Global Information Assurance Certification Paper

GCIA Practical Assignment

Version 3.0

Intrusion Detection in-Depth

Pedro Bueno

Assignment 1 – Describe the State of Intrusion Detection

LNIDS – A Free Linux Network Intrusion Detection

Introduction

Over the past few years, the security community began to understand that the idea to have a simple firewall in the network wasn't enough anymore. There was definitely a need for some security element that could be capable to alert the administrator about possible attacks. This element is what we know as the Intrusion Detection System.
The following figure (Figure 1), from Counterpane [1], is a good illustrative example.

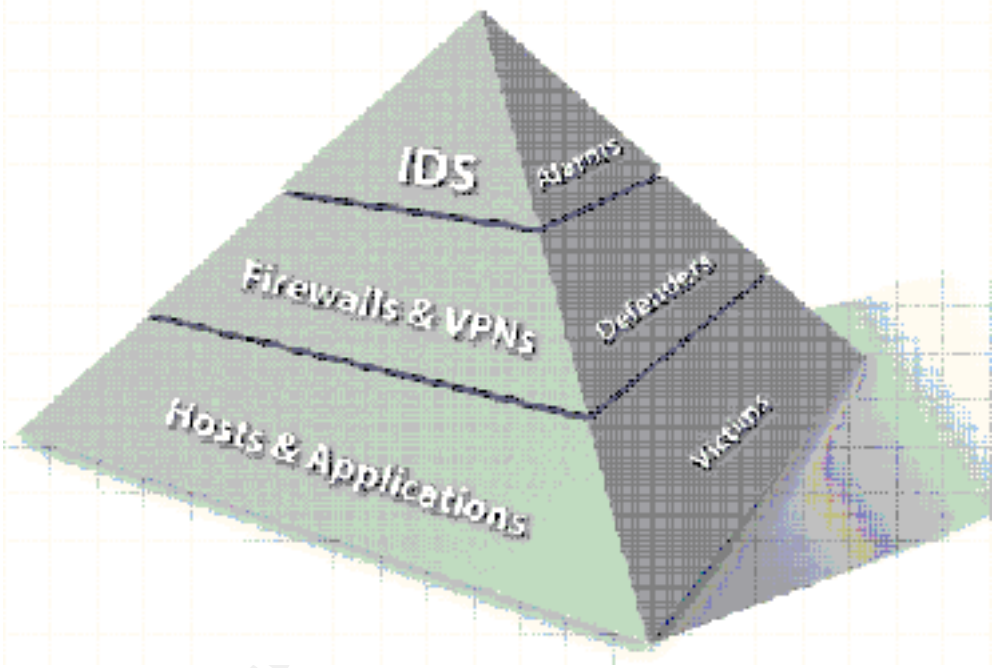


Figure 1

I believe that the best description of IDS is that it works like a burglar alarm. Rebecca Bace [2] explains that "The Firewall is the equivalent of a security fence around your property and the guard post at the front gate. It can keep the most unsavory of characters out, but cannot necessarily tell what is going on inside the compound. Intrusion Detection systems are the equivalent of multi-sensor video monitory and burglar alarm systems."
Someone once said that the best Intrusion Detection System is the one you full understand. After experienced various commercial NIDS, I discovered that the best NIDS is the one that I could customize and run in a system that I have plenty knowledge. For this reason, this paper is focused in *NIX system, Linux in particular. I will not try to create step-by-step guide to deploy and install tools, but instead, show how to get the best from your Linux box, using it as Network Intrusion Detection System.
The needs of hardening your box, choosing the best IDS software, deploying, managing and creating mechanisms to best handle the information generated is the goal of this paper.

Hardening your box

Ok, you decided to use your Linux box as IDS, but the very first thing to do is to be sure that the machine itself is not a point of failure in the network. Good options to create bastion machines are two different projects:

- Bastille-Linux [3]

Bastille Linux is a project created by Jay Beale and Jon Lasser. Jay Beale says [4] that "Bastille Linux is a project to harden, or 'lock down', Linux Systems". He states that Bastille Linux tries to "make a more secure environment for every class of user, without restricting them too much." It is really a great project that should be used by every Linux user. It currently supports the recent releases of Red Hat and Mandrake systems, and it is already in development for Debian, SuSE, TurboLinux and HP-UX systems. This is a good advantage against Openwall, which I will show next.

- Openwall [5]

The openwall project provides patches which are a "collection of security-related features for the kernel". These features, configurable as kernel options, are included as a lot of restriction to non-root users, as restricting access to the proc directory, for instance. In this case, non-root users can only see their own process unless they belong to a special group. This is also a great project, but in my opinion, I see two disadvantages here. The first one is that it is applied as kernel patches, so it's necessary for the administrator to have another level of expertise in Linux systems, to apply it and rebuild/install the new kernel. The second one is that it currently only supports Linux kernel 2.0 and 2.2 and not yet the 2.4 series.

Besides that, there are a lot of security guides, like SANS Securing Linux Step-by-Step [6] which teaches how to secure your linux machine but I believe that the minimum steps to guarantee this success are:

1. Being update with the latest vendor patches [6]

There are plenty guides which helps to apply this concept. The purpose of this item is to enforce the need of a good update policy to ensure that the system is not vulnerable for old attacks and is always updated.

2. Disable unnecessary services [7]

This is another step which is part of the most basic security guides and is directly related with the previous item. Imagine that your system has an *sshd* running, but you never noticed. The logical thought is that you will never upgrade it because you don't even know that it is running in your machine! So, picture the scenario that only in 2001, CVE [8] points 34 entries revealing vulnerabilities related *sshd,* and that you never upgraded your packages…Not good, huh?

The Types of IDS

As stated before, this paper will focus NIDS, the Network Intrusion Detection System, but a bit of explanation about the other IDS types are needed.
Marcus Ranum, in his Coverage in Intrusion Detection Systems [9] points the two primary types of IDS:

1. The Host-based IDS (HIDS) - which mainly consists in tools to correlate logs, check and verify file integrity and detect port maps. Good products examples are the ABACUS Suite from Psionic, Tripwire and SWatch.[10]. This kind of IDS is the one that should be installed all machines in the network, but if this is not possible, it must be installed at least in the main servers.
2. The Network-based IDS (NIDS) – consists in analyzing the network traffic, operating in a promiscuous mode, which makes possible to capture all traffic and not just the packets with its destination address. Based on his signatures, the NIDS can detect inappropriate traffic, like portscans, connections attempts, buffer overflow against a service and other known attacks. Also, the IDS can work to verify a policy violation, as access to porn content, or to measure the use of applications as Instant Message tools.

Recently, another type of IDS has emerging as a good option for those who want to have an HIDS with the proprieties of an NIDS. This solution is called Hybrid-IDS. I like the Marcus Ranum definition: "this approach basically makes each host run a mini-NIDS within its operating system environment."
The Prelude-IDS [11] is a good example of this kind of IDS. One good advantage of Prelude is that its signature engine can read snort rules, and according its website "by simply adding parsers, it should permit to load rulesets from any NIDs easily".

About the Software

As I will cover NIDS, I had to choose the best NIDS software. I could chose both commercial and non-commercial software, but since we have good free software and I intend to create a free Linux Network Intrusion Detection System, I choose non-commercial software, and my choice was Snort[12], the "lightweight IDS".

There are good reasons for choosing Snort as our NIDS. I believe that the first one is that it is Free! Ok, that it is free, but wouldn't have any good if it didn't work, but it does and very well. The best points that I like in Snort are the quick signatures deployment, the simplicity to create signatures process and the powerful control that it gives to the analyst.

In my point of view, besides the standard graphical interface and commercial support, there is no other difference between Snort and commercial vendors, since it also has Active response (or Flexible response) and Anomaly detection using the snort beta plug-in SPADE. By default Snort only comes with a non-graphical interface and uses the Linux shell as its command line interface, which is very powerful and allows you to create useful scripts and alerts, as we will see below.
The following applications mentioned here can substantially help in a successful NIDS deployment:

- ACID [13], is a graphical interface to handle multiple sensors. It is possible to receive information generated by the sensors and store in a database. In this way it makes easy to query the database for events, IP addresses and ports. Also it can display the packet that generated the alert in a graphical way, and even send alerts about the incidents.

Windows users that use Snort have the option to use Demarc [14], which is used to manage centralized sensors.
- SnortSnarf [15], from SiliconDefense which reads the snort alerts file and display it in html pages with statistics and information about the packets that generated the alerts.
- SnortSAM [16], an OPSEC plug-in which offers integration with the market vendor leader Firewall-1, from CheckPoint.

In the retaliatory battle field, we can also name some useful tools that can make easier (or harder) an administrator's life:

- Guardian [17], is a tool that reads the snort output and uses ipchains to deny any further packets from the attacker to get to the system.
- Snort2iptables [18], which can read the snort log file and add dynamic rules to an iptables ruleset.
- Blockit [19], very similar to guardian, but uses iptables to create its rules.
- Hogwash [20], a signature based firewall, based on Snort. This tool is particularly different from the tools above because instead of close or deny access to ports, "it drops or modifies specific packets based on a signature match".

As I said before, the administrator's life can be easier for the obvious reason, but why it can also be harder? The reason is that the attacker can spoof known IP addresses, as www.yahoo.com, or your own mail server and the tools like Blockit or Guardian, will block access to this addresses. So it must be tested with extreme care before deployment in the network.

Another software that could be very useful for the analyst is the SWatch [10], or Simple Log Watcher. Swatch is a tool that monitors the system messages, looking for unauthorized logins, error messages and other patterns, and alarms both using visual and sound alerts. It also can be easily configured to be used with Snort, by looking for snort alerts and send emails or execute whatever other application.

Deploying your LNIDS

Well, this definitely isn't an easy topic. Deploying IDS is always a reason for discussion in whatever book or mailing list about IDS. The reason is that everyone has their own reason to choose where to deploy it.
The question: Inside or Outside the Firewall? Simple, put it in all the places that you need to be aware about the traffic! Stephen Nortcutt´s Network Intrusion Detection's book [21] refers to it discussing the reasons for both choices. He stands that the IDS outside the firewall is a good place because it can detect attacks that could be blocked by the firewall. He also says that IDS inside the firewall can help to detect misconfigured firewalls, "if attacks get through that are supposed to be stopped, for example". Also, an IDS inside the firewall can inspect a VPN traffic which is decrypted at the firewall. As a final conclusion he says, and I completely agree, that more is better, or in other words, a sensor outside and a sensor inside the firewall could give a more complete solution for the analyst.

Another problem that may occur is when you decide to use it on a switched network. There are some schemas [22] that people use to plug in the IDS:

- Hubs – normally people use a hub to plug the IDS in, since it broadcasts all the traffic, in order to listen to all the incoming/outgoing traffic from the Internet.

- Mirroring Ports – using a spanning port at switches, it's possible to copy all traffic to a special port in the switches, called mirror port. Sometimes, due the lower speed of this port (comparing to the switch backplane [23]), the ids may not see all the traffic, letting packets go through without being noticed, and these packets may be part of an attack.

- Taps – the use of taps in the network could also be very useful. It could be placed between a switch and a resource, for example. Taps are usually unidirectional, passing the traffic from a resource and a switch directly to the IDS, preventing IDS traffic to the switch. Since the taps are not so discussed, the following picture(Figure-2), from ISS's Brain Laing [22] may be useful to understand:

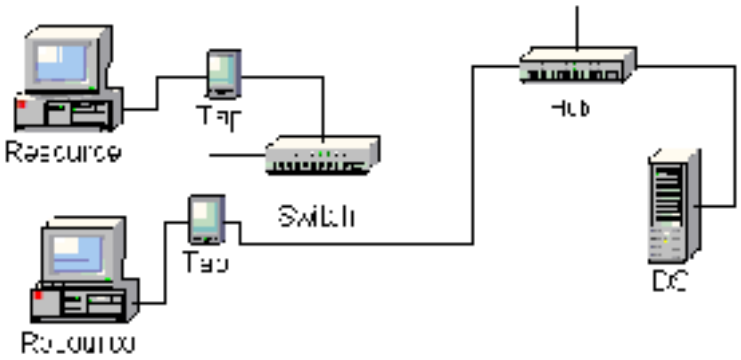


Figure 2

Manage your IDS

Imagine the scenario that your IDS is running perfectly ok, but a new attack is released. The next logical step is:

a) create yourself an signature to detect the attack
Or
b) upgrade your signatures with new packages

For both steps you will need to have access to your IDS sensor. Telnet and Ftp are not a good solution, since there are a lot of vulnerabilities associated with these services, so the use of Secure Shell is strongly recommended. But, in this case, an extra care must be taken, to just be aware and always upgrade the SSH daemon, due the recent vulnerabilities associated with it [8].

I use a schema that I believe that is a good way to deploy IDS. My LNIDS has two network interface cards. My goal is to watch only the traffic that come and go to the Internet, so I configured my snort only to listen to the interface plugged to external network, without an

IP address assigned to it. For the other interface I use an IP address of our internal network, so I can have access to it. In this case I use the first schema showed above, with a hub between the router and the firewall. Also, I used Linux's IPtables [24] to allow access only from my machine, so I can access SSH and HTTP from our internal network.

Signatures and the traffic generated

Ok. You have your LNIDS running, you can see the alerts, but how to handle with all the information that it generates?
The first thing that must be done is to customize the rules and try to reduce false-positives. As you can see in Snort Configuration File, there are lots of rules files, which are files with the Snort signatures for a service, as smtp.rules for example, which describe signatures for the SMTP service. If you don't run an SMTP server, you may think that wouldn't be necessary to include more 19 rules in the Snorts Rule set.
Also, another thing that people must have in mind is that if you want to be alerted when some signature is triggered in Snort, you definitely will have to trust on those signatures. Signatures based only on ports, or protocols, are examples that people must take care. Some false positives may appear in the system more frequently than you expect. I got my snort once alerting me with a signature about DDoS (Distributed Denial of Service). That is definitely not a good thing to see, but I had to check and I got just a normal http traffic...The signature that triggered that alert and generated this false positive was the following:

alert tcp $EXTERNAL_NET any -> $HOME_NET **20432** (msg:"DDOS shaft client to handler"; flags: A+; reference:arachnids,254; classtype:attempted-dos; sid:230; rev:1;)

Looking at the logs generated by Snort and correlated by tcpdump logs, I saw that a machine in my network using the port **20432** was receiving data from another machine in the internet using port **80.** Port 80 is the usual port for web servers, so the machine in my network which is my firewall, and was doing NAT (Network Address Translation), was using a high-numbered port, normal behavior for a client – server traffic, and coincidentally was the port number 20432, which triggered the signature.

But what about the amount of traffic generated by Snort? I like to create three different ways to handle with it: The Minimum, the Fast and the Full way.

- Minimum Way - I have an idea that minimum way to handle snort alerts would be using SnortSnarf. By putting Snort to send alerts to Syslog file, and using the Cro n scheduler to run SnortSnarf in pre-determined times, you can have a visual way (Html files) to check the alerts against your network and still verify statistics like the top source and top destination IP addresses

- Fast Way – The fast way is essentially the Minimum Way plus the use of Swatch to send emails or any other kind of alerts to the analyst about pre-determined alerts. If I have traffic involving well-known Trojan in my network, I definitely will want to be informed!

- Full Way – The Full way is a bit more complicated, because it involves database installation, besides the ACID software. In this way, you can have easy access to past data to correlate attacks and IP addresses. It displays statistics with graphics and can send alerts to the analyst. It also makes possible to query the database by using signatures or protocols types.

Note that I didn't include any of the retaliatory tools that I described above. I, particularly prefer the active response provided by Snort, which can be very useful, if used with caution.

Sandro Poppi [25] created the Snort-Setup for Statistics HOWTO. In his document, he shows how to setup Snort and configure some of the tools mentioned here, to get statistics from the snort IDS.

Conclusion

There are plenty IDS appliances and software vendors, and even Snort has its commercial approach with Silicon Defense's Sentarus[26] and also SourceFire's OpenSnort[27] Sensor. Snort, combined with tools like Swatch, SnortSnarf and others, can create very powerful NIDS in Linux environments. The ability to create plug-ins, like the OPSEC plug-in, makes Snort even more powerful. By running an NIDS in a system that you know and trust is half of a successful IDS implementation. Subjects like Sensor Placement and the way to handle the information generated must not be viewed here as standard because it will have to adequate to your network needs. The tools and theory showed in this paper can lead you to an effective Linux Network Intrusion Detection System (LNIDS) solution, allowing the intrusion analyst to take the best decision.

Bibliography

1. Counterpane Presentation – Pg 26
URL: http://www.counterpane.com/presentation2.pdf

2. An Introduction to Intrusion detection Assessment
URL: http://www.icsalabs.com/html/communities/ids/whitepaper/Intrusion1.pdf

3. Bastille-Linux Project
URL: http://www.bastille-linux.org

4. Bastille-Linux`s Jay Beale Interview
URL: http://www.linuxsecurity.com/feature_stories/feature_story-59.html

5. Openwall Project

URL: http://www.openwall.com

6. Securing Linux – Step-by-Step – Item 3.4
URL: http://www.sans.org/newlook/publications/linux_toc.htm

7. Beginner's guide to armoring Linux – Eliminating Services item
URL: http://www.enteract.com/~lspitz/linux.html

8. Common Vulnerabilities and Exposure (CVE)
URL: http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=sshd

9. Marcus Ranum – Coverage in IDS
URL: http://www.nfr.net/forum/white-papers/Coverage-in-IDS-White-Paper-final.pdf

10. Simple Log Watcher – Swatch
URL: http://www.stanford.edu/~atkins/swatch/

11. Prelude IDS
URL: http://www.prelude-ids.org

12. Snort IDS
URL: http://www.snort.org

13. Analysis Console for Intrusion Database (ACID)
URL: http://www.cert.org/kb/acid/

14. Demarc
URL: http://www.demarc.org

15. Using Snort v.18 with SnortSnarf on a RedHat Linux System
URL: http://rr.sans.org/intrusion/snortsnarf.php

16. SnortSAM – OPSEC Plug-in
URL: http://www.snortsam.net/

17. Snort2iptables
URL: http://dsli.hannover-internet.de

18. Guardian
URL: http://home.golden.net/~elim

19. Blockit
URL: http://blockit.teknofx.com

20. Hogwash
URL: http://hogwash.sourceforge.net

21. Network Inrtrusion Detection, an Analyst Handbook
URL: http://www.newriders.com/books/title.cfm?isbn=0735710082

22. SANS ID FAQ
URL: http://www.sans.org/newlook/resources/IDFAQ/switched.htm

23. Robert Graham – IDS FAQ
URL: http://www.robertgraham.com/pubs/network-intrusion-detection.html

24. Linux IPtables HowTo
URL: http://www.linuxguruz.org/iptables/howto/iptables-HOWTO.html

25. Snort-Setup for Statistics HowTo
URL: http://www.lug-burghausen.org/projects/Snort-Statistics/t1.html

26. Silicon Defense
URL: http://www.silicondefense.com

27. SourceFire
URL: http://www.sourcefire.com

Assignment 2 – Network Detects

Log Format

The data captured in this assignment was collected using the Snort IDS in a real world company in Brazil. All times showed in the logs are in GMT:-03:00.

The data was first analyzed using the Snort files created in the snort log directory. All information was also correlated with tcpdump log files which help to identify all the traffic related to attackers IP address.

Bellow, a brief description of the Log messages format from Snort Portscan File and TcpDump.

Snort Portscan Log Messages

Feb 28 15:01:53 61.120.12.42:21 -> xxx.xxx.xxx.37:21 SYNFIN ******SF

(1)	(2)	(3)	(4)	(5)

Field 1- Date and Time

Filed 2- Source IP Address : Source Port

Field 3- Direction of the packet

Field 4- Destination IP Address : Destination Port

Field 5- Flags Set in the Packet

Tcpdump Log Format

15:01:53.560000 61.120.12.42.ftp > xxx.xxx.xxx.27.ftp: SF [tcp sum ok]

(1)	(2)	(3)	(4)	(5)	(6)

Field 1- Time

Field 2- Source IP Address . Source Port

Field 3- Direction of the packet

Field 4- Destination IP Address . Destination Port

Field 5- Tcp Flags

Field 6- Integrity Check

629014454:629014454 (0) win 1028 (ttl 17, id 39426, len 40)

(7)	(8)	(9)	(10)	(11)	(12)

Field 7- Sequence Number

Field 8- Bytes in Packet

Field 9- Windows Size

Field 10- TTL – Time-To-Live

Field 11- IP ID

Field 12- Lenght

Intrusion Signatures and Analysis, Northcutt, Cooper, Fearnow and Frederick

Detect 1 - Port Scan/Retaliatory Action

Snort – Portscan Log File

```
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:169 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:248 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:1450 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:700 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:964 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:5192 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:336 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:598 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:102 SYN ******S*
Mar 11 15:04:10 200.206.132.138:60174 -> xxx.xxx.xxx.30:5979 SYN ******S*
Mar 11 15:04:22 200.206.132.138:60176 -> xxx.xxx.xxx.30:169 SYN ******S*
Mar 11 15:04:16 200.206.132.138:60175 -> xxx.xxx.xxx.30:248 SYN ******S*
Mar 11 15:04:16 200.206.132.138:60175 -> xxx.xxx.xxx.30:1450 SYN ******S*

Warping…

Mar 11 15:09:51 200.206.132.138:60174 -> xxx.xxx.xxx.30:6007 SYN ******S*
Mar 11 15:09:51 200.206.132.138:60174 -> xxx.xxx.xxx.30:1515 SYN ******S*
Mar 11 15:09:51 200.206.132.138:60174 -> xxx.xxx.xxx.30:9992 SYN ******S*
```

```
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:859 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:1469 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:244 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:620 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:268 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:9876 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:547 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:6007 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:1515 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60175 -> xxx.xxx.xxx.30:9992 SYN ******S*
Mar 11 15:09:57 200.206.132.138:60174 -> xxx.xxx.xxx.30:719 SYN ******S*
```

1-Source of Trace:

Real world network in Brazil

2- Detect was generated by:

Snort 1.8.3 and correlated by tcpdump logs

3- Probability the source address was spoofed:

Not likely. I will show in the item ´Description of the attack´, that this source address wasn't spoofed. The next item will show that there was a previous http traffic involving the same source address.

4- Description of the attack:

My first guess when I saw this snort Portscan log file was that it was just another portscan, but just to be sure, I decided to walkthrough the traffic. When I first checked my tcpdump logs, I saw the first trace on 13:51:00 (trace below).

```
13:51:00.120000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: S [tcp sum ok]
1918733600:1918733600(0) win 65535 <mss 1460,nop,wscale 0,nop,nop,timestamp
2060858 0> (ttl 63, id 10503, len 60)
13:51:00.160000 200.206.132.138.http > xxx.xxx.xxx.30.23915: S [tcp sum ok]
28009614:28009614(0) ack 1918733601 win 5792 <mss 1460,nop,nop,timestamp
58117978 2060858,nop,wscale 0> (DF) (ttl 53, id 0, len 60)
13:51:00.160000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: . [tcp sum ok] ack 1 win
65535 <nop,nop,timestamp 2060858 58117978> (ttl 63, id 10504, len 52)
```

```
13:51:00.160000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: P 1:258(257) ack 1 win
65535 <nop,nop,timestamp 2060858 58117978> (ttl 63, id 10505, len 309)
13:51:00.180000 200.206.132.138.http > xxx.xxx.xxx.30.23914: . [tcp sum ok] ack 259
win 6432 <nop,nop,timestamp 58117981 2060858> (DF) (ttl 53, id 64111, len 52)
13:51:00.220000 200.206.132.138.http > xxx.xxx.xxx.30.23915: . [tcp sum ok] ack 258
win 6432 <nop,nop,timestamp 58117984 2060858> (DF) (ttl 53, id 43985, len 52)
13:51:00.310000 200.206.132.138.http > xxx.xxx.xxx.30.23915: P 1:577(576) ack 258 win
6432 <nop,nop,timestamp 58117984 2060858> (DF) (ttl 53, id 43986, len 628)
13:51:00.310000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: F [tcp sum ok] 258:258(0)
ack 577 win 65535 <nop,nop,timestamp 2060858 58117984> (ttl 63, id 10511, len 52)
13:51:00.320000 xxx.xxx.xxx.30.23916 > 200.206.132.138.http: S [tcp sum ok]
1918884292:1918884292(0) win 65535 <mss 1460,nop,wscale 0,nop,nop,timestamp
2060858 0> (ttl 63, id 10515, len 60)

warping...
```

This definitely got my attention. In the first trace (Snort – Portscan Log file), the date of the portscan was 15:09:45. So, the portscan, which is a method to detect listening ports in the hosts, most of time sending packets with the SYN flag set, expecting to receive packets with SYN/ACK flags set, in fact happened. But wasn't just that.

5- Attack Mechanism:

This scan is called half-open scanning because the attacker doesn't complete the three way handshake, required to open a TCP connection. In this kind of scan, the attacker sends a packet with the SYN flag set, and then waits for the packet with a SYN/ACK flag set, to know that the destination port is listening. The common behavior of closed port is to send a packet with a RST flag set.

```
15:04:16.540000 200.206.132.138.60175 > xxx.xxx.xxx.30.169: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 35075, len 40)
15:04:16.540000 200.206.132.138.60175 > xxx.xxx.xxx.30.248: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 48071, len 40)
15:04:16.540000 200.206.132.138.60175 > xxx.xxx.xxx.30.1450: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 17738, len 40)
15:04:16.550000 200.206.132.138.60175 > xxx.xxx.xxx.30.700: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 23078, len 40)
15:04:16.550000 200.206.132.138.60175 > xxx.xxx.xxx.30.964: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 36754, len 40)
15:04:16.570000 200.206.132.138.60175 > xxx.xxx.xxx.30.5192: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 21382, len 40)
15:04:16.580000 200.206.132.138.60175 > xxx.xxx.xxx.30.336: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 37978, len 40)
15:04:16.580000 200.206.132.138.60175 > xxx.xxx.xxx.30.598: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 40982, len 40)
```

```
15:04:16.580000 200.206.132.138.60175 > xxx.xxx.xxx.30.iso-tsap: S [tcp sum ok]
1100094058:1100094058(0) win 3072 (ttl 47, id 21280, len 40)
15:04:16.580000 200.206.132.138.60175 > xxx.xxx.xxx.30.5979: S [tcp sum ok]

waping…
```

This was definitely a PortScan. But what about the initial trace, which started more than one hour before? Following the trace I saw normal http traffic between my network and the 200.206.132.138 web server which was initiated by my network!

The portscan was clearly created by an automated scan, due to high speed of the scan and aspects as IP ID , which are random and should be sequential, and the initial sequence number (ISN) which is constant and should be random for each new connection.

From the http traffic itself, aspects as ISN (Initial Sequence Number), source port numbers and IP ID are normal:

a) Source Port: Incrementing normally. At each new connection a new high numbered port (above 1024) is bound to the connection. In the trace, no anomaly could be noticed.
b) IP ID: Incrementing normally. At each new connection, the IP ID should be incremented by 1.

A nslookup to the webserver host show me nothing conclusive:

```
Server:         xxx.xxx.xxx.xxx
Address:        xxx.xxx.xxx.xxx#53

Non-authoritative answer:
138.132.206.200.in-addr.arpa        name = 200-206-132-138.dsl.telesp.net.br.

Authoritative answers can be found from:
```

A cable modem user hosted by Brazilian telecom provider Telefonica.

When I checked their website, I saw a webpage from a Web Design Company in Brazil.

So, I called them and related the portscan and the operator's answer was kind of funny. He told me that he noticed his IDS reporting a Denial of service from my ip address and he decided to run a Scanner to find out who we are…Good…

Checking our firewall logs, our operator told me that the connection was initiated by our cache server which was retrying to update a page in the site.

6- Correlations

This explanation can be correlated by the tcpdump logs:

```
15:14:58.600000 200.206.132.138.http > xxx.xxx.xxx.30.45965: P 1:577(576) ack 258 win 6432
<nop,nop,timestamp 58621760 2070930> (DF) (ttl 53, id 28117, len 628)

        E..tm.@.5.k.....
        ..T..P..>Q......
        ..............~.@
        ....HTTP/1.1.301
        .Moved.Permanent
                        ly

15:14:58.600000 xxx.xxx.xxx.30.45965 > 200.206.132.138.http: F [tcp sum ok] 258:258(0) ack 577
win 65535 <nop,nop,timestamp 2070932 58621760> (ttl 63, id 29048, len 52)

        E..4qx..?.....T.
        .......P....>Q.)
        ....j...........
        .~.@

15:14:58.610000 xxx.xxx.xxx.30.46077 > 200.206.132.138.http: S [tcp sum ok]
2534795550:2534795550(0) win 65535 <mss 1460,nop,wscale 0,nop,nop,timestamp 2070932 0>
(ttl 63, id 29052, len 60)

        E..<q|..?.....T.
        .......P........
        ................
        ..........

15:14:58.620000 200.206.132.138.http > xxx.xxx.xxx.30.46188: F [tcp sum ok] 577:577(0) ack 259
win 6432 <nop,nop,timestamp 58621793 2070912> (DF) (ttl 53, id 51721, len 52)

        E..4..@.5.......
        ..T..P.l=e.....n
        ..............~.a
                        ....

15:14:58.620000 xxx.xxx.xxx.30.46188 > 200.206.132.138.http: . [tcp sum ok] ack 578 win 0 (ttl 63,
id 29053, len 40)

        E..(q}..?..%..T.
        .....l.P...n=e..
        P.............

15:14:59.300000 xxx.xxx.xxx.30.46077 > 200.206.132.138.http: S [tcp sum ok]
2534795550:2534795550(0) win 65535 <mss 1460,nop,wscale 0,nop,nop,timestamp 2070934 0>
(ttl 63, id 29061, len 60)

        E..<q...?.....T.
        .......P........
        ................
```

```
          ..........

15:14:59.370000 200.206.132.138.http > xxx.xxx.xxx.30.46077: S [tcp sum ok]
1057134707:1057134707(0) ack 2534795551 win 5792 <mss 1460,nop,nop,timestamp 58621841
2070932,nop,wscale 0> (DF) (ttl 53, id 0, len 60)

          E..<..@.5.....
          ..T..P..?..s....
          .....n.........
          .~.........

15:14:59.370000 xxx.xxx.xxx.30.46077 > 200.206.132.138.http: . [tcp sum ok] ack 1 win 65535
<nop,nop,timestamp 2070934 58621841> (ttl 63, id 29062, len 52)

          E..4q...?.....T.
          .......P....?..t
          ................
                          .~..

15:14:59.370000 xxx.xxx.xxx.30.46077 > 200.206.132.138.http: P 1:258(257) ack 1 win 65535
<nop,nop,timestamp 2070934 58621841> (ttl 63, id 29063, len 309)

          E..5q...?.....T.
          .......P....?..t
          ....}e.........
          .~..GET./favicon
          .ico.HTTP/1.0..A
                          cc
```

This excerpt shows the server returning an error message *301 moved permanently* for the request *GET ./favicon.ico. HTTP/1.0* .This file, favicon.ico is that small icon that IE5 shows beside the url at the favorites folder.Internet Explorer 5 assumes that every web server must have it and insists to download every time that someone puts the url at the IE favorites folder. Our cache server then, waited less than 1(one) second and tried again…Apparently, the SQUID also have problems related this, and sometimes didn't cache this icon.

Favicon.ico Info
http://www.wdvl.com/Authoring/Design/Images/Favicon/

Apache error messages
http://www.htmlcenter.com/tutorials/tutorials.cfm?ID=150&type=general

7- Evidence of active targeting:

The port scanning was definitely trace of active targeting.

8- Severity

Criticality	4 – My firewall
Lethality	2 – Not the attack itself, but the scan is the first warning

	of the attack
System Countermeasures	5 – My firewall doesn't have any external open port.
Network Countermeasures	5 – The machine is my firewall

Using the formula learned in class:

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

(4 + 2) – (5 + 5) = Severity

Severity = -4

9- Defensive recommendation:

The use of ACLs in my Router should be used to restrict traffic that is not allowed in my network.

10 Multiple choice test question:

13:51:00.120000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: S [tcp sum ok] 1918733600:1918733600(0) win 65535 <mss 1460,nop,wscale 0,nop,nop,timestamp 2060858 0> (ttl 63, id 10503, len 60)
13:51:00.160000 200.206.132.138.http > xxx.xxx.xxx.30.23915: S [tcp sum ok] 28009614:28009614(0) ack 1918733601 win 5792 <mss 1460,nop,nop,timestamp 58117978 2060858,nop,wscale 0> (DF) (ttl 53, id 0, len 60)
13:51:00.160000 xxx.xxx.xxx.30.23915 > 200.206.132.138.http: . [tcp sum ok] ack 1 win 65535 <nop,nop,timestamp 2060858 58117978> (ttl 63, id 10504, len 52)

This trace consists:

a) Slow SynFlood
b) SynScan
c) Normal Tcp handshake
d) Half Scan

Answer: c)

Detect 2 - SYN/FIN PortScan FTP

Snort – Portscan Log File

```
Feb 28 15:01:53 61.120.12.42:21 -> xxx.xxx.xxx.30:21 SYNFIN ******SF
Feb 28 15:01:53 61.120.12.42:21 -> xxx.xxx.xxx.32:21 SYNFIN ******SF
Feb 28 15:01:53 61.120.12.42:21 -> xxx.xxx.xxx.33:21 SYNFIN ******SF
Feb 28 15:01:53 61.120.12.42:21 -> xxx.xxx.xxx.37:21 SYNFIN ******SF

Warping…

Feb 28 15:01:54 61.120.12.42:21 -> xxx.xxx.xxx.60:21 SYNFIN ******SF
Feb 28 15:01:54 61.120.12.42:21 -> xxx.xxx.xxx.61:21 SYNFIN ******SF
Feb 28 15:01:54 61.120.12.42:21 -> xxx.xxx.xxx.62:21 SYNFIN ******SF
Feb 28 15:01:54 61.120.12.42:21 -> xxx.xxx.xxx.63:21 SYNFIN ******SF
```

TcpDump Log File

```
15:01:53.560000 61.120.12.42.ftp > xxx.xxx.xxx.27.ftp: SF [tcp sum ok]
629014454:629014454(0) win 1028 (ttl 17, id 39426, len 40)
15:01:53.800000 61.120.12.42.ftp > xxx.xxx.xxx.30.ftp: SF [tcp sum ok]
629014454:629014454(0) win 1028 (ttl 17, id 39426, len 40)
15:01:53.800000 61.120.12.42.ftp > xxx.xxx.xxx.32.ftp: SF [tcp sum ok]
629014454:629014454(0) win 1028 (ttl 17, id 39426, len 40)

Warping…

15:01:54.050000 61.120.12.42.ftp > xxx.xxx.xxx.51.ftp: SF [tcp sum ok]
629014454:629014454(0) win 1028 (ttl 17, id 39426, len 40)
15:01:54.100000 61.120.12.42.ftp > xxx.xxx.xxx.52.ftp: SF [tcp sum ok]
316446246:316446246(0) win 1028 (ttl 17, id 39426, len 40)

Warping…

15:01:54.390000 61.120.12.42.ftp > xxx.xxx.xxx.63.ftp: SF [tcp sum ok]
316446246:316446246(0) win 1028 (ttl 17, id 39426, len 40)
15:01:57.900000 61.120.12.42.ftp > xxx.xxx.xxx.46.ftp: SF [tcp sum ok]
629014454:629014454(0) win 1028 (ttl 17, id 39426, len 40)
```

1-Source of Trace:

Real world network in Brazil

2- Detect was generated by:

Snort 1.8.3 and correlated by tcpdump logs

3- Probability the source address was spoofed:

It is hard to say. As I will show in the item ´Attack Mechanism´, these are obviously crafted packets, and the tool could also spoof the source ip address. But, the purpose of scans like that is to receive answer from the hosts in my network, to detect listening FTP servers. So, there are another possibility that the attacker could be in the same network as the source ip address and sniffing the network to see the replies from my machines.

4- Description of the attack:

This attack is an attempt to detect ftp servers in my network. The aspect that makes this attack different of a usual scan is that it uses both SYN and FIN flags together in order to receive answer from open ports. Typically, Linux machines answer this request with an SYN/FIN/ACK packet, so it can also be used to OS fingerprinting.

Once the attacker finds a listen FTP port, he/she can easily apply the proper exploit to get root access or use this FTP server as a hacker/mp3 repository.

The FTP service is know to have multiple vulnerabilities, as reported Gary Portner in his practical, and currently, the CVE database reports 154 vulnerabilities associated with it.
http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=ftp

Gary Portner Detect 1
http://www.giac.org/practicals/Gary_Portnoy.zip

Linux – SYN/FIN Request
http://lists.netspace.org/cgi-bin/wa?A2=ind9807B&L=bugtraq&P=R2441

5- Attack Mechanism

The purpose of this attack with the flags SYN and FIN turned on, is to evade filtering devices [SANS Conference Track3 – Pattern Analysis], since some filtering devices only block unsolicited packets with the ACK flag.

Some patterns in the trace make me believe that the packets were crafted with an automated tool:

- The IP ID number of the packets

The IP ID of the packets is constant and shouldn't be. The normal behavior should be incrementing the IP ID by 1

- The initial sequence number

The initial sequence number is quite strange too. The normal behavior should be a random initial sequence number being generated in each new connection. In this detect we can see only two different ISNs of all the scan

- High speed of the Scan

Scripts kiddies or Scanners can do a very fast scan in a very large range of ip address. In this detect we can see that it can scan the average number of 15 hosts per second.

Another detail that should be considered is that the source port is a ´low port´, below 1024, which means that the tool must be run by a user with root privileges. This detail is important because we can assume that:

a) The machine was compromised by a hacker which have super user privileges
b) The attack was done by the machine's owner, which consist in an inappropriate user activity

The SynScan tool is a tool that can produce craft packets with the pattern that I found in these detects:

-IP ID: 39426
-Source port = destination port

More SynScan Information can be found here:
http://archives.neohapsis.com/archives/snort/2000-11/0445.html

6- Correlations:

The port scan is also associated with an attack because it can be interpreted as the reconnaissance for a next step. However, at the time of writing this paper there was no new bug associated with the ftp server service.
A post to Security Focus Incidents List by Nicolas Gregoire on June 19th, 2001, shows the exactly same detect:
The URL: http://lists.jammed.com/incidents/2001/06/0142.html

```
From: Nicolas Gregoire (nicolas.gregoire@7thzone.com)
Date: Tue Jun 19 2001 - 02:25:34 PDT
Hi,

here some logs from probes done by compromised boxes.
The first one (hacked_1) is a default RedHat 6.2 and the second one
(hacked_2) is a default Cobalt 5.0
Admins have been notified.

Jun 17 21:23:22 my_box_1 snort[468]: SCAN-SYN FIN: hacked_1:511 ->
my_box_1:511
Jun 17 21:23:22 my_box_2 snort[5207]: SCAN-SYN FIN: hacked_1:511 ->
my_box_2:511

Jun 18 20:52:42 my_box_2 snort[5207]: SCAN-SYN FIN: hacked_2:21 ->
my_box_2:21
Jun 18 20:52:42 my_box_1 snort[468]: SCAN-SYN FIN: hacked_2:21 ->
my_box_1:21

...

The first one has a root shell binded to port 511, not the second one.
The strange thing is that these 2 boxes are located in France, like me,
and have the same patterns.
```

```
Every packet have the same values for a few fields :
TOS:0x0 ID:39426 IpLen:20 DgmLen:40 Win: 0x404  TcpLen: 20

Have you ever seen that ?

Nicob
(please excuse my english)
```

A look at Incidents Top Ten Ports shows often that the port 21 is always in the top attacked ports.

Dshield Top Ten Ports:
http://www.dshield.org/topports.html

7- Evidence of active targeting:

The scans don't look to be a direct attack since most of the addresses on the ip range are not in use.

8- Severity

Criticality	4 – The Web server, DNS proxy and Mail proxy are in the range
Lethality	1 – Not the attack itself, but the scan is the first warning of the attack
System Countermeasures	5 - Not running FTP server
Network Countermeasures	1 – The machines are outsite of the firewall perimeter

Using the formula learned in class:

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

(4 + 1) – (5 + 1) = Severity

Severity = -1

9- Defensive recommendation:

Defenses are ok, since we don't run FTP server. Defenses can be planned if we start to use this service. Also, a stateful host based firewall should be considered in the machines outside of the firewall perimeter.

10 Multiple choice test question:

15:01:53.560000 61.120.12.42.ftp > xxx.xxx.xxx.27.ftp: SF [tcp sum ok] 629014454:629014454(0) win 1028 (ttl 17, id **39426**, len 40)

15:01:53.800000 61.120.12.42.ftp > xxx.xxx.xxx.30.ftp: SF [tcp sum ok] 629014454:629014454(0) win 1028 (ttl 17, id **39426**, len 40)
15:01:53.800000 61.120.12.42.ftp > xxx.xxx.xxx.32.ftp: SF [tcp sum ok] 629014454:629014454(0) win 1028 (ttl 17, id **39426**, len 40)

In the trace above, the IP ID is an anomaly because:

a) Should be random all the time
b) Should increment a least by 1
c) It is not an anomaly. It is the normal behavior.
d) Should be decremented by 1

Answer: b)

Detect 3 – DNS Version Bind - port 53

TcpDump - Log File

```
06:02:16.830000 24.168.138.15.4659 > xxx.xxx.xxx.62.domain: S [tcp sum ok]
1305458799:1305458799(0) win 32120 <mss 1460,sackOK,timestamp 46246899
0,nop,wscale 0> (DF) (ttl 44, id 38997, len 60)
06:02:16.840000 24.168.138.15.4630 > xxx.xxx.xxx.33.domain: . [tcp sum ok] ack 1 win
32120 <nop,nop,timestamp 46246919 552423363> (DF) (ttl 44, id 39014, len 52)
06:02:16.920000 24.168.138.15.1304 > xxx.xxx.xxx.33.domain:  [udp sum ok] 20160 TXT
CHAOS)? version.bind. [|domain] (ttl 44, id 39016, len 58)
06:02:16.920000 xxx.xxx.xxx.33.domain > 24.168.138.15.1304:  [udp sum ok] 20160*-
1/0/0 version.bind. CHAOS) TXT 9.1.0 (48) (DF) (ttl 63, id 0, len 76)
06:02:17.110000 24.168.138.15.1304 > xxx.xxx.xxx.33.domain:  20160 inv_q+
[b2&3=0x980] (465) (ttl 44, id 39018, len 493)
06:02:17.110000 xxx.xxx.xxx.33.domain > 24.168.138.15.1304:  [udp sum ok] 20160
inv_q ServFail- [0q] 0/0/0 (12) (DF) (ttl 63, id 0, len 40)
06:02:17.280000 24.168.138.15.4630 > xxx.xxx.xxx.33.domain: F [tcp sum ok] 1:1(0) ack
1 win 32120 <nop,nop,timestamp 46246964 552423363> (DF) (ttl 44, id 39019, len 52)
06:02:17.280000 xxx.xxx.xxx.33.domain > 24.168.138.15.4630: F [tcp sum ok] 1:1(0) ack
2 win 5792 <nop,nop,timestamp 552423426 46246964> (DF) (ttl 63, id 50764, len 52)
06:02:17.450000 24.168.138.15.4630 > xxx.xxx.xxx.33.domain: . [tcp sum ok] ack 2 win
32120 <nop,nop,timestamp 46246981 552423426> (DF) (ttl 44, id 39020, len 52)
06:02:19.640000 24.168.138.15.4624 > xxx.xxx.xxx.27.domain: S [tcp sum ok]
1312788362:1312788362(0) win 32120 <mss 1460,sackOK,timestamp 46247198
0,nop,wscale 0> (DF) (ttl 44, id 39382, len 60)
06:02:19.650000 24.168.138.15.4634 > xxx.xxx.xxx.37.domain: S [tcp sum ok]
1314227108:1314227108(0) win 32120 <mss 1460,sackOK,timestamp 46247198
0,nop,wscale 0> (DF) (ttl 44, id 39391, len 60)
06:02:19.650000 24.168.138.15.4632 > xxx.xxx.xxx.35.domain: S [tcp sum ok]
```

```
1309612923:1309612923(0) win 32120 <mss 1460,sackOK,timestamp 46247198
0,nop,wscale 0> (DF) (ttl 44, id 39389, len 60)

Warping...
```

Snort Log File

```
[**] DNS named version attempt [**]
03/27-06:02:16.920000 0:6:53:3:7E:20 -> 0:10:DB:8:9C:C1 type:0x800 len:0x48
24.168.138.15:1304 -> xxx.xxx.xxx.33:53 UDP TTL:44 TOS:0x0 ID:39016 IpLen:20 DgmLen:58
Len: 38
4E C0 00 00 00 01 00 00 00 00 00 00 07 76 65 72  N...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

[**] DNS named version attempt [**]
03/27-06:02:16.920000 0:6:53:3:7E:20 -> 0:10:DB:8:9C:C1 type:0x800 len:0x48
24.168.138.15:1304 -> xxx.xxx.xxx.33:53 UDP TTL:44 TOS:0x0 ID:39016 IpLen:20 DgmLen:58
Len: 38
4E C0 00 00 00 01 00 00 00 00 00 00 07 76 65 72  N...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

1-Source of Trace:

Real world network in Brazil

2- Detect was generated by:

Snort 1.8.3 and correlated by tcpdump logs

3- Probability the source address was spoofed

Not likely. By watching the snort logs, I could only see the UDP traffic, which could be easily spoofed packets. But when I was looking at the TcpDump logs, I could see a previous TCP traffic, mostly tcp packets with SYN flag turned on, expecting to receive answer from the hosts in my network, to detect listening DNS servers. As I will show at the item ´Description of the attack´, it will receive answer from my DNS proxy and attempt to query its version bind.

4- Description of the attack

This attack is an attempt to find listening DNS (Domain Name Service) servers, port 53, in my network and query its version bind. As we could see, the attacker was first sending tcp packets with the SYN flag turned on, to find hosts with the port 53 listening. Once it finds a listening port 53, receiving an SYN/ACK, typical step 2 at the three way handshake, it tried to discover the version of the DNS server:

```
06:02:16.920000 24.168.138.15.1304 > xxx.xxx.xxx.33.domain:  [udp sum ok] 20160 TXT
CHAOS)? version.bind. [|domain] (ttl 44, id 39016, len 58)
```

Receiving the answer form my host:

```
06:02:16.920000 xxx.xxx.xxx.33.domain > 24.168.138.15.1304:  [udp sum ok] 20160*-
1/0/0 version.bind. CHAOS) TXT 9.1.0 (48) (DF) (ttl 63, id 0, len 76)
```

Just to complete the information, in this case, he/she is using the udp protocol which is the standard for queries. The DNS server also accepts tcp packets when it tries give a large answer for the query, or for Zone transfers.

Once the attacker gets the version of the Bind server, it can try multiple exploits associated with the version. CERT points 18 advisories about BIND.

This query is very easy to create, using applications as dig, or the very common nslookup.

```
[root@bigbrother root]# nslookup
Note:  nslookup is deprecated and may be removed from future releases.
Consider using the dig' or host' programs instead.  Run nslookup with
the -sil[ent]' option to prevent this message from appearing.
> server xxx.xxx.xxx.33
Default server: xxx.xxx.xxx.33
Address: xxx.xxx.xxx.33#53
> set class=chaos
> set query=txt
> version.bind
Server:         xxx.xxx.xxx.33
Address:        xxx.xxx.xxx.33#53

version.bind  text = "9.1.0"
>
```

Nslookup Model

```
[root@bigbrother root]# dig @xxx.xxx.xxx.33 version.bind chaos txt

; <<>> DiG 9.1.3 <<>> @xxx.xxx.xxx.33 version.bind chaos txt
;; global options:  printcmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2875
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;version.bind.                  CH      TXT

;; ANSWER SECTION:
version.bind.           0       CH      TXT    "9.1.0"

;; Query time: 101 msec
;; SERVER: xxx.xxx.xxx.33#53(xxx.xxx.xxx.33)
;; WHEN: Fri Apr 12 14:54:04 2002
;; MSG SIZE  rcvd: 48

[root@bigbrother root]#
```

Dig Model

CERT Advisories:
URL:http://search.cert.org/query.html?col=certadv&ht=0&qp=&qt=bind&qs=&qc=&pw=100%25&ws=1&la=en&qm=0&st=1&nh=25&lk=1&rf=2&rq=0&si=1
Bind Security
URL: http://www.isc.org/products/BIND/bind-security.html

5- Attack Mechanism

Other aspects as ISN (Initial Sequence Number), source port numbers and IP ID are apparently normal:

c) Source Port: Incrementing normally. At each new connection a new high numbered port is bound to the connection. In the trace, no anomaly could be noticed.
d) IP ID: Incrementing normally. At each new connection, the IP ID should be incremented by 1.
e) The initial sequence number. The normal behavior should be a random initial sequence number being generated in each new connection.

These packets seem to be crafted. The reasons are:

- High speed of the Scan,

Scripts kiddies or Scanners can do a very fast scan in a very large range of ip address.

Also, we could see that the tool tries to find a listening for DNS servers and immediately after it finds one, it tries to get the version.

6- Correlations

The port scan is also associated with an attack because it can be interpreted as the reconnaissance for a next step as we could see. However, at the time of writing this paper there was no new bug associated with the Bind server.
A post to Security Focus Incidents List by Erik Finchtner on December 29th, 1999, shows the same behavior:

The URL: http://cert.uni-stuttgart.de/archive/incidents/1999/12/msg00000.html

Re: Large number of BIND probes.

- *To*: INCIDENTS@SECURITYFOCUS.COM
- *Subject*: Re: Large number of BIND probes.
- *From*: Erik Fichtner <techs@obfuscation.org>
- *Date*: Wed, 29 Dec 1999 22:19:48 -0500

```
On Wed, Dec 29, 1999 at 10:56:24AM -0600, Craig H. Rowland wrote:
> Last night I received a very large number of probes to TCP port 53 on
one
> of my DNS servers. I filter out all TCP port 53 traffic to these
systems
> except for secondaries/primaries for zone transfers, but would like to
> know if anyone has seen a pickup in this activity as well. There were
six
> separate hosts yesterday alone that tried. This is a new record for me.
> Seems like there may be a new BIND attack going around. :(

I only got hit with one set of version.bind probes last night, and only
on
the primary nameservers listed with the InterNIC, but they were rather
persistant at trying to figure out if it was a bind4 or a bind8 with
version.bind shut off..

The attacking domain's security contact confirmed that the machine on
their
end was compromised (how, i have no idea).


--
Erik Fichtner; Warrior SysAdmin (emf|techs)
34.9908%
http://www.obfuscation.org/~techs      N 38 53.055'  W 77 21.860'  764
ft.
       "What's the most effective Windows NT remote management tool?"
          "A car."  --  Stephen Northcutt
```

7- Evidence of active targeting:

The scans don't look to be a direct attack since most of the addresses on the ip range are not in use, but the tool checked the only host which responded its SYN packet, so it can also be interpreted as active targeting.

8- Severity

Criticality	4 – The DNS proxy are in the range
Lethality	2 – Not the attack itself, but the scan is the first warning of the attack, as well the query. Also, the attacker could see the version of running bind
System Countermeasures	3 – No bug associated with this version of bind, but a new bind version is available
Network Countermeasures	1 – The machines are outsite of the firewall perimeter

Using the formula learned in class:

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

(4 + 2) – (3 + 1) = Severity

Severity = 2

9- Defensive recommendation:

There are some defensive recommendations that can be applied in this case. To avoid this situation, where the attacker can get the version of running bind in the network, the administrator can include a version "unknow version" statement on the named.conf file, usually located at /etc directory. This would look like:

```
[root@bigbrother  root]# cat /etc/named.conf
// generated by named-bootconf.pl

options  {
        directory "/var/named";
        version "unknown version";
};

…
```

Besides this, since the version is 9.0.1, and there is a new version available at ISC, an upgrade is recommended.

Also, a more secure configuration should be done, like restrict queries only to authorized ip addresses, as well the zone transfers directive.

ISC – bind9
http://www.isc.org/products/BIND/bind9.html

10- Multiple choice test question:

The TCP and UDP packets observed in port 53 traffic are:

a) An anomaly. It is not normal
b) TCP is not acceptable for DNS traffic
c) UDP is not acceptable for DNS traffic
d) Could be normal, depending of the traffic needs.

Answer: d)

Detect 4 - LPRng attempt- port 515

Snort – Portscan Log File

```
Feb 28 18:55:50 211.252.136.190:1702 -> xxx.xxx.xxx.37:515 SYN ******S*
Feb 28 18:55:50 211.252.136.190:1695 -> xxx.xxx.xxx.30:515 SYN ******S*
Feb 28 18:55:50 211.252.136.190:1713 -> xxx.xxx.xxx.48:515 SYN ******S*
Feb 28 18:55:50 211.252.136.190:1722 -> xxx.xxx.xxx.57:515 SYN ******S*
Feb 28 18:55:50 211.252.136.190:1728 -> xxx.xxx.xxx.63:515 SYN ******S*
```

TcpDump Log File

```
18:55:50.830000 211.252.136.190.1702 > xxx.xxx.xxx.37.printer: S [tcp sum ok]
3621788119:3621788119(0) win 32120 <mss 1460,sackOK,timestamp 14213890
0,nop,wscale 0> (DF) (ttl 41, id 44234, len 60)
18:55:50.840000 211.252.136.190.1695 > xxx.xxx.xxx.30.printer: S [tcp sum ok]
3611981330:3611981330(0) win 32120 <mss 1460,sackOK,timestamp 14213890
0,nop,wscale 0> (DF) (ttl 41, id 44227, len 60)
18:55:50.840000 211.252.136.190.1713 > xxx.xxx.xxx.48.printer: S [tcp sum ok]
3605905395:3605905395(0) win 32120 <mss 1460,sackOK,timestamp 14213890
0,nop,wscale 0> (DF) (ttl 41, id 44245, len 60)
18:55:50.840000 211.252.136.190.1722 > xxx.xxx.xxx.57.printer: S [tcp sum ok]
3607489491:3607489491(0) win 32120 <mss 1460,sackOK,timestamp 14213890
0,nop,wscale 0> (DF) (ttl 41, id 44254, len 60)
18:55:50.840000 211.252.136.190.1728 > xxx.xxx.xxx.63.printer: S [tcp sum ok]
3610987969:3610987969(0) win 32120 <mss 1460,sackOK,timestamp 14213891
0,nop,wscale 0> (DF) (ttl 41, id 44260, len 60)
```

1-Source of Trace:

Real world network in Brazil

2- Detect was generated by:

Snort 1.8.3 and correlated by tcpdump logs

3- Probability the source address was spoofed:

I would say the same as for detect 2, but this time aspect as IP ID and ports are apparently normal, but it is also possible that it was spoofed. This time the attacker was expecting to receive answer from the hosts in my network, to detect listening Printer-515 ports, usually associated with the line printer spooler in *NIX machines.

4- Description of the attack:

This attack is an attempt to find a specific open port number in my network. In this case, the specific port is the Printer port number 515, mostly associated with the the line printer spooler, LPR in *NIX machines. LPRng, which is a rewrite of Berkeley LPR, claims to have great enhanced security checks, but it is still a very dangerous service to be available in the internet.

Nyheter Trojan Port List also list port 515 as used by the lpdw0rm, which installs backdoors with no password associated and with root privileges.

Lpdw0rm:
http://www.simovits.com/trojans/tr_data/y984.html

Currently, CVE lists 5 LPR references
http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=lpr
and 6 LPRng references
http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=lprng

SANS – Trojan Port List
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm

5- Attack Mechanism:

This scan likely to be a half-open scanning because the attacker doesn't complete the three way handshake, required to open a TCP connection. In this kind of scan, the attacker sends a packet with the SYN flag set, and then waits for the packet with a SYN/ACK flag set, to know that the destination port is listening. The common behavior of closed port is to send a packet with a RST flag set. Unfortunately, since none of the targets were alive, I couldn't assume this.

If the attacker finds a listening 515 port he/she has plenty exploits available on the internet to compromise the machines and have root access on it.

Doing a Passive OS FingerPrinting in this traces of the packet made me believe that the attacker was using a Linux, kernel version 2.2, Operating System:

a) TCP Options = MSS, SackOK, WindowScale, Timestamp, one NOP
b) WindowSize = 32120
c) Packet Length = 60 bytes

Other aspects as ISN (Initial Sequence Number), source port numbers and IP ID are apparently normal:

f) Source Port: Incrementing normally. At each new connection a new high numbered port is bound to the connection. In the trace, no anomaly could be noticed.
g) IP ID: Incrementing normally. At each new connection, the IP ID should be incremented by 1.

LPR Exploits
http://astalavista.box.sk/cgi-bin/robot?srch=lpr+exploit

Passive OS FingerPrinting
http://www.incidents.org/papers/OSfingerprinting.php

6- Correlations

The port scan is also associated with an attack because it can be interpreted as the reconnaissance for a next step. An Arrigo Triulzi post at Incidents.Org List shows a snort log detect of port 515 scan.

```
From: Arrigo Triulzi [mailto:arrigo@northsea.sevenseas.org]
Sent: Tuesday, November 06, 2001 6:12 AM
To: intrusions@incidents.org
Subject: Repeated lpr exploit attempts from 142.232.68.3


Copy of text sent off to the admins of the site (British Columbia
Institute of Technology).  Excuse the feeble attempts at humour but it
is a cold day in London and a rather boring one too.

Arrigo

-*-*-

Dear Sir,

the host at IP address 142.232.68.3 which belongs to your network has
```

```
managed to attempt to break my lpr daemons 2565 times in the past day
which is a remarkable achievement.

In particular here is a selection of his/her exploits (times are in
GMT, NTP-locked to a stratum 2 server).  First a scan for port 515
across all my hosts:

Nov  6 03:43:22 eolo snort: [1:120005:1] ULTRA -  TCP closed port
access attempt 54-1024 [Classification: Traffic not in Web Whitelist]
[Priority: 6]: {TCP} 142.232.68.3:2022 -> 212.18.234.50:515

[then onto scan every machine on my subnet, 212.18.234.48/29]

Then an attempt using the old RedHat 7 exploit:

Nov  6 03:43:55 eolo snort: [1:0:0]
IDS457/lpr_LPRng-redhat7-overflow-security.is [Classification: system
integrity attempt] [Priority: 11]: {TCP} 142.232.68.3: 4376 ->
212.18.234.50:515

[again over all my subnet...]

Nov  6 03:43:57 eolo snort: [1:0:0]
IDS457/lpr_LPRng-redhat7-overflow-security.is [Classification: system
integrity attempt] [Priority: 11]: {TCP} 142.232.68.3: 4981 ->
212.18.234.61:515

Then repeated once more, since the first one failed...

Nov  6 03:44:00 eolo snort: [1:0:0]
IDS457/lpr_LPRng-redhat7-overflow-security.is [Classification: system
integrity attempt] [Priority: 11]: {TCP} 142.232.68.3: 2061 ->
212.18.234.50:515

[again all over my subnet...]

Then a portscan to check if the ports are open...

Nov  6 03:44:01 eolo snort: spp_portscan: portscan status from
142.232.68.3: 6 connections across 3 hosts: TCP(6), UDP(0)

Ah, perhaps it isn't LPRng but standard LPR so let us try it all over
again for the whole subnet:

Nov  6 03:44:02 eolo snort: [1:302:1] EXPLOIT redhat 7.0 lprd overflow
[Classification: Attempted Administrator Privilege Gain] [Priority:
1]: {TCP} 142.232.68.3:2517 -> 212.18.234.60:515

Another portscan...

Nov  6 03:44:07 eolo snort: spp_portscan: portscan status from
142.232.68.3: 6 connections across 3 hosts: TCP(6), UDP(0)

And so on (repeat the above countless time, or rather 2000 and odd),
to get to the end:

Nov  6 03:45:14 eolo snort: spp_portscan: portscan status from
```

```
142.232.68.3: 2 connections across 1 hosts: TCP(2), UDP(0)
Nov  6 03:45:25 eolo snort: spp_portscan: End of portscan from
142.232.68.3: TOTAL time(78s) hosts(18) TCP(59) UDP(0)

Could you please investigate the machine in question and suggest to
the owner that if the exploit fails the first time repeating it
doesn't work (like a number of other things in life)?

Thanks,

Arrigo
```

7- Evidence of active targeting:

This is quite strange. None of the five connection attempts are or were ever been used in our network. It doesn't appear to be a direct attack.

8- Severity

Criticality	4 – The Web server, DNS proxy and Mail proxy are in the range
Lethality	1 – Not the attack itself, but the scan is the first warning of the attack
System Countermeasures	5 – The machines alive in the network didn't offer printer service.
Network Countermeasures	1 – The machines are outside of the firewall perimeter

Using the formula learned in class:

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

(4 + 1) – (5 + 1) = Severity

Severity = -1

9- Defensive recommendation:

In spite of the fact that, currently, there aren't open ports 515, printer service, in any of the machines network, it is recommended the use of ACLs in the router, blocking packets with destination port 515.

10 Multiple choice test question:

Which best Snort Rule could be used to alert about external connection attempts to port 515 in your network:

a) alert tcp $EXTERNAL_NET any -> $HOME_NET 515 (msg: “Printer port access attempt”; flags:A+;)

b) alert tcp $EXTERNAL_NET 515 -> $HOME_NET any (msg: “Printer port access attempt”; flags: A+;)

c) alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: “Printer port access attempt”; flags: A+;)

d) alert tcp $HOME_NET any -> $EXTERNAL_NET 515 (msg: “Printer port access attempt”; flags: A+;)

Answer: a)

Detect 5 - SunRPC Scan - port 111

Snort – Portscan Log File

```
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.49:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.50:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.37:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.39:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.41:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.45:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.46:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.48:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.47:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.51:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.40:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.36:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.42:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.43:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.44:111 SYN ******S*
Feb 20 23:25:32 202.108.95.200:111 -> xxx.xxx.xxx.38:111 SYN ******S*
```

TcpDump Log File

```
23:25:32.000000 202.108.95.200.sunrpc > xxx.xxx.xxx.49.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.000000 202.108.95.200.sunrpc > xxx.xxx.xxx.50.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.120000 202.108.95.200.sunrpc > xxx.xxx.xxx.37.sunrpc: S [tcp sum ok]
```

```
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.120000 202.108.95.200.sunrpc > xxx.xxx.xxx.39.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.120000 202.108.95.200.sunrpc > xxx.xxx.xxx.41.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.130000 202.108.95.200.sunrpc > xxx.xxx.xxx.45.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.130000 202.108.95.200.sunrpc > xxx.xxx.xxx.46.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.430000 202.108.95.200.sunrpc > xxx.xxx.xxx.48.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.520000 202.108.95.200.sunrpc > xxx.xxx.xxx.47.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.530000 202.108.95.200.sunrpc > xxx.xxx.xxx.51.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.580000 202.108.95.200.sunrpc > xxx.xxx.xxx.40.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.660000 202.108.95.200.sunrpc > xxx.xxx.xxx.36.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.660000 202.108.95.200.sunrpc > xxx.xxx.xxx.42.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.670000 202.108.95.200.sunrpc > xxx.xxx.xxx.43.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.670000 202.108.95.200.sunrpc > xxx.xxx.xxx.44.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.840000 202.108.95.200.sunrpc > xxx.xxx.xxx.38.sunrpc: S [tcp sum ok]
1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
```


1-Source of Trace:

Real world network in Brazil

2- Detect was generated by:

Snort 1.8.3 and correlated by tcpdump logs

3- Probability the source address was spoofed:

This is also hard to tell. The detect 2 shows the same behavior and my answer is the same as item 3 and detect 2, except that the attacker was expecting to receive answer from the hosts in my network, to detect listening port 111 , usually associated with the SUN Remote Procedure Calls or Portmap.

4- Description of the attack:

This attack is an attempt to find a specific open port number in my network. In this case, the specific port is the Portmap port number 111, mostly associated with portmap service on *NIX machines. Portmap is a RPC program number mapper. According to the Portmap Man Page, "when an RPC server is started, it will tell portmap what port number it is listening to , and what RPC program numbers it is prepared to serve".

In Incident.Org List email from Mike Manco illustrates that a packet with same from and destination ports , a constant IP ID, ttl minor than 128 and datagram length equal to 40, it is likely to be Synscan1.8 or greater.

Mike Manco email to Intrusions List
http://www.incidents.org/archives/intrusions/msg03126.html

Portmap Man Page
http://www.rt.com/man/portmap.8.html

5- Attack Mechanism:

This scan is likely to be a half-open scanning because the attacker doesn't complete the three way handshake, required to open a TCP connection. In this kind of scan, the attacker sends a packet with the SYN flag set, and then waits for the packet with a SYN/ACK flag set, to know that the destination port is listening. The common behavior of closed port is to send a packet with a RST flag set. Unfortunately, since none of the targets were alive, I couldn't assume this.

Once the attacker finds a listeing Portmap port, he/she can send queries about which ports are bound to the RPC services. RPC services are known to have serious vulnerabilities associated with them. Examples of RPC services are rpc.mountd, rpc.statd, rpc.lock…

Snort has about 48 signatures related to RPC services in the rpc.rules file. Once the attacker has information about the proper ports that applications are bound, he/she can explore it with the exploits available in the internet.

Unix systems offers an application called rpcinfo to verify either local or remote information about rpc process in the machines, by querying the portmap.

Using the rpcinfo application the attacker will get:

```
[root@bigbrother  tmp]# rpcinfo –p victim.in.the.net

  program vers proto   port
   100000    2   tcp    111  portmapper
   100000    2   udp    111  portmapper
   100011    1   udp    655  rquotad
   100011    2   udp    655  rquotad
   100011    1   tcp    658  rquotad
   100011    2   tcp    658  rquotad
   100005    1   udp  32806  mountd
   100005    1   tcp  39209  mountd
   100005    2   udp  32806  mountd
   100005    2   tcp  39209  mountd
   100005    3   udp  32806  mountd
   100005    3   tcp  39209  mountd
   100003    2   udp   2049  nfs
   100003    3   udp   2049  nfs
   100021    1   udp  32807  nlockmgr
   100021    3   udp  32807  nlockmgr
   100021    4   udp  32807  nlockmgr
```

Some patterns in the trace make me believe that the packets were crafted with an automated tool:

- The IP ID number of the packets

  The IP ID of the packets is constant and shouldn't be. The normal behavior should be incrementing the IP ID by 1

- The initial sequence number

  The initial sequence number is quite strange too. The normal behavior should be a random initial sequence number being generated in each new connection. In this detect we can see only one ISN of all the scan

Rpcinfo Man Page
http://campuscgi.princeton.edu/man?rpcinfo

CVE currently points 33 vulnerabilities associated with RPC
http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=rpc

RPC.statd vulnerabilities
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0666
http://www.securityfocus.com/bid/1480

6- Correlations

The port scan is also associated with an attack because it can be interpreted as the reconnaissance for a next step. In this case, the attacker probably would get information about the RPC process running in the machines. Snort has currently about 48 signatures associated with RPC. Since I don't have RPC process running and the attacker couldn't receive any answer, it is hard to tell the attackers intent. A Derek Kwan post at Security Focus Incident List shows how the community still receives plenty Portmap scan.

From: Derek Kwan (dkwan@KWAN.CA)
Date: Mon Jan 15 2001 - 13:54:38 CST

Yes I have seen alot of sunrpc scan on my cable modem too.

Since Jan 1, 2001 I get appx 3-4 sunrpc scan daily. Here are a list of IPs
for sunrpc scan on my server since 1 Jan 2001.

216.128.39.125
208.35.4.25
216.253.248.140
24.108.84.147
24.70.222.168
24.22.169.216
24.167.61.7
152.101.127.222
211.172.14.13
211.75.16.178
160.78.31.151
211.100.8.165
211.5.191.200
64.2.219.110

Also there is a scan from 24.0.0.203 (authorized-scan1.security.home.net)
on port 119 atleast 2-3 times daily too. Does other cable modem user have
a similiar scan on their machine?

\|/ _____ \|/ **********
"@'/ , . \`@" This e-mail is send with 100% recyclable electrons.
/_| ___/ |__\ **********
___U_/ Derek@KWAN.ca

On Sun, 14 Jan 2001, Steve Buttgereit wrote:

> I'm beginning see a lot, too. All different IPs though. I'm also seeing a
> lot of scans in my snort log that follow this pattern: FIN scan to port
> 111 --> RPC Info. Query --> RPC portmap-request status --> Shellcode x86
> NOPS. It all started about a week ago.
>
> SCB
> -----Original Message-----
> From: Jason Lewis [mailto:jlewis@JASONLEWIS.NET]
> Sent: Sunday, January 14, 2001 10:20 PM
> To: INCIDENTS@SECURITYFOCUS.COM

> *Subject: Re: anyone else seen an increase in sunrpc scans these days?*
> *I couldn't find any of those addresses, but I have similar scans in my logs.*
> *63.91.6.36*
> *64.32.209.213*
> *64.21.114.2*
> *66.22.62.2*
> *216.98.160.251*
> *Last 24 hours....all the above IP's are looking for Sun RPC.*
> *jas*
> *http://www.rivalpath.com*

Also, an email from Jens Hektor , at Incidents list, on April 24th 2001, shows an increase in the number or portmap probes received. Richard Grant also posted an email to Security Focus Incidents List pointing multiple probes to portmap port.

Jens Hektor email
http://lists.jammed.com/incidents/2001/04/0112.html

Richard Grant email
http://cert.uni-stuttgart.de/archive/incidents/2001/03/msg00124.html

7- Evidence of active targeting:

It doesn't appear to be a direct attack. Just one machine was alive in the network range used by the scanner. It could be probably part of a large scan.

8- Severity

Criticality	4 – The Web server, DNS proxy and Mail proxy are in the range
Lethality	1 – Not the attack itself, but the scan is the first warning of the attack
System Countermeasures	5 – The machines are just running webserver, DNS and SMTP services.
Network Countermeasures	1 – The machines are outsite of the firewall perimeter

Using the formula learned in class:

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

(4 + 1) – (5 + 1) = Severity

Severity = -1

9- Defensive recommendation:

In spite of we don't run rpc services in our network, or in the machines outside the firewall, is recommended to include ACLs to block port 111, both TCP and UDP at the border router.

10 Multiple choice test question:

23:25:32.000000 202.108.95.200.sunrpc > xxx.xxx.xxx.50.sunrpc: S [tcp sum ok] 1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)
23:25:32.120000 202.108.95.200.sunrpc > xxx.xxx.xxx.37.sunrpc: S [tcp sum ok] 1284063461:1284063461(0) win 26436 (ttl 117, id 25211, len 40)

In the trace above, we can consider strange fields:

a) TTL , IP ID and SRC/DST Ports
b) ISN and IP ID
c) ISN , TTL and IP ID
d) IP ID, SRC/DST Ports and ISN

Answer: d)

Assignment 3 – Analyze This

Executive Summary

First of all, I would like to thank you for the opportunity to analyze the logs collected by the Intrusion Detection System in our University. The analysis of the data generated by your IDS will try to give you substantial information about the security issues related your network. This is an important step, but not the only level of the analysis. After these analyses, one more important step is needed: the appropriate defensive recommendation for your network.

This document is divided in the follow parts:

Introduction – In this phase I will give some introductory explanation about the files analyzed.

Methodology – In this phase, I will give some explanation about the methodology used in the analysis.

Network Assumptions – based on the information gathered on the analysis, I could do some assumptions about your network.

Alert Analysis – In this phase, I will show the analysis results of the alert files generated by your Snort IDS.

Scan Analysis – In this phase, I will show the analysis results of the Scan files generated by your Snort IDS.

Out-of-Spec Analysis – In this phase, I will show the analysis results of the Out of Specification files generated by your Snort IDS.

Defensive Recommendation – In this phase, after received all the data generated by the previous analysis, I will indicate some defensive acts that would improve your network security.

Introduction

The analysis consists of 5 consecutive days of your files in the time period of January 27th and January 31st. I have the chance to look at 2.413964 scans, 1.074.704 Alerts and 72 lines of Out of Spec files (OOS), 4 in total (Table 1). By using manual and automates tools to do the analysis, this report will not analyze every single alert generated , but show a list sorted by priority, based on factors like number of alerts generated and severity. Also, I will show

a “Top 10 list” with the ten most active source addresses and ten most active destination ports.

The files collected at: http://www.research.umbc.edu

-rw-r--r--	1 pbueno	pbueno	8850279	Mar 13 11:57 alert.020127.gz
-rw-r--r--	1 pbueno	pbueno	29315675	Mar 13 11:57 alert.020128.gz
-rw-r--r--	1 pbueno	pbueno	27415978	Mar 13 11:58 alert.020129.gz
-rw-r--r--	1 pbueno	pbueno	29296349	Mar 13 11:58 alert.020130.gz
-rw-r--r--	1 pbueno	pbueno	34488420	Mar 13 11:58 alert.020131.gz
-rw-r--r--	1 pbueno	pbueno	1067	Mar 13 12:01 oos_Jan.27.2002.gz
-rw-r--r--	1 pbueno	pbueno	1261	Mar 13 12:01 oos_Jan.28.2002.gz
-rw-r--r--	1 pbueno	pbueno	0	Mar 13 12:01 oos_Jan.29.2002.gz
-rw-r--r--	1 pbueno	pbueno	36	Mar 13 12:02 oos_Jan.30.2002.gz
-rw-r--r--	1 pbueno	pbueno	0	Mar 13 12:02 oos_Jan.31.2002.gz
-rw-r--r--	1 pbueno	pbueno	12268335	Mar 13 15:05 scans.020127.gz
-rw-r--r--	1 pbueno	pbueno	37695697	Mar 13 15:05 scans.020128.gz
-rw-r--r--	1 pbueno	pbueno	32283978	Mar 13 15:05 scans.020129.gz
-rw-r--r--	1 pbueno	pbueno	31908843	Mar 13 15:05 scans.020130.gz
-rw-r--r--	1 pbueno	pbueno	38483673	Mar 13 15:06 scans.020131.gz

Table 1

Summary of Alerts:

Timeframe	: 01/27/02 -> 01/31/02
Total Alerts	: 1.074.704
Total Scans	: 2.413.964
Total OOS	: 4

Methodology

In the analysis of these data, I had to deal with large amount of data. As I showed before, the files were very large, like the portscan and alerts file (120 and 152 Mb, respectively). I first tried put all the scans files together and run SnortSnarf to generated friendly html files. As the result, my machine couldn’t handle with the data and my RAM memory wasn’t enough. So I decided to use SnortSnarf in each different file to collect some info and also use the common *NIX application as uniq, cut, awk, grep, cat and sort.
I also had a chance to test and modify some scripts used by Lenny Zeltser [9] SANS practical, also used by Dennis Ruck [8] in his practical.

Manual Analysis

For the manual analysis I first had to put everything together:

cat alert.020127.gz alert.020128.gz alert.020129.gz alert.020130.gz alert.020131.gz > alerts.univ

The same for the scans files, except that I had to edit the alert files to cut out the headers.

Then I had to use awk, cut and sort to order the exact fields that I want, using command like, for instance:

awk ´$6 {print $6}´ scans.IP > file

cut –d ´:´ -f 2 scans.IP | sort | uniq –c > scan.IP.fields

cat scans-fields | sort –r –n > scans-fields-sort

SnortSnarf

To get the alerts generated by the alert files, I had to use snortsnarf for each day of alerts, to then make the sum to generate the top ten alerts.
After that, I had to use the same command line tools to get info like the top ten and count.

Network Assumptions

As I didn't have your network topology or any other information about your network, I had to do some assumptions about your servers:

The machines:

MY.NET.1.3
MY.NET.1.4
MY.NET.1.5
Are your DNS servers

The machines:

MY.NET.1.4
MY.NET.1.7
Are your NTP servers

The machines:

MY.NET.253.114

Is working as a web server. In Defensive recommendations, I put some other information about this machine

MY.NET.150.198
Is working as a print server.

Alert Analysis

The alerts used in this Analysis started at January 27th and finished at January 31st

Top Ten Alerts generated:

Top	Signatures	Count
1	Connect to 515 from inside	97974
2	Spp_http_decode: IIS Unicode attack detected	83769
3	Misc UDP large packet	48820
4	SMB name Wildcard	40670
5	SNMP public access	24286
6	ICMP Echo Request L3retriever Ping	20107
7	Spp_http_decode: CGI Null Byte attack detected	11263
8	High port 65535 udp – possible Red Worm Traffic	7073
9	INFO MSN IM Chat data	6197
10	Watchlist 000220 IL-ISDNNET-990517	3454

Table 2

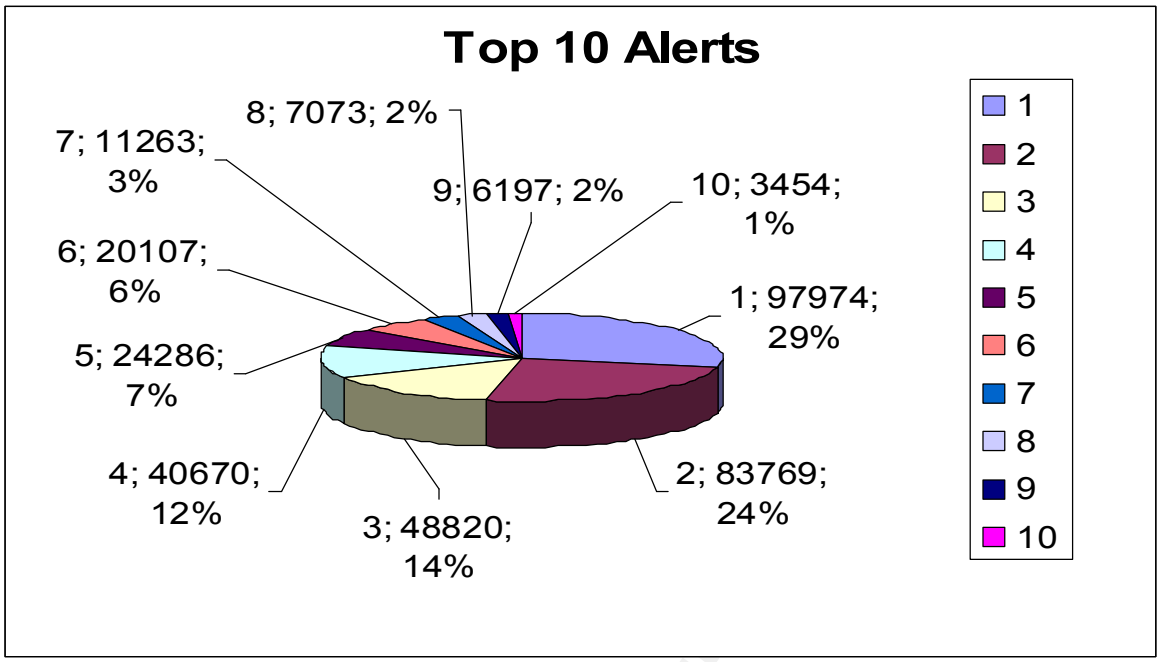


Table 3

Top 10 attackers – General

Count	IP addresses
15396	MY.NET.153.119
12634	MY.NET.70.177
10739	63.250.209.34
9859	63.250.211.165
9751	MY.NET.153.111
9496	63.250.209.88
8928	MY.NET.11.6
8748	MY.NET.153.114
8347	MY.NET.153.126
8308	MY.NET.153.122

Alert #1 – Connect to 515 from inside

Top 10 Attackers

Count	IP addresses
12152	MY.NET.153.119
9136	MY.NET.153.111
8202	MY.NET.153.126
8133	MY.NET.153.114

5214	MY.NET.153.118
4209	MY.NET.153.122
3712	MY.NET.153.113
3299	MY.NET.153.120
2892	MY.NET.88.148
2556	MY.NET.153.115

The Alert Example:

```
01/31-23:27:49.512712  [**] connect to 515 from inside [**] MY.NET.153.110:1241 ->
MY.NET.150.198:515
```

All the top ip addresses are from the internal network. This attack is trigged by a Snort signature that looks for connects from local network ip addresses to local network ip address on port 515. From the alert file is difficult to tell about external address connects to MY.NET machines. Correlating this port with the Scan files, I could notice that there are some external ip addresses that also probed port 515 in MY.NET. Snort ruleset has two rules that detect exploit to LPRng from EXTERNAL_NET to HOME_NET port 515. I believe that the reason that Snort didn't triggered alerts from this attack is that the router or the firewall may be blocking port 515 TCP from external to MY.NET.

Port 515 is the LPR port number and it is know to have plenty vulnerabilities associated with it. Mitre CVE [10] reports 9 entries associated with both lpr and lprng.

By looking at the destination IP addresses, and due the high number of hits to the destination hosts, I can assume that it is false-positive, and also that the server MY.NET.150.198 is used as a print server.

Alert #2 - Spp_http_decode: IIS Unicode attack detected

Top 10 Attackers

Count	IP Address
5214	MY.NET.153.123
3896	MY.NET.153.203
3730	MY.NET.153.122
3240	MY.NET.153.119
3211	MY.NET.153.113
3088	MY.NET.153.185
3067	MY.NET.153.110
2762	MY.NET.153.137
2697	MY.NET.153.187
2501	MY.NET.153.115

The Alert Example:

```
01/31-23:48:59.996998  [**] spp_http_decode: IIS Unicode attack detected [**]
```

```
MY.NET.153.169:1648 -> 216.33.240.250:80
```

All the alerts triggered by this alert were generated by hosts in your local network. These alerts are commonly generated by traffic associated with the NIMDA worm. All destination hosts are not located in your local network. Although this look like an NIMDA worm, just 276 *WEB-MISC Attempt to execute cmd* alerts were located in the 5 days analysis against 83769 alerts of IIS Unicode attack alert. The *WEB-MISC Attempt to execute cmd* alert is also triggered by the NIMDA worm. The unicodeexecute2.pl [11] perl script is a tool which also attempt directory transversal attempt. In case of Nimda or the use of this perl script, these machines should be investigated by inappropriate user activity. These also could be some false positives, like destination to known domain as home.netscape.com and mail.yahoo.com, but also some of the destination ip addresses point to Korea sites, which most of time indicate unsolicited/inappropriate traffic.

Another thing that could indicate false-positives is that the Snort preprocessor http_decode, the one that generated this alert, is known [26] to generate false positives with normal user traffic, especially when using the Netscape client.

Alert #3 Misc Large UDP Packet

Top 10 Attackers

Count	IP Address
10739	63.250.209.34
9859	63.250.211.165
9495	63.250.209.88
7246	63.250.210.50
4148	63.250.208.34
3109	203.231.232.15
1028	211.233.70.162
840	202.58.33.70
677	64.152.216.77
354	211.233.70.163

The Alert Example

```
01/31-20:14:10.284017  [**] MISC Large ICMP Packet [**] 158.42.4.2 ->
MY.NET.153.121
```

This alert is generated by the following Snort signature:

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

This alert is triggered when an UDP packet is greater than 4000 bytes is detected.

Dennis Ruck [12] got the same in his practical for GCIA on November 2001. He had two assumptions, one is that it could be a denial of service, but due the not so high rate, he took it as unlikely to be a DoS. Gary Portnoy [13], in his GCIA practical on December 2001 , also got the same pattern and also didn't believe to be a DoS.

After trying to identify the top 5 sources, I got the same conclusion as Gary Portner:

Host 63.250.209.34

```
Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
  2914 Taylor st
  Dallas, TX 75226
  US

  Netname: NETBLK2-YAHOOBS
  Netblock: 63.250.192.0 - 63.250.223.255
  Maintainer: YAHO

  Coordinator:
    Bonin, Troy  (TB501-ARIN)  netops@broadcast.com
    214.782.4278 ext. 2278

  Domain System inverse mapping provided by:

  NS.BROADCAST.COM          206.190.32.2
  NS2.BROADCAST.COM         206.190.32.3

  ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

  Record last updated on 29-Jun-2001.
  Database last updated on  20-Mar-2002 19:58:52 EDT.

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

Host: 63.250.211.165

```
Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
   2914 Taylor st
   Dallas, TX 75226
   US

   Netname: NETBLK2-YAHOOBS
   Netblock: 63.250.192.0 - 63.250.223.255
   Maintainer: YAHO

   Coordinator:
      Bonin, Troy  (TB501-ARIN)  netops@broadcast.com
      214.782.4278 ext. 2278
```

```
   Domain System inverse mapping provided by:

   NS.BROADCAST.COM 206.190.32.2
   NS2.BROADCAST.COM 206.190.32.3

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 29-Jun-2001.
   Database last updated on  20-Mar-2002 19:58:52 EDT.

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

Host 63.250.210.50

```
Non-authoritative answer:
50.210.250.63.in-addr.arpa    name = nsevent.broadcast.com.
50.210.250.63.in-addr.arpa    name = wmcontent.broadcast.com.
```

Host 63.250.209.88

```
Non-authoritative answer:
88.209.250.63.in-addr.arpa    name = dias.broadcast.com.
```

Host 63.250.208.34

```
Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
  2914 Taylor st
  Dallas, TX 75226
  US

  Netname: NETBLK2-YAHOOBS
  Netblock: 63.250.192.0 - 63.250.223.255
  Maintainer: YAHO

  Coordinator:
    Bonin, Troy  (TB501-ARIN)  netops@broadcast.com
    214.782.4278 ext. 2278

  Domain System inverse mapping provided by:

  NS.BROADCAST.COM          206.190.32.2
  NS2.BROADCAST.COM         206.190.32.3

  ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

  Record last updated on 29-Jun-2001.
  Database last updated on  20-Mar-2002 19:58:52 EDT.
```

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

It looks like a multimedia streaming traffic. Some reasons that can explain this conclusion are in the top 5 source ip addresses information, in which all indicates some broadcast activity, and also that most streaming application uses UDP instead of TCP.

Alert #4 SMB Name Wildcard

Top 10 attackers

Count	IP Address
8928	MY.NET.11.6
7829	MY.NET.11.7
2101	MY.NET.11.5
715	MY.NET.152.171
609	MY.NET.152.158
438	MY.NET.152.179
431	MY.NET.152.183
416	MY.NET.152.165
404	MY.NET.152.157
401	MY.NET.152.176

The Alert Example

01/31-23:59:13.065064 [**] SMB Name Wildcard [**] MY.NET.152.175:137 -> MY.NET.11.7:137

This alert is triggered when snort captures packets with local network ip address and port 137 and with wildcard '*' in the packet. In general, when a windows machine wants to discover another machine's name in the network, it tries to first query the DNS. If it receives a negative or no response, it will query NetBIOS. I agree with Robert Graham Site's FAQ: Firewall Forensics [14] explains:

"Exact signature: If the Windows box is trying to find the name for the IP address 192.0.2.21, it will do the following steps:

* Lookup the DNS "PTR" record for 21.2.0.192.in-addr.arpa; this request is sent to the local DNS server, which recursively forwards the query to the appropriate DNS server as required.

* If the DNS answer comes back, it won't query NetBIOS. If a negative response comes back, it will immediately query NetBIOS. If the DNS server times-out, it will wait 14-seconds, then query NetBIOS.

* When resolving with NetBIOS, it will send out a "NodeStatus" query that is sent to the 192.0.2.12:137 from x.x.x.x:137. (I.e. the query is sent to the IP address being resolved to its port 137, and is sent from the Windows machine port 137).

* The NetBIOS request is a "NodeStatus" query that looks up the name "*". It is 50 bytes worth of data (58 including the UDP header, 78 including the IP header, 92 including an Ethernet header). Three NetBIOS queries are sent with a 1.5 second timeout. "

Also, in Intrusion List from Incidents.org, Mr. James c. Slora Jr. sent an interesting email exactly about that:

Date 04/16/2002
Subject: RE: Outbond traffic to 4.4.4.4

```
Mystery solved - 4.4.4.4 is a web counter sent out in spam from
stockupticks.com. The spam had a link to a page on www.webcounter.com,
and this particular page redirects to http://4.4.4.4.

4.4.4.4 does not acknowledge the hit. Since it is a numeric address and
does not respond, IE next attempts to resolve the name to connect via
NetBIOS over TCP/IP - which generates a second outbound connection to UDP
port 137 on \\4.4.4.4. This traffic was blocked at the user's firewall.
I'm sure the server exists, though  - probably to keep a database of
addresses, browser versions, Windows versions, and NetBIOS names so
"unique hits" can be verified. I'm sure they keep this information highly
secured, and that they delete it quickly so there is little chance it
could ever be abused (sarcasm intended).

Maybe I should redirect my 404s to 4.4.4.4 so they can have lots of hits
whenever Nimda starts knocking on my door ;)

Source below:

GET /0/b1?a=17160 HTTP/1.1
Host: www.webcounter.com
Connection: close

Read 299 bytes from host www.webcounter.com, path /0/b1?a=17160 HTTP/1.1
302 Found
Date: Mon, 15 Apr 2002 21:18:36 GMT
Location: http://4.4.4.4/
Content-Length: 148
Connection: close
Content-Type: text/html

<head><title>Document moved</title></head>
<body><h1>Document moved</h1>
This document has moved <a href="http://4.4.4.4/">here</a>.<p>
</body>
-----Original Message-----
From: James C. Slora Jr. [mailto:Jim.Slora@phra.com]
Sent: Monday, April 15, 2002 2:15 PM
To: intrusions@incidents.org
Subject: Outbound traffic to 4.4.4.4
```

```
Has anyone seen outbound web traffic to (non-existent web server)
4.4.4.4?

Internet Explorer on a user's computer tries to visit http://4.4.4.4,
then attempts a UDP 137 connection on the same host. I have not port-
scanned that host (cuz that would be bad) to see if it exists, but it
definitely has no default web server running on TCP 80.

I suspect some sort of web bug or tracker bot link in spam, but have
found no direct evidence of it. The user is trustworthy - any ideas about
the source? IE does not appear to be trojaned.

- Jim
```

The Snort Signature:
misc-lib:alert udp any any -> $HOME_NET 137 (msg:"SMB Name Wildcard"; content:"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA|0000|";)

Besides that the alerts are apparently false positive, looking at this alert, I could notice strange traffic involving the machine MY.NET.5.96. It first received SMB traffic from the machine 12.91.133.194:

```
Name =   194.washington-15rh16rt.dc.dial-access.att.
Address:  12.91.133.194
```

After that , I could also notice that the machine MY.NET.5.96 also appears in the scan files receiving probes from other two cable modem/dial-up user to ports 21 (ftp) and 22(ssh).
The machine MY.NET.70.177 also should be checked against inappropriate user activity, because it was constantly scanning this machine.
Defensive recommendations are applicable. More details on the Defensive Recommendation Section.

Alert #5 SNMP Public Access

Top 10 attackers

Count	IP Address
12589	MY.NET.70.177
3673	MY.NET.150.198
2418	MY.NET.150.41
1870	MY.NET.150.245
1849	MY.NET.153.220
1302	MY.NET.88.240
285	MY.NET.84.155
129	MY.NET.150.49
92	MY.NET.186.10
33	MY.NET.183.11

The Alert Example

```
01/31-23:52:04.259312  [**] SNMP public access [**] MY.NET.153.220:1250 ->
MY.NET.152.109:161
```

This alert is generated by a signature that identifies SNMP requests with the ´public´ password. This is dangerous [15] because multiple implementations use this default password and some administrators never change it. The attack can then make queries and retrieve information and gather reconnaissance data about the device for further exploit. All destination IP addresses are in local network. Just one source IP addresses are outside of the local network, the IP 193.91.212.66, which belongs to KPNQwest. The destination IP address was MY.NET.150.133, which could also be investigated.
This alert per itself is very alarming, but recent vulnerabilities in multiple vendors´ SNMP protocols revealed in last February [26], show that an extra caution should be taken when we see alerts related SNMP.
If the use of SNMP is really necessary, the use of SNMP v3 should be considered. The v3 correct some of the previous vulnerabilities and also if it is not strictly necessary, you should disable the SNMP.

Alert #6 ICMP Echo Request L3retreiver Ping

Top 10 attackers

Count	IP Address
724	MY.NET.152.171
610	MY.NET.152.158
437	MY.NET.152.179
426	MY.NET.152.165
422	MY.NET.152.183
414	MY.NET.152.157
404	MY.NET.152.180
403	MY.NET.152.166
399	MY.NET.152.159
397	MY.NET.152.176

The Alert Example

```
01/31-23:58:29.055211  [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.163
-> MY.NET.11.7
```

The Snort signature that triggered this alert was:

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

All the alerts were generated by hosts in your local network. This is mostly associated with the L-3 security scanner [16], which is used for vulnerabilities analysis. Currently the L-3 Network Security site points to the Symantec site, which acquired it [17].
This indicates that users in your network may be using it to discover vulnerabilities in other hosts for further exploration.
Also, it was noticed that Windows2k client boxes [18] also match the pattern used in the Snort Signature. So it may be simple Windows echo requests.

Alert #7 Spp_http_decode: CGI Null Byte attack detected

Top 10 attackers

Count	IP Address
5335	MY.NET.153.157
3003	MY.NET.152.179
2320	MY.NET.153.159
176	MY.NET.152.165
97	MY.NET.226.22
77	MY.NET.105.113
76	MY.NET.153.153
43	MY.NET.153.112
40	MY.NET.153.141
34	MY.NET.153.121

The Alert Example

```
01/31-19:44:14.913743  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.153.157:1501 -> 209.10.239.135:80
```

This alert is triggered by the http preprocessor. If the http decoding routine finds a %00 in a http request, it will alert with this message. An attacker may append a %00 to confuse perl scripts about where the end of input is.
This signature also may alerts with false positives with sites that include urlencoded binary data [19].

I believe that this is the case here. Watching the destination address of the top 10, I got that the top three source addresses were accessing ifilm.com**,** which is a film content provider, very accessed by students.

Alert #8 High port 65535 udp – possible Red Worm Traffic

Top 10 attackers

Count	IP address
1396	MY.NET.6.49

1240	MY.NET.6.52
1125	MY.NET.6.48
1028	MY.NET.6.50
399	64.152.108.142
357	63.210.134.141
278	64.152.108.141
156	MY.NET.6.60
130	MY.NET.6.53
109	MY.NET.6.45

The Alert Example

```
01/31-23:32:45.297297  [**] High port 65535 udp - possible Red Worm - traffic [**]
MY.NET.6.49:65535 -> MY.NET.152.175:65280
```

Keven Murphy [20] in his GCIA practical of Dec 2001, detected the same alerts. He states that this traffic can be alerted by the following snort signatures:

alert UDP $EXTERNAL any -> $INTERNAL 65535 (msg: " High port 65535 udp - possible Red Worm – traffic ";classtype: system;)

alert UDP $EXTERNAL 65535 -> $INTERNAL any (msg: " High port 65535 udp - possible Red Worm – traffic ";classtype: system;)

I would also include the follow:

Alert UDP $INTERNAL 65535 -> $INTERNAL any (msg: "High port 65535 udp – possible Red Worm – traffic ";classtype: system;)

These alerts explains the Alerts being displayed triggered by both Local and External network machines with ports 65535.
This traffic can be associated with the worm which uses exploits of BIND to install itself and replace some binaries. I would suggest investigate the machines in the local network in the Top 10 list.
All the Local IP addresses are unlikely to be generating false positive. By checking the traffic associated with each IP on the top 10 list, I could also check that much of the traffic have same source and destination port 65535, and traffic from high port to high port.
Should also be noticed that the these local IP addresses didn't triggered any other alert from the top 10 alerts.
There are also 3 non-local network IP addresses in the Top 10 List. These addresses seem to be producing legal traffic

Host 63.210.134.141

```
Level 3 Communications, Inc. (NETBLK-LEVEL4-CIDR) LEVEL4-CIDR
                                                  63.208.0.0 - 63.215.255.255
Streaming Media Corporation (NETBLK-NETBLK-STRM6) NETBLK-STRM6
```

```
                                    63.210.134.0 - 63.210.134.255

To single out one record, look it up with "!xxx", where xxx is the
handle, shown in parenthesis following the name, which comes first.

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

Host 64.152.108.142 and 64.152.108.141

```
Level 3 Communications, Inc. (NETBLK-LC-ORG-ARIN) LC-ORG-ARIN
                                    64.152.0.0 - 64.159.255.255
Streaming Media Corporation (NETBLK-NETBLK-STRM9) NETBLK-STRM9
                                    64.152.108.0 - 64.152.108.255

To single out one record, look it up with "!xxx", where xxx is the
handle, shown in parenthesis following the name, which comes first.

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

They belong to SMC (Streaming Media Corporation), which provides video hosting among other services. This can explain the udp protocol, very used by streaming applications.

The machines MY.NET.152.158, MY.NET.153.209, MY.NET.153.162, MY.NET.153.148, MY.NET.153.141, MY.NET.153.210, MY.NET.152.170, MY.NET.152.166, should be also checked, because was notice additional traffic involving them and the top 10 attackers.

Alert #9 INFO MSN IM Chat data

Top 10 attackers

Count	IP Addresses
587	MY.NET.153.46
368	MY.NET.153.122
359	MY.NET.153.109
314	MY.NET.88.181
309	MY.NET.150.165
272	MY.NET.153.45
250	MY.NET.88.227
183	64.4.12.165
177	64.4.12.169

166 64.4.12.189

The Alert Example

01/31-23:56:30.677961 [**] INFO MSN IM Chat data [**] 64.4.12.173:1863 -> MY.NET.88.181:1675

This alert is generated by an Snort signature similar to this:

alert tcp $EXTERNAL_NET 1863 -> $HOME_NET any (msg:"INFO msn chat access";flags: A+; content:"|746578742F706C61696E|"; depth:100; classtype:not-suspicious; sid:540; rev:2;)

This alert always comes up when data with the proper content and the external host with the port 1863 is detected in the network traffic. In general it alerts about IM (instant message) traffic. IM applications became very popular in the internet over the past years. ICQ [21] and AOL IM are examples. This signature in particular detects traffic from the Microsoft Instant Message application.

The last three addresses are:

Non-authoritative answer: 165.12.4.64.in-addr.arpa name = msgr-sb16.msgr.hotmail.com.
Non-authoritative answer: 169.12.4.64.in-addr.arpa name = msgr-sb20.msgr.hotmail.com. Authoritative answers can be found from: 12.4.64.in-addr.arpa nameserver = ns1.jsnet.com. 12.4.64.in-addr.arpa nameserver = ns1.hotmail.com. 12.4.64.in-addr.arpa nameserver = ns2.jsnet.com. 12.4.64.in-addr.arpa nameserver = ns3.hotmail.com. ns1.jsnet.com internet address = 209.1.113.3 ns2.jsnet.com internet address = 209.1.113.4
Non-authoritative answer: 189.12.4.64.in-addr.arpa name = msgr-sb40.msgr.hotmail.com.

As we could see, they all belong to hotmail which is part of the Microsoft.
There is some vulnerability associated with these applications [22], but I couldn't notice any malicious behavior associated with the top IP addresses, so the use of these kind of application depends of the university policy and this signature could be used as a tool to check the use of these kind of application.

Alert #10 - Watchlist 000220 IL-ISDNNET-990517

Top 10 attackers

Count IP Address

1419	212.179.35.8
1119	212.179.35.118
351	212.179.27.176
79	212.179.27.6
56	212.179.125.79
35	212.179.35.121
34	212.179.75.132
32	212.179.71.214
25	212.179.126.3
24	212.179.19.2
22	212.179.125.254

The Alert Example

```
01/31-22:44:48.034098  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.118:1214 -> MY.NET.150.41:1288
```

This alert is correlated with Jasmir Beciragic [23] GCIA practical on Feb 2001, which describes as Gnutella Traffic. Gnutella is just one of the various Peer-to-Peer applications used to transfer files, which most of them are MP3 files, very popular for music storage. Gary Portnoy also describes this traffic as P2P file sharing program. Most of the ports are associated with the KaZaa P2P application, which uses port 1214.
These applications are known to have exploits [24] associated with them, so a good policy would be not allow users to make use of these applications.

Scan Analysis

The Scans file appears to be in the Snort Portscan.log files format. This log format was already explained in beginning of Assignment 2, but essentially it presents:

Feb 28 15:01:53 61.120.12.42:21		->	xxx.xxx.xxx.37:21 SYNFIN ******SF	
(1)	(2)	(3)	(4)	(5)

Field 1- Date and Time

Filed 2- Source IP Address : Source Port

Field 3- Direction of the packet

Field 4- Destination IP Address : Destination Port

Field 5- Flags Set in the Packet
After analyzed the 2.413.964 lines of the Scan files, I could verify that the Local network (MY.NET) is responsible of 88,81% of the total of source IP addresses scanners (2144041).

Also, it is possible to verify that the scans were done by 1104 different IP addresses

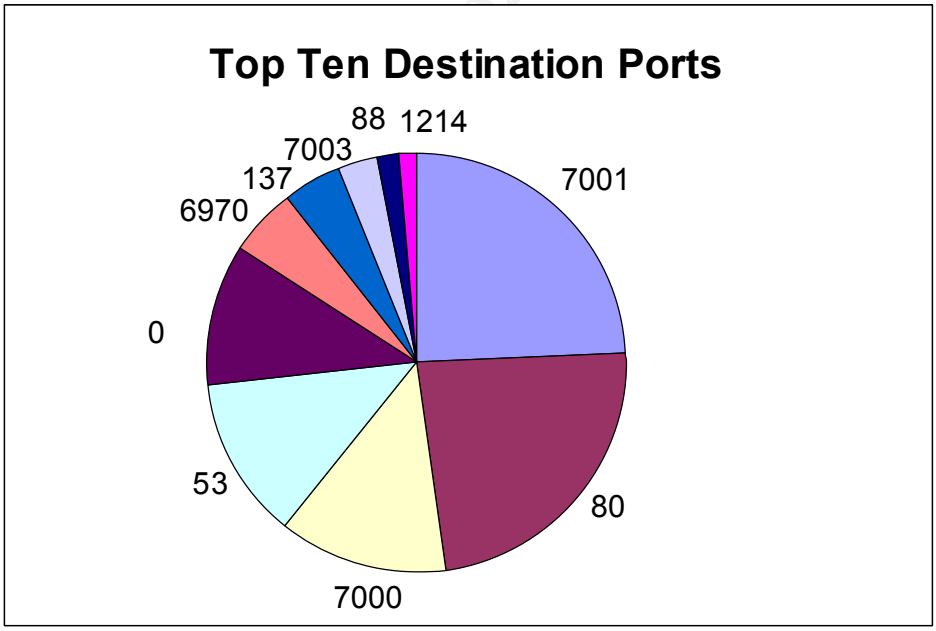
Bellow we can see the summary of these activities with the top 10 source IP addresses and the top 10 destination ports:

The Top Ten Source IP Addresses

Count	IP Addresses-Source
466947	MY.NET.60.43
123184	MY.NET.6.52
121472	MY.NET.6.49
115332	MY.NET.6.45
114115	MY.NET.6.48
88259	MY.NET.6.50
54756	MY.NET.6.53
50339	MY.NET.6.60
45757	64.152.108.141
40577	63.210.134.141

The Top Ten Destination Ports

Top	Count	Ports-Destination
1	301116	7001
2	295140	80
3	164164	7000
4	158149	53
5	132351	0
6	68385	6970
7	55349	137
8	37967	7003
9	18997	88
10	18884	1214



Official Services associated with the ports – IANA [7]

0	Unassigned
53	Domain Name Server
80	HTTP
88	Kerberos v5
137	NetBIOS name service
1214	Kazaa

6970	unassigned
7000	afs3 – File server
7001	afs3-callback
7003	volume location database

As we can see, much of the traffic is related with ports 7000, 7001 and 7003 (Top 1, 3 and 8). Although the port 7000 is associated with the Trojan SubSeven and SubSeven 2.1 Gold, the port 7001 are also associated with the [5]AFS distributed file system, and was previously worked in [1] D Mac Leod Practical for SANS. He found pretty much the same traffic as I did UDP traffic to/from 7001 to 7000 and 7003. David Singer also shows that ports 7000 and 7001 can be used by AOL chat programs. Also Port 6970 (Top 6) has a high number of scans associated with it. Investigating this port in the Scan Files, I saw that it is in most of time, associated with streaming traffic, as RealAudio.

The problem is that traffic as Real Audio goes from a range of 6970 to 7170, which may confuse with the ports used by AFS.

Port 0, which is number 5 at the Top Ten destination ports, also reveals some interesting aspects. According to [2] David Singer's Practical for SANS, this can be used to check the reaction from some OSs about traffic to port 0 for OS Fingerprinting, but, due the high number of scan associated with this port, it is unlikely to be part of OS fingerprinting traffic.
Matt Scarborough wrote an article [27] about gamming on the net and he states that "A mix of TCP connection attempts and UDP traffic from one or many hosts can make Online gaming difficult to differentiate from DoS attempts or pseudo-random port scans and probes" and that one of the ports used by MSN Zone offers online gamming is port 0. Matt also indicates a Microsoft document that says that "On some proxy servers, such as Microsoft Proxy Server, you will need to open UDP port 0 as an additional Subsequent UDP Inbound port.", so this could also be part of gamming traffic.

Port 137, which is number 7 at the Top Ten destination ports, is detected as a scan and also received a high number of alerts at the Alert Analysis (Number 4). I described as a NetBIOS discover that happens when a machine tries to discover another machine's name. A look at the Alert Analysis Section (Alert 4) can illustrate better.

Port 53, which is number 4 at the Top Ten destination ports, is detected as a scan, but since port 53 is associated with the DNS, it looks like queries to the DNS to resolve names/ip addresses. Another assumption that I can do is about your DNS servers. Looks like the machines MY.NET.1.3, MY.NET.1.4 and MY.NET.1.5 are part of your DNS servers, due the high number of hits.

Port 1214, which is number 10 at the Top Ten destination ports, is detected as a scan looking for KaZaa servers.[4]

Port 80, which is number 2 at the Top Ten destination ports, is detected as a scan looking for Web Servers, but a closer look at the destination addresses, indicates multiple attempts

to connect to the machines at same domain. Examples are multiple attempts to hotmail.com , Akamai Techonologies and the well-known search engine google. Also, there are some UDP packets with port 80. [3] NIPC has an advisory that indicates DDos associated with UDP packets with destination port 80.
The machine MY.NET.253.114 seems to be a very popular web server in your university. If it is not your official web server, a close look to this machine should be done, at least to check the content that it provides.

Port 88, which is number 9 at the Top Ten destination ports, is detected as a scan, but since port 88 is associated with Kerberos, we could assume that it is being used for Kerberos authentication: "When contacting a Kerberos server (KDC) for a KRB_KDC_REQ request using IP transport, the client shall send a UDP datagram containing only an encoding of the request to port 88 (decimal) at the KDC's IP"[6]

Bellow, we will find data about the top ten destination ports for each top ten talker.

Scanner #1 MY.NET.60.43

Top	Count	Destination Ports
1	70654	7001
2	33100	7000
3	12695	0
4	10995	123
5	1952	1030
6	1690	1051
7	1594	1052
8	1531	1053
9	1475	1054
10	1400	1055

Scanner #2 MY.NET.6.52

Top	Count	Destination Ports
1	17456	7001
2	11204	7000
3	8811	0
4	618	1
5	597	65535

6	531	8224
7	442	1552
8	353	25970
9	347	44299
10	302	2048

Scanner #3 MY.NET.6.49

Top	Count	Destination Ports
1	19346	7001
2	11642	7000
3	9179	0
4	555	65535
5	535	8224
6	520	1
7	398	1552
8	339	2048
9	337	25970
10	327	44299

Scanner #4 MY.NET.6.45

Top	Count	Destination Ports
1	88585	7001
2	49833	7000
3	18042	0
4	156	123
5	77	1
6	69	29487
7	53	25970
8	52	30067
9	51	12149
10	50	26227

Scanner #5 MY.NET.6.48

Top	Count	Destination Ports
1	17755	7001
2	11028	7000
3	7874	0
4	552	8224
5	530	1

6	499	65535
7	452	1552
8	329	25970
9	318	44299
10	279	15650

Scanner #6 MY.NET.6.50

Top	Count	Destination Ports
1	12850	7001
2	8225	7000
3	5913	0
4	482	65535
5	333	1
6	325	8224
7	227	44299
8	226	2048
9	221	26473
10	218	25970

Scanner #7 MY.NET.6.53

Top	Count	Destination Ports
1	30947	7001
2	17127	7000
3	14704	0
4	108	65535
5	82	8224
6	81	1
7	73	44299
8	64	29487
9	62	26982
10	59	26473

Scanner #8 MY.NET.6.60

Top	Count	Destination Ports
1	28679	7001
2	17361	7000
3	14185	0
4	123	65535
5	83	44299
6	77	1

7	68	8224
8	47	29487
9	47	12637
10	46	23854

Scanner #9 64.152.108.141

Top	Count	Destination Ports
1	8830	0
2	3895	7000
3	1715	2253
4	902	7001
5	271	256
6	211	8224
7	156	25970
8	146	29487
9	135	65535
10	125	15677

Scanner #10 63.210.134.141

Top	Count	Destination Ports
1	8602	0
2	2779	7000
3	1915	516
4	648	7001
5	535	4874
6	525	4521
7	452	33365
8	191	256
9	180	65535
10	142	8224

Out of Spec Analysis

OOS or Out of Specification packets are those whom don't fit the correct standard TCP/IP implementation.
In the five consecutive days of OOS files, I could only get 4 OOS packets, 2 on January 27th and 2 on January 28th .

January 27th

```
01/27-06:25:02.953126 65.129.33.127:18245 -> MY.NET.5.96:21536
TCP TTL:21 TOS:0x0 ID:18458  DF
2*SFRP*U Seq: 0x2F62696E   Ack: 0x2F636F6D   Win: 0x6E2F
2E 70 6C 20 48 54 54 50 2F 31                    .pl HTTP/1

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/27-06:25:06.806710 65.129.33.127:18245 -> MY.NET.5.96:21536
TCP TTL:21 TOS:0x0 ID:20506  DF
2*SFRP*U Seq: 0x2F62696E   Ack: 0x2F636F6D   Win: 0x6E2F
2E 70 6C 3F 62 62 61 74 74 3D                    .pl?bbatt=

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

These packets presents a strange behavior, that is the flags set 2*SFRP*U. Strange flags combination can be used either for OS fingerprint or Denial of Service. Also, there are errors reported about routers which under high load traffic mangle some packets. Another strange thing that we can notice is the info revealed in the payload. In the first packet we can see what apparently is an http request for some perl script (.pl) and the other on also appears to be another request for a perl file. This is strange also because the ports used in the traffic: Both are high ports, and high ports are not usually used by webservers., except by 8080, 8000 and others well-known ports used by proxy servers. This can enforce the hardware error idea.

01/27-06:25:02.953126	01/27-06:25:06.806710
.pl HTTP/1	.pl?bbatt=

January 28th

```
01/28-20:46:33.703718 64.166.209.137 -> MY.NET.88.162
TCP TTL:110 TOS:0x0 ID:33339  DF MF
Frag Offset: 0x0   Frag Size: 0x22
5A 1D 6B 5E 5B 1D 6C 99 22 37 5C 74 DD D3 2A 0C  Z.k^[.l."7\t..*.
C6 7A 15 8E E0 DC 01 2D 3E D6 87 7A D4 83 DF 32  .z.....->..z...2
3D B0                                            =.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/28-20:46:33.815778 64.166.209.137 -> MY.NET.88.162
TCP TTL:110 TOS:0x0 ID:33595  DF MF
Frag Offset: 0x0   Frag Size: 0x22
5B DC 9B FC 60 DE C0 BA E9 C4 23 F9 DA E5 95 D9  [...`.....#.....
E5 9A C7 D1 02 A6 EA 8D E4 6F 39 A3 53 B2 EB 18  .........o9.S...
75 57                                            uW

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

These packets are interesting because they set both DF and MF flags. These flags mean Don't Fragment (DF) and More Fragment (MF).

Richard Stevens [25] states that ´…the "more fragments" bit is turned on for each fragment comprising a datagram except the final fragment´ and bout the Don't fragment bit, he says

that ´…if this (bit) is turned on, IP will not fragment the datagram.´. In these OOS packets we can see both bits turned on, which indicate crafted packets.

Both IPs belong to Home users:

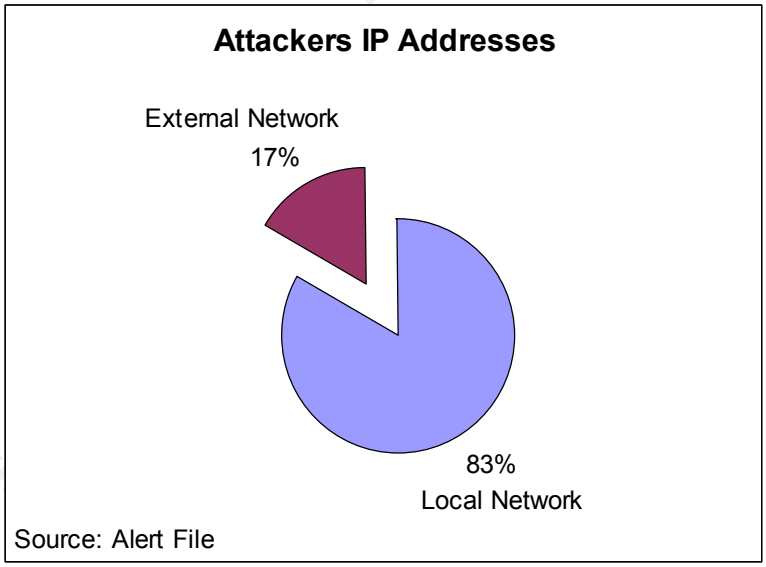
```
Non-authoritative answer:
137.209.166.64.in-addr.arpa
name = adsl-64-166-209-137.dsl.lsan03.pacbell.net.

Non-authoritative answer:
127.33.129.65.in-addr.arpa
name = 0-1pool33-127.nas34.philadelphia1.pa.us.da.qwest.net.
```

Defensive Recommendation

This section is designed to present the defensive recommendation for your network, based on the previous analysis. First I will give some high level recommendation and after that, a list with specific machines to be verified against hacker compromise or inappropriate user activity.

Overview



Your network seems to be a very large one and with a high traffic daily. Due the high number of machines, a split into several subnets with firewalls between them would increase the security.
Some other security improvements should be done in your network as for example a fine tune in your Firewall, Router and the Snort sensor, which could eliminate a lot of false positive identified in the analysis.

The SNMP is a very powerful management protocol, but the needs of it in the University should be considered with caution. Last February, CERT [27] released an advisory about multiple vulnerabilities associated with it. It is necessary to ensure that all devices that uses SNMP have the latest vendor patch applied. Also, the network administrator should be aware to check the community names, and not use the default public and private strings because this could compromise the device's use, giving the attacker a total control of the device. And, for a more complete solution, the use of ACLs at the router, blocking access to port 161 is also necessary.
I could notice that the NetBIOS protocol is allowed on both incoming and outgoing. A malicious user can take advantage of this to propagate virus and worms after identified public shares. This protocol should be blocked, either with ACLs at the router or on the firewall, on both directions.
The DNS servers received a high number of hits, and in spite of I couldn't find any evidence of malicious activity related them, I recommend to apply some security directives, following security guides as the SANS Reading Room Securing BIND [29].
The machine MY.NET.253.114 has a high number of accesses on port 80. If it is not your official web server, you should consider to at least taking a look and check the content that it is providing to the other students in your university and see if it is not against your security policy.
The security policy is another important subject if you are not yet using a policy to regulate your users' activity. There are a high number of alerts generated by applications like MSN Instant message and Kazaa. Although these as very common tools, and worldwide used, it should be a good idea to verify the amount of band used by them. And, at last but not least, you should consider include in your security policy, a Penetration test at least four times a year.

A specific look in these machines is also necessary:

MY.NET.5.96	Netbios traffic with external machines and different attempts from 12.91.133.194 and 65.129.33.127
MY.NET.70.177	Scanning to ports 135, 111 and 161 Against MY.NET.5.x
MY.NET.152.158	65535 to 65535 port traffic
MY.NET.153.209	65535 to 65535 port traffic
MY.NET.153.162	65535 to 65535 port traffic
MY.NET.153.148	65535 to 65535 port traffic
MY.NET.153.141	65535 to 65535 port traffic
MY.NET.153.210	65535 to 65535 port traffic
MY.NET.152.170	65535 to 65535 port traffic
MY.NET.152.166	65535 to 65535 port traffic
MY.NET.150.133	External SNMP access from 193.91.212.66
MY.NET.253.114	Possible Web server ?

Thank you and hope to work with you again.

References

1- D MacLeod Practical
http://www.sans.org/y2k/practical/D_MACLEOD_GCIA.doc

2- David Singer Practical
http://www.giac.org/practical/David_Singer_GCIA.doc

3- NIPC –Ddos 80
http://www.nipc.gov/warnings/advisories/2001/01-012.htm

4- Kazaa
http://users.pandora.be/lechat/Morpheus%20Exploit.htm

5- AFS
http://www.phrack.com/phrack/55/P55-13

6- Kerberos RFC
http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1510.html

7- IANA Official Port Numbers
http://www.iana.org/assignments/port-numbers

8- Dennis Ruck SANS Practical
http://www.giac.org/practical/Dennis_Ruck_GCIA.zip

9- Lenny Zeltser SANS Practical
http://www.sans.org/y2k/practical/Lenny_Zeltser.htm

10 - CVE – LPR e LPRng
http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=lprng

11- GeoCrawler – spp_http_decode
http://www.geocrawler.com/archives/3/4890/2001/8/0/6521002/

12- Dennis Ruck GCIA Practical
http://www.gcia.org/practical/Dennis_Ruck_GCIA.zip

13- Gary Portnoy GCIA Practical
http://www.gcia.org/practical/Gary_Portnoy_GCIA.zip

14- Robert Graham Site
http://www.shmoo.com/mail/ids/mar00/msg00065.shtml

15- CISCO Secure Consulting
http://www.cisco.com/warp/public/778/security/vuln_stats_02-03-00.html

16- L3 Retriever
http://www.westcoast.com/asiapacific/articles/standalone/l3/l3_retriever.htm

17- Symantec WebSite
http://enterprisesecurity.symantec.com/content.cfm?articleid=7&PID=10410963&EID=1

18- Snort-Users List
http://www.ultraviolet.org/mail-archives/snort-users.2001/0531.html

19- Neophasis:
http://archives.neohapsis.com/archives/snort/2000-11/0244.html
http://archives.neohapsis.com/archives/snort/2000-11/0248.html

20- Keven Murphy GCIA Practical
http://www.gcia.org/practical/Keven_Murphy_GCIA.zip

21- CERT advisory ICQ
http://www.cert.org/advisories/CA-2002-02.html

22- AOL Advisory IM
http://content.techweb.com/custom/security/1078.html

23- Jasmir Beciragic GCIA Practical
http://www.giac.org/practical/Jasmir_Beciragic_GCIA.doc

24- P2P exploits
http://www.opennet.ru/base/exploits/999103082_194.txt.html
http://www.wup.it/article.php/sid/1703

25- Richard Stevens – TCP/IP Illustrated Volume 1.

26- Snort FAQ – Item 4.17
http://www.snort.org/docs/faq.html#4.17

27- CERT Advisory CA-2002-03
http://www.cert.org/advisories/CA-2002-03.html

28. Matt Scarobough Gamming Article
http://www.incidents.org/detect/gaming.php

29. SANS Reading Room –Securing Bind
http://rr.sans.org/DNS/sec_BIND.php

Last Updated: February 17th, 2019

Upcoming Training

SANS Scottsdale 2019	Scottsdale, AZ	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS New York Metro Winter 2019	Jersey City, NJ	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS San Francisco Spring 2019	San Francisco, CA	Mar 11, 2019 - Mar 16, 2019	Live Event
SANS Madrid March 2019	Madrid, Spain	Mar 25, 2019 - Mar 30, 2019	Live Event
SANS 2019	Orlando, FL	Apr 01, 2019 - Apr 08, 2019	Live Event
Blue Team Summit & Training 2019	Louisville, KY	Apr 11, 2019 - Apr 18, 2019	Live Event
SANS Riyadh April 2019	Riyadh, Kingdom Of Saudi Arabia	Apr 13, 2019 - Apr 18, 2019	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201904,	Apr 24, 2019 - May 30, 2019	vLive
Community SANS New York SEC503	New York, NY	Apr 29, 2019 - May 04, 2019	Community SANS
SANS Security West 2019	San Diego, CA	May 09, 2019 - May 16, 2019	Live Event
SANS Northern VA Spring- Reston 2019	Reston, VA	May 19, 2019 - May 24, 2019	Live Event
SANS Amsterdam May 2019	Amsterdam, Netherlands	May 20, 2019 - May 25, 2019	Live Event
SANS San Antonio 2019	San Antonio, TX	May 28, 2019 - Jun 02, 2019	Live Event
San Antonio 2019 - SEC503: Intrusion Detection In-Depth	San Antonio, TX	May 28, 2019 - Jun 02, 2019	vLive
SANS London June 2019	London, United Kingdom	Jun 03, 2019 - Jun 08, 2019	Live Event
SANSFIRE 2019	Washington, DC	Jun 15, 2019 - Jun 22, 2019	Live Event
Security Operations Summit & Training 2019	New Orleans, LA	Jun 24, 2019 - Jul 01, 2019	Live Event
SANS Paris July 2019	Paris, France	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS Columbia 2019	Columbia, MD	Jul 15, 2019 - Jul 20, 2019	Live Event
Rocky Mountain 2019 - SEC503: Intrusion Detection In-Depth	Denver, CO	Jul 15, 2019 - Jul 20, 2019	vLive
SANS Rocky Mountain 2019	Denver, CO	Jul 15, 2019 - Jul 20, 2019	Live Event
SANS Boston Summer 2019	Boston, MA	Jul 29, 2019 - Aug 03, 2019	Live Event
SANS Chicago 2019	Chicago, IL	Aug 19, 2019 - Aug 24, 2019	Live Event
SANS Copenhagen August 2019	Copenhagen, Denmark	Aug 26, 2019 - Aug 31, 2019	Live Event
SANS Network Security 2019	Las Vegas, NV	Sep 09, 2019 - Sep 16, 2019	Live Event
SANS Oslo September 2019	Oslo, Norway	Sep 09, 2019 - Sep 14, 2019	Live Event
SANS Dallas Fall 2019	Dallas, TX	Sep 23, 2019 - Sep 28, 2019	Live Event
SANS London September 2019	London, United Kingdom	Sep 23, 2019 - Sep 28, 2019	Live Event
SANS Baltimore Fall 2019	Baltimore, MD	Oct 07, 2019 - Oct 12, 2019	Live Event
SANS San Diego 2019	San Diego, CA	Oct 07, 2019 - Oct 12, 2019	Live Event
SANS OnDemand	Online	Anytime	Self Paced