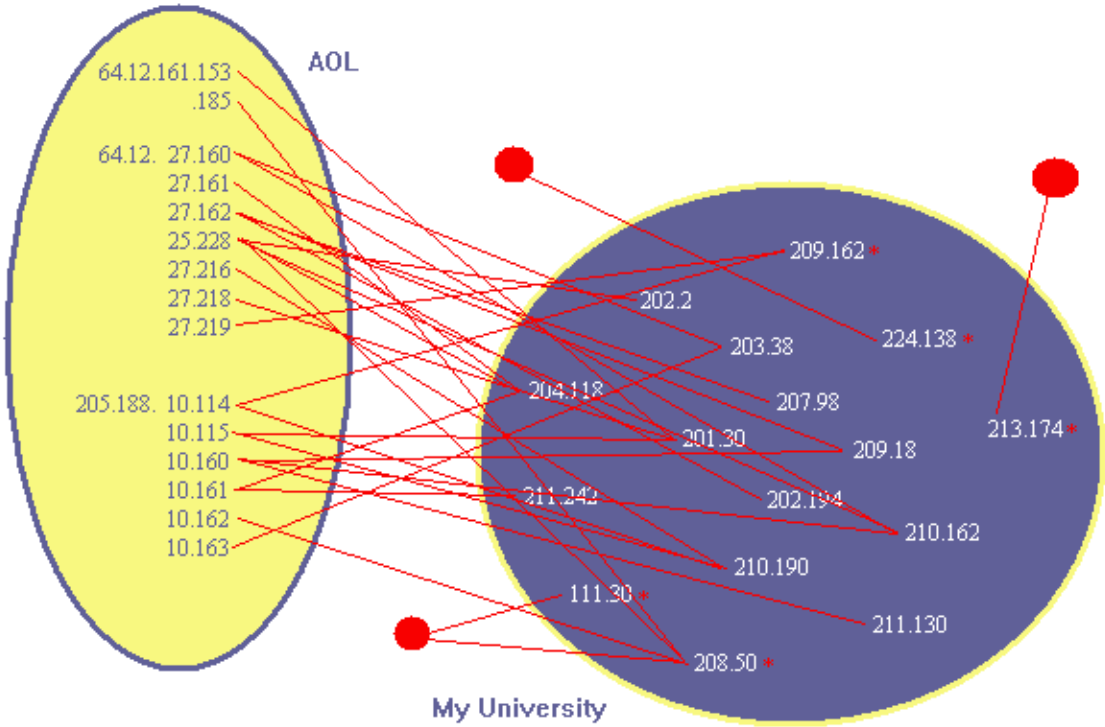
```
10/27-12:42:32.561241  [**] Possible trojan server activity [**] MY.NET.98.121:2463 -> 62.248.33.18:27374
10/27-12:42:32.738214  [**] Possible trojan server activity [**] MY.NET.98.121:2477 -> 62.85.129.148:27374
10/27-12:42:33.265896  [**] Possible trojan server activity [**] MY.NET.98.121:2478 -> 62.85.129.157:27374
10/27-12:42:34.684469  [**] Possible trojan server activity [**] 62.85.128.64:27374 -> MY.NET.98.121:2472
10/27-12:42:36.036398  [**] Possible trojan server activity [**] MY.NET.98.121:2484 -> 172.172.18.178:27374
10/27-12:42:38.053582  [**] Possible trojan server activity [**] MY.NET.98.121:2475 -> 62.85.128.92:27374
10/27-12:42:41.158134  [**] Possible trojan server activity [**] 62.85.129.157:27374 -> MY.NET.98.121:2478
10/27-12:42:45.100358  [**] Possible trojan server activity [**] MY.NET.98.121:2464 -> 62.29.114.29:27374
```

This machine is scanning hosts for the Subseven backdoor. The scan is slow enough to evade detection by the port scan preprocessor, but fortunately, that port is flagged by rule.

```
10/24-16:29:49.055553  [**] Possible trojan server activity [**] MY.NET.97.226:2007 -> 216.191.154.197:27374
10/24-16:29:50.895833  [**] Possible trojan server activity [**] MY.NET.97.226:2029 -> 216.191.154.220:27374
10/24-16:29:51.009121  [**] Possible trojan server activity [**] MY.NET.97.226:2039 -> 216.191.154.230:27374
10/24-16:29:54.370588  [**] Possible trojan server activity [**] MY.NET.97.226:2043 -> 216.191.154.234:27374
10/24-16:30:13.872322  [**] Possible trojan server activity [**] MY.NET.97.226:2035 -> 216.191.154.226:27374
10/24-16:30:16.961820  [**] Possible trojan server activity [**] MY.NET.97.226:2079 -> 216.191.155.16:27374
10/24-16:30:36.608739  [**] Possible trojan server activity [**] MY.NET.97.226:2086 -> 216.191.155.23:27374
10/24-16:30:36.816493  [**] Possible trojan server activity [**] MY.NET.97.226:2143 -> 216.191.155.81:27374
10/24-16:30:39.955584  [**] Possible trojan server activity [**] MY.NET.97.226:2144 -> 216.191.155.82:27374
10/24-16:30:46.279722  [**] Possible trojan server activity [**] MY.NET.97.226:2143 -> 216.191.155.81:27374
```

This machine is also scanning for the subseven backdoor. It is not randomizing the target scope like the previous scanner, yet the timing is consistent. These machines appear to know for what they are looking, and how best to find it. Is it possible these people know each other.

The last event of interest is really a large collection of events. These events transpired over the five days of logs like clockwork. It seems as though several hosts on My University's network are connecting to machines on AOL's netblock. These machines are connecting with TFTP (trivial file transfer protocol). If it were a worm that uses TFTP to propagate, the connection pattern would look much different. It helps to see this graphically.



Notice that the machines on MY.NET are connecting to multiple hosts at AOL. This is extremely odd. Why would they just connect to hosts at AOL? The asterisks denote hosts that exploited other hosts outside of MY.NET, or that had similar exploits ran against them. Notice the connection between 208.50/111.30. Both machines run the same exploit against the same machine and are also part of the TFTP maze.

```
10/28-14:59:06.701929  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.208.50:1037 -> 205.188.10.162:69
10/28-14:59:21.671905  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.208.50:1037 -> 205.188.10.162:69
10/28-15:00:11.118594  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.208.50:1037 -> 205.188.10.162:69

10/28-17:26:34.343433  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.208.50:1943 -> 216.239.35.100:80
10/28-17:26:34.343433  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.208.50:1943 -> 216.239.35.100:80
10/28-17:26:34.343433  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.208.50:1943 -> 216.239.35.100:80
10/28-17:26:34.343433  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.208.50:1943 -> 216.239.35.100:80
10/28-20:51:49.820924  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.111.30:2294 -> 216.239.35.100:80
10/28-20:51:49.822913  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.111.30:2296 -> 216.239.35.100:80
10/28-20:51:50.023362  [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.111.30:2301 -> 216.239.35.100:80
```

The odd thing about this trace relates to the times on the UNICODE connections from 208.50. This could be a worm. It is not likely because of the isolation of destinations and the correlation of attack events. The reason it does not look like a worm is that there is no apparent stimulus to the TFTP connections. The hosts seem to just act of their own will at different, almost random times. Something has to be initiating these connections. Here is a sample of the logs:

```
10/25-00:02:24.820176  [**] WEB-MISC Attempt to execute cmd [**] 130.166.107.62:4197 -> MY.NET.204.44:80
10/25-00:02:29.780073  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:38622 -> MY.NET.160.145:80
10/25-00:02:34.117250  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:48337 -> MY.NET.185.210:80
10/25-00:02:44.831654  [**] spp_http_decode: IIS Unicode attack detected [**] 211.92.76.56:3499 -> MY.NET.237.141:80
10/25-00:02:44.831654  [**] WEB-MISC Attempt to execute cmd [**] 211.92.76.56:3499 -> MY.NET.237.141:80
10/25-00:02:47.566389  [**] WEB-MISC Attempt to execute cmd [**] 211.92.76.56:3585 -> MY.NET.237.141:80
10/25-00:02:54.343352  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:48902 -> MY.NET.142.49:80
10/25-00:03:00.513639  [**] WEB-MISC Attempt to execute cmd [**] 64.171.186.248:2995 -> MY.NET.205.177:80
10/25-00:03:09.841448  [**] WEB-MISC Attempt to execute cmd [**] 211.91.14.5:4032 -> MY.NET.201.199:80
10/25-00:03:22.834156  [**] spp_http_decode: IIS Unicode attack detected [**] 211.90.176.59:54375 -> MY.NET.185.210:80
10/25-00:03:22.834156  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:54375 -> MY.NET.185.210:80
10/25-00:03:36.741386  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.210.190:1051 -> 205.188.10.114:69
10/25-00:03:45.886237  [**] WEB-MISC Attempt to execute cmd [**] 172.173.254.208:4452 -> MY.NET.215.195:80
10/25-00:03:50.825529  [**] spp_http_decode: IIS Unicode attack detected [**] 211.90.176.59:47530 -> MY.NET.142.49:80
10/25-00:03:50.825529  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:47530 -> MY.NET.142.49:80
10/25-00:03:57.440008  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.210.190:1055 -> 64.12.27.216:69
10/25-00:04:06.241801  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.210.190:1051 -> 205.188.10.114:69
10/25-00:04:09.205359  [**] spp_http_decode: IIS Unicode attack detected [**] 211.90.176.59:56254 -> MY.NET.106.80:80
10/25-00:04:09.205359  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:56254 -> MY.NET.106.80:80
10/25-00:04:22.306525  [**] TFTP - Internal TCP connection to external tftp server [**] MY.NET.224.138:3819 -> 66.8.139.42:69
10/25-00:04:36.430154  [**] WEB-MISC Attempt to execute cmd [**] 210.83.172.218:3759 -> MY.NET.202.75:80
10/25-00:04:45.027807  [**] spp_http_decode: IIS Unicode attack detected [**] 130.18.27.106:3302 -> MY.NET.211.222:80
10/25-00:04:45.027807  [**] WEB-MISC Attempt to execute cmd [**] 130.18.27.106:3302 -> MY.NET.211.222:80
10/25-00:05:02.377721  [**] TFTP - Internal TCP connection to external tftp server [**] 205.188.10.114:69 -> MY.NET.210.190:1051
10/25-00:05:07.874911  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:51403 -> MY.NET.203.57:80
10/25-00:05:35.534693  [**] WEB-MISC Attempt to execute cmd [**] 203.192.196.160:4475 -> MY.NET.69.134:80
10/25-00:05:40.983828  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:56461 -> MY.NET.69.115:80
10/25-00:05:41.436104  [**] WEB-MISC Attempt to execute cmd [**] 130.34.225.125:2075 -> MY.NET.216.219:80
10/25-00:05:48.964616  [**] spp_http_decode: IIS Unicode attack detected [**] 211.57.94.85:1629 -> MY.NET.204.197:80
10/25-00:05:48.964616  [**] spp_http_decode: IIS Unicode attack detected [**] 211.57.94.85:1629 -> MY.NET.204.197:80
10/25-00:05:48.964616  [**] spp_http_decode: IIS Unicode attack detected [**] 211.57.94.85:1629 -> MY.NET.204.197:80
10/25-00:05:48.964616  [**] WEB-MISC Attempt to execute cmd [**] 211.57.94.85:1629 -> MY.NET.204.197:80
10/25-00:05:53.704897  [**] WEB-MISC Attempt to execute cmd [**] 211.90.176.59:49150 -> MY.NET.53.252:80
10/25-00:05:55.404782  [**] spp_http_decode: IIS Unicode attack detected [**] 211.92.76.56:3511 -> MY.NET.98.239:80
10/25-00:05:55.404782  [**] spp_http_decode: IIS Unicode attack detected [**] 211.92.76.56:3511 -> MY.NET.98.239:80
10/25-00:05:55.404782  [**] spp_http_decode: IIS Unicode attack detected [**] 211.92.76.56:3511 -> MY.NET.98.239:80
10/25-00:05:55.404782  [**] WEB-MISC Attempt to execute cmd [**] 211.92.76.56:3511 -> MY.NET.98.239:80
10/25-00:06:03.919047  [**] spp_http_decode: IIS Unicode attack detected [**] 211.90.176.59:61966 -> MY.NET.179.246:80
```

Our TFTP connections are in red. Notice that there is no-worm related activities to stimulate this activity. There is not an incoming "cmd" or UNICODE exploit telling the host to connect back to some TFTP server; it just seems to be doing it.

Analysis

My University submitted the following files for analysis:

Alerts	OOS	Scans
alert.011024.clean	oos_Oct.24.2001	scans.011024.clean
alert.011025.clean	oos_Oct.25.2001	scans.011025.clean
alert.011026.clean	oos_Oct.26.2001	scans.011026.clean
alert.011027.clean	oos_Oct.27.2001	scans.011027.clean
alert.011028.clean	oos_Oct.28.2001	scans.011028.clean

Each of the alert files, OOS files, and scan files were condensed into three large files. The alert file was processed with SnortSnarf and custom Perl scripts. The scan file was also processed by Perl and imported into Microsoft Access. The Out of Spec file was used as a backdrop and primarily parsed with grep to correlate anomalies.

For instance, to find everything that happened at 9:44 – 9:47 on the 25th:

```
C:\Sans>grep 10/25-00:09:4[4-7] alerts.all
10/25-00:09:44.291595  [**] MISC Large UDP Packet [**] 61.153.17.188:52619 -> MY.NET.111.221:22264
10/25-00:09:44.794325  [**] MISC Large UDP Packet [**] 61.153.17.188:52619 -> MY.NET.111.221:22264
10/25-00:09:44.949440  [**] SMTP relaying denied [**] MY.NET.253.51:25 -> 144.126.188.201:1707
10/25-00:09:45.896414  [**] MISC Large UDP Packet [**] 61.153.17.188:1671 -> MY.NET.111.221:4298
10/25-00:09:46.582725  [**] MISC Large UDP Packet [**] 61.153.17.188:4277 -> MY.NET.111.221:27168
10/25-00:09:47.746257  [**] WEB-MISC Attempt to execute cmd [**] 24.249.90.20:1211 -> MY.NET.238.20:80
```

Recommendations

My University needs to consider some light border filtering. Most Universities want to remain as open as possible, but it is not practical anymore without putting yourself and others at risk.

My University should consider a firewall, that does simple filtering for known problems. For example, it should pass packet with a source address other than that of MY.NET. Ideally, My University should implement automated shunning utilizing FW1 and Snortsam. This would automatically shun offenders for a predefined amount of time as well as some basic perimeter filtering.

Appendix

correlate.pl

```
#-----------------------------------------------------------------------------------
open (INPUT, $ARGV[0]);

while (<INPUT>) {
       chomp;
   if (!m/portscan|UDP(.)SRC|Large(.)UDP/) {
      ($time, $attack, $addresses) = split(/ \[\*\*\] /);
      ($source, $destination) = split(/ -> /, $addresses);
      ($sourceip, $ephimeral) = split(/:/, $source);
      ($destip, $destport) = split(/:/, $destination);
      $targethash{$destip}++;
      $attackerhash{$sourceip}++;
   }
}

my @common = ();
foreach (keys %targethash) {
    print "$_, $targethash{$_}, $attackerhash{$_}\n" if exists $attackerhash{$_}
}

close INPUT;
#-----------------------------------------------------------------------------------
```

portscan.pl

```
#-----------------------------------------------------------------------------
open (INPUT, $ARGV[0]);

while (<INPUT>) {
   chomp;
   ($firsthalf, $secondhalf) = split(/ -> /);
   ($month, $day, $time, $sourceandport) = split(/ /, $firsthalf);
   ($source, $ephimeral) = split(/:/, $sourceandport);
   ($dest, $portandproto) = split(/:/, $secondhalf);
   ($port, $proto) = split(/ /, $portandproto);

   print "$source, $dest, $port, $proto\n";
}

close INPUT;


#-----------------------------------------------------------------------------
```

top_targets.pl

```
#-----------------------------------------------------------------------------------
open (INPUT, $ARGV[0]);

while (<INPUT>) {
  if (!m/portscan|UDP(.)SRC|Large(.)UDP/) {
        chomp;
    ($time, $attack, $addresses) = split(/ \[\*\*\] /);
    ($source, $destination) = split(/ -> /, $addresses);
    ($sourceip, $ephimeral) = split(/:/, $source);
    ($destip, $destport) = split(/:/, $destination);
    $hash{$destip}++;
  }
}
for (keys %hash) {
   if ($hash{$_} > 100) {print "$_, $hash{$_}\n";}
}

close INPUT;

#-----------------------------------------------------------------------------------
```

exploits.pl

```
while (<>) {
  if(!m/portscan|UDP(.)SRC|Large(.)UDP/) {
    chomp;
    split / \[\*\*\] /;
    $hash{"$_[1]"}++;
  }
}

@keys = reverse (sort{$hash{$a} <=> $hash{$b}} keys %hash);

for ($i=0; $i<$#keys; $i++) {
  printf "%-77s %-4d\n", $keys[$i], $hash{$keys[$i]};
}
```

Host	Dest	Src	Avg	Diff	Total
MY.NET.208.246	107	8345	1690.4	-8238	8452
61.134.9.88	34	2319	470.6	-2285	2353
MY.NET.14.1	70	1616	337.2	-1546	1686
211.90.176.59	5	1334	267.8	-1329	1339
MY.NET.225.98	188	839	205.4	-651	1027
MY.NET.207.22	149	625	154.8	-476	774
MY.NET.205.46	94	429	104.6	-335	523
MY.NET.153.111	1	412	82.6	-411	413
130.205.92.178	1	397	79.6	-396	398
130.18.27.106	1	353	70.8	-352	354
MY.NET.153.146	20	346	73.2	-326	366
MY.NET.85.111	4	240	48.8	-236	244
61.150.5.19	6	234	48	-228	240
MY.NET.99.39	24	204	45.6	-180	228
MY.NET.97.191	11	204	43	-193	215
130.166.14.51	1	200	40.2	-199	201
130.228.101.40	3	171	34.8	-168	174
MY.NET.234.198	151	156	61.4	-5	307
MY.NET.203.238	26	154	36	-128	180
MY.NET.253.51	8	154	32.4	-146	162
MY.NET.234.182	13	137	30	-124	150
MY.NET.228.182	3	116	23.8	-113	119
MY.NET.60.8	5	114	23.8	-109	119
MY.NET.233.226	8	109	23.4	-101	117
MY.NET.97.197	28	105	26.6	-77	133
MY.NET.97.168	14	102	23.2	-88	116
MY.NET.228.6	43	101	28.8	-58	144
MY.NET.98.121	12	96	21.6	-84	108
MY.NET.98.127	24	94	23.6	-70	118
MY.NET.253.125	7	94	20.2	-87	101
MY.NET.153.141	3	85	17.6	-82	88
MY.NET.153.197	11	80	18.2	-69	91
MY.NET.111.139	1	79	16	-78	80
MY.NET.97.249	114	78	38.4	36	192
217.80.150.185	46	78	24.8	-32	124
MY.NET.153.174	4	78	16.4	-74	82
MY.NET.156.106	2	77	15.8	-75	79
MY.NET.210.190	33	73	21.2	-40	106
MY.NET.207.46	11	72	16.6	-61	83
MY.NET.97.226	26	71	19.4	-45	97
MY.NET.233.234	159	68	45.4	91	227
205.188.10.115	9	66	15	-57	75
64.4.12.150	78	65	28.6	13	143
MY.NET.253.52	2	65	13.4	-63	67
MY.NET.223.118	1	65	13.2	-64	66
MY.NET.222.246	23	60	16.6	-37	83
64.4.12.172	101	58	31.8	43	159
MY.NET.53.37	5	56	12.2	-51	61
MY.NET.104.97	2	56	11.6	-54	58
MY.NET.209.162	2	54	11.2	-52	56
MY.NET.6.7	16	52	13.6	-36	68
MY.NET.98.167	12	52	12.8	-40	64
MY.NET.100.165	1616	50	333.2	1566	1666
MY.NET.238.182	28	49	15.4	-21	77
MY.NET.153.154	236	48	56.8	188	284
64.4.12.160	124	48	34.4	76	172
64.4.12.176	118	48	33.2	70	166
130.166.107.62	3	48	10.2	-45	51
MY.NET.98.220	7	47	10.8	-40	54
MY.NET.98.149	4	47	10.2	-43	51