Global Information Assurance Certification Paper

GCIA Practical

By Joe Bowling

Assignment

Version 3.3

(submitted September 20, 2003)

SANS Inner Harbor 2003

Baltimore, MD April 7-12th

ABSTRACT: The first assignment is on the Future of IDS. This assignment covers how IDS have developed over time as well as future changes to the technology.
The second Assignment of this practical is analyzing three network detects. The three detects covered are “version bind request”, “port 0” “Gnutella Connect”.
The third Assignment is the analysis of five days worth of snort generated logs from a campus environment

TABLE OF CONTENTS

Assignment 1 FUTURE OF IDS

Introduction
Much has been asked about what is the future of Intrusion Detection Systems (IDS) as a technology. These questions have come from executives as well as from tech boards such as www.whitehats.com. The purpose of this paper is to explain how IDS are going to function in the future.

This assignment will consist of five areas.

Overview of TCP/IP communications
Why have an IDS? Isnt' a firewall enough?
IDS of past and present
Challenges and Limitations of IDS
Future of IDS

Before we jump into the Future of IDS we need to back up and start with an overview of TCP/IP to lay some foundational understanding. We then will briefly cover why even have an IDS, give some brief history of IDS, and current day functionality/challenges of IDS. The final portion will cover how problems of today's IDS will be met in the future. IDS.

Overview of TCP/IP communications
Before covering what I believe to be coming in the near future of IDS one will need to have some understanding of how the TCP/IP suit of protocols operate to have a appreciation/understand how the future IDS will be enhanced from what IDS are today. Lets have a quick review of how the TCP/IP suite of protocols function.

The Protocols we will cover are TCP, UDP, and ICMP.

TCP is a protocol that works at layer 4 in the OSI model and handles host-to-host communication. TCP is a reliable protocol due to its connection-oriented method of communication via sequence numbers.

There are multiple flags that are set to help the TCP protocol function. These flags are

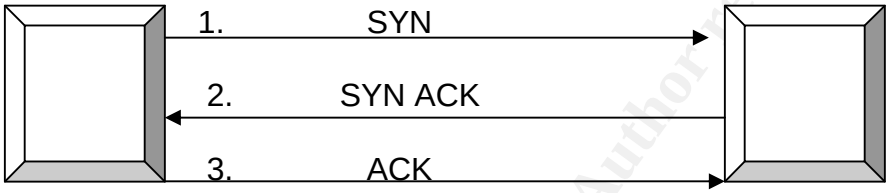
- Syn --------1st packet sent to Synchronize (start communication) with another host
- Ack---------used to acknowledge what data has been received by a host
- Push-------Tells a host to push data from the buffers up the TCP stack

- Urgent----- indicates which data is important and takes priority over other data
- Reset------used to immediately terminate a communication session
- Fin----------used to gracefully end communication session
- ECN--------used to notify host of congestion

To open a TCP connection between two hosts a “3 way handshake” must take place.

1. It starts with Host A sending a Syn packet to host B.
2. Host B responds sending its own Syn packet along with an Ack to acknowledge host A’s Syn
3. Host A responds to Host B’s Syn with a Ack

At this point we have a full duplex session set up. The hosts will exchange information until its time for each host to end the session.



The same process will be done to close a session except this time Fin packets will be used to end the session instead of SYN packets (which are used to start a session)

1. It starts with Host A sending a FIN packet to host B.
2. Host B responds sending an Ack acknowledge host A’s Fin (closing that direction of the communication session
3. When Host B is finished it will send its Fin packet to Host A
4. Host A will respond with an Ack to Host B’s Fin packet thus fully closing the session

A session can be immediately terminated by sending a reset packet. A reset packet does not require an ACK to close the session.

UDP/ICMP

These protocols generally work together. Neither UDP nor ICMP is connection oriented meaning they can’t verify if the data they send ever arrives at its destination. They are the “send and pray” protocols. UDP operates at Layer 4 and ICMP is considered to operate at layer 3 in the OSI model. The advantage of UDP has over TCP is that UDP is quicker.

ICMP is used to help alert of error conditions. Such alerts include

- Host unreachable messages
- Port unreachable messages
- Protocol unreachable messages
- Net unreachable messages
- Admin prohibited messages
- Echo and echo reply (ping)

This is just a simple basic summary of how these protocols work.

For a more information on the TCP/IP suite of protocols The TCP/IP Illustrated by Dr. Richard Stevens is considered by many to be the best reference available.

Why have an IDS? Isn't a firewall enough?

"No one can hack our network…we have a firewall"
-an uninformed system administrator

"Why have an Intrusion Detection System?…we have a firewall" That is a common first question response that management ask when proposed with the idea of implementing an IDS. Some system administrators may tell you "We are totally secured. We have a firewall and we don't allow anything into our network unless we send it out first". To briefly answer these valid questions lets start with a brief overview of what a firewall does and does not do, and what IDS does and does not do. Then we will wrap up with how the IDS and firewall are both needed to help secure a network.
Firewalls are made to stop unnecessary network traffic into or out of your network. Packet filtering firewalls typically will scan a packet for layer 3 and layer 4 protocol information. A typical packet filtering firewall blocks everything by default. It is only after you apply rules that "punch holes" (combination of IP address, protocol such as TCP/UDP, and port number) into the firewall is how you allow traffic into or out of your network. There aren't very much dynamic defensive abilities to most firewalls. The traffic approaching the firewall either matches up to applied rule and is allowed through or the traffic is stopped. Borrowing a similar example from Robert Graham (Security consultant) http://www.robertgraham.com/pubs/network-intrusion-detection.html Think of a firewall as a Fence around your network with strategically placed gates. If traffic is coming to your network, as long as it comes in at a position that has a gate, (a gate being a combination of IP address, Protocol, and port number) the traffic will pass through the firewall. Conversely if the traffic goes to a position on the fence/firewall where there is no gate the traffic will be blocked. Now what happens if malicious traffic goes through a gate? It has essentially bypassed your defense and worst yet it will not be listed in the firewall logs either so now we don't even know that the event has happened.

In contrast to firewalls, IDS will scan all packets at layer 3 and 4 as well as the application level protocols (most IDS can tell if the traffic is http, DNS, SMB etc) looking for back door Trojans, denial of service attacks (DOS), worms, buffer overflow attacks and detect scans against the network etc. IDS can be compared to a surveillance camera on the network in that it gives visibility to all traffic coming over the wires. IDS have a predefined rule list (also known as signatures or filters) Rules of an IDS have very specific parameters that are set up, when a packet matches those parameters the IDS will alarm. These parameters include IP address, port number, transport protocols (TCP/UDP), application protocols, and content (also known as payload). What happens if malicious traffic such as packets containing a NOOP sled passes through one of the holes that are "punched" into the firewall? The firewall will simply check to see if the IP address/port number/transport protocol matches up to any allowed rules to determine if the traffic will pass through. In this example the network traffic matches the "allowed in" rule on the firewall and is allowed to pass through. The IDS will scan the same traffic and detect a known malicious signature (the NOOP sled) in the content portion of the packet and will alert (notice we said alert and not necessarily stop the malicious traffic).

A simple example of this is you have a hole punched into your firewall to allow connections to an FTP server but someone attempts to download the passwd file from the ftp server. The firewall recognizes traffic to the FTP server as ok and doesn't block it. The IDS however will see this happening and will alarm. IDS can essentially can help you learn about vulnerabilities of the network as well as give us more visibility as to what is coming in and out of the network.

With only a firewall there are major problems when it comes to securing the network.

What if……

- There was an attack that the firewall did not block? How long before the attack is discovered if ever? How will one ever be able to trace it back to the attacker? Typically Firewalls are set up to log network traffic that was denied access so there would be no record of the event.
- There was an internal attack behind the firewall? The internal disgruntled/bored employee who thinks its fun to hack into corporate resources would go undetected by the firewall
- A worm infects a machine on the inside of the network and is being used to attack other hosts on our network or attack other external networks? For example a webserver infected with Nimda could be attacking business partners, internal networks, or random targets on the web, a firewall would not detect the initial traffic that infected the webserver (assuming port 80 is open on the firewall) nor would it recognize the attack traffic heading out of the network.
- The firewall did not detect a slow and stealth recon scan? How would you ever be able to do some preventive action? Ex. Renaming a server, changing its IP with that of a honey pot etc.

- Attacks that come in on ports that are intentional left open? Such as port 80 for web traffic. It would go undetected by the firewall (most companies have port 80 open so they can access the internet).

To conclude is IDS worth having? It all boils down to risk. How much risk is the company willing to take. We have briefly covered how attacks can happen even with a firewall. If a company has information and assets that are vital to the company's ability to function then the answer is yes. Having IDS provides much greater visibility to detect signs of an attack and compromised hosts. We still need a firewall to block traffic before it enters our network but, we also need an IDS to make sure the traffic that does gets past the firewall will be monitored.

A look at IDS: Past and Present

"Why should we upgrade we haven't been hacked yet?"
-Management

IDS are a fairly new technology that is still in its growing stages. IDS have already taken some major steps forward in a few short years in regards to capability and usability.

PAST: The first IDS were very limited in their functionality and flexibility. In the earlier days some IDS vendors did not allow users to write their own rules/signatures! Some vendors would not even allow their customers to view the rule list that came with the IDS. To say the least if a user can't write their own rules or see the rules that are used for alarms generated, the user is stuck trying to read a book in a dark room. Under such circumstances customizing a rule set based on one's network is totally out of the question. They would be overwhelmed by false positives! A false positive is when an alert is triggered for traffic that is allowed (think of a false alarm). IDS work by doing signature matching. Signature matching is when you identify a specific signature unique to an attack/virus/worm etc. When the packets are scanned by the IDS if t a matching signature is found in the IDS rule list, the IDS will alert. Older IDS were also stateless i.e. The ids would run each packet through the IDS rule engine on a per packet standalone basis thus there was no way for the IDS to maintain a TCP session or IP fragmentation session. The problem with stateless filtering was that it allowed hackers who fragmented their attacks to go undetected. Fragmented traffic has been used to create Denial Of Service (DOS) attacks on Routers, firewalls, and workstations. Fragmented traffic has been used to hide recon scans. Insertion and evasion attacks could easily bypass a stateless IDS.

A quick summary of insertion and evasion attacks is changing the traffic flow to purposely dodge the signature of the IDS sensor. Making the traffic so that it appears one way at the sensor (not matching the signature in the IDS rule list) but looks another way at the end destination. Borrowing an example taught by SANS would be to write a script that would telnet to a server and login with a logon of root but if we know that the IDS will alert on a "root" logon we will trigger an alert on the IDS so we set the program to send each character to the word root across individually but we also inject something else say the letter H so when the script runs the sensor will see the signature rooHt…which will not match the signature of "root". But to make sure we that "H" in "rooHt" doesn't get to the server (cause the logon is root not rooHt) there are a number of ways to accomplish this. One way is to have the packet carrying the "H" to have a TTL (time to live) that will expire immediately after it passes the sensor (which requires knowing the hop counts from the server it was logging into and where the sensor is) or an easier way would be to purposely corrupt the tcp checksum of the packet carrying the "H". In essence the sensor would read "rooHt" but once all the packets arrived at the server the server would discard the packet carrying the "H" due to the bad TCP checksum so the logon would only receive the letters "r" "o" "o" "t" equaling "root". Thus going undetected by the IDS. This paper will not go into the details of different attacks that employ insertion and evasion attacks but a good source of information is a paper written by Thomas H. Ptacek and Timothy N. Newshamcan found at http://www.snort.org/docs/idspaper/

IDS of past also did not have the ability to decode the protocol to see if the signature of a rule was needed to be scanned or not. An example is if you do not use the FTP command "put" but, you build a signature to alert when the "put" command is found. Without protocol decoding anytime the letters "put" showed up in anything from an email to a document, the alert would go off thus creating false positives. With protocol decoders the IDS knows to only apply the rule for "put" when the protocol is FTP

Signature matching is the type of NIDS that is mostly deployed but there is another "style" known as anomaly-based intrusion detection. Anomaly-based intrusion detection works on finding abnormal activity on the network but the problem starts with what is normal and what isn't normal. A good tool for this type of intrusion detection is "Shadow" which was created by Stephen Northcutt and his team in the Navy. It can be found at http://www.nswc.navy.mil/ISSEC/CID/
This paper will focus on signature matching "style" of intrusion detection

IDS sensors in the past could not sustain high speed traffic volume thus they would drop packets when the traffic load was to large…packets that could contain backdoor attacks or even the DOS attack that could be used to increase the bandwidth beyond the sensors ability to detect. Tools have been created to purposely set off IDS alerts to basically overwhelm the IDS and create a smokescreen of false positives. An example is Snot and Stick.

Large networks with multiple connections to the internet faced a major challenge of being able to correlate information that was gathered over multiple sensors to try to turn

the information into a useful data ex. It was difficult to detect slow stealth scans that would come through multiple sensors (especially if the analyst had to switch back and forth between screens looking at the alerts).

PRESENT: Currently today some of the major issues of the past have been resolved. Users are allowed to write there own rules list to customized to their own network even with commercial IDS such as NFR's (Network Flight Recorder) IDS. An example of customizing a rule list is if the user doesn't run IIS servers they would not write filters to detect the Nimda worm (the Nimda worm only effected IIS webservers) or in the case of commercial IDS where the rules are already written for the user or the analyst would simply not have the IIS rules loaded in the sensor to detect. Users are able to specify filters with great detail today. Filters can be set on IP, port, source/destination, protocol, TCP flag combinations, and content strings. For faster IDS performance some IDS will allow the user to specify where to begin looking for a string or how far to look for the string. For example, searching for the content beginning at the 5th byte in the application payload and searching the next 15 bytes would end on byte 20 in the payload). Other features are an option known as "tagging". Tagging allows IDS admin to specify that if a certain detect is found, that the IDS would record that connection either by specifying a time length…ex. 30 seconds, 2 mins or by the life of the connection. Tagging can also be set to grab a certain amount of packets after the detection is made.

Stateful inspection is also available on many IDS that now allow IDS admin to apply filters that "maintain state". This allows IDS to keep track of a TCP session or fragmented traffic in its memory buffers prior to applying a filter. For example fragmented traffic would be reassembled in the IDS memory buffers (sometimes referred to as preprocessors). Upon completion of the assembled fragmented traffic would it be run through the IDS filters. Most IDS can decode by protocol (recall example above about the "put" command for FTP) thus allowing for more refined filters (based on protocol) as well as many IDS today can decode Unicode/url encoding. Unicode is a universal number assigning to characters no matter the language, platform, and program. An example of Unicode is
http://www.sans.org is the same as http://www.%73%61%6e%73.org/
The IDS that decode Unicode would recognize that both URL's are the same…IDS that do not support Unicode would not alert on the signature that it has Unicode. Unicode can be used in URL's (possibly for directory transversal) as well as in commands such as cmd.exe is also %63%6d%64%2e%65%78%65 in Unicode.

Sensors today are able to handle much larger volumes of traffic. For example Snort has been able to keep with 150mbps speed and not drop packets (When considering the speed of traffic that an IDS can keep up it with must be taken with a grain of salt. Snap length (how much of each packet to capture off the wire) and number of rules has a major impact on the IDS performance). Load balancers are also available today from a company called Toplayer. A load balancer can take large streams of traffic and use a round robin approach to feeding the information down to multiple sensors (so you have multiple sensors processing the traffic of a single connection) ex. If the load balancer feeds into 8 separate sensors that are capable of 100mbps each you can essentially

process 800mbps without dropping any packets (which answers one of the Gartner reports concerns about IDS's being able to keep up with large volumes of traffic). Networks that use multiple sensors usually have all sensors report back to a management console and the management console helps to compile all of the detects into a usable readable format. For example the ability to compile all similar alerts together on 1 screen makes it much easier to detect stealth recon scans, correlate DDOS and correlate events from different parts of the network.

With all the progress that IDS have made over the last few years, it still has some major challenges.

Challenges and Limitations of IDS

"No need to investigate that alert…it happens all the time"
-an over worked intrusion detection analyst

False positives are one of the biggest problems when working with IDS a well-known fact as stated in the recent Gartner report on IDS. Please see URL http://www3.gartner.com/5_about/press_releases/pr11june2003c.jsp. False positives dull the ambition of an analyst by overwhelming them with alerts that turn out to not be alerts after all. False positives can come from not having specific enough filters or the filters may be set fairly well but that the attacks may go to subnets or devices where vulnerable systems do not exist. An example of this would be if you saw syn packets (syn packets will be explain better later on) being sent to multiple hosts on the 10.10.10.0 /24 subnet on port 135 (RPC service for Microsoft operating systems). Lets assume you have both Solaris and Windows operating systems on your subnet but, the only PCs on that particular subnet that received the syn packets had the Solaris operating system on the PCs thus the potential attack is not nearly as critical had there been Windows operating systems on those PCs. The analyst would have to see the alert and would have to investigate all the operating systems on all the hosts that received the syn packets destined for port 135 to verify if this traffic could potentially exploit vulnerability. To do so would require the analyst to have a solid knowledge of the network itself (which is very difficult to do in mid-large size networks) or the analyst will have to perform a lot of legwork. Sounds like fun doesn't it? It might be in the very beginning but even the most dedicated analyst will quickly begin to grow numb to these alerts that could be in the 1000's per day. The problem gets compounded if the network is constantly changing as many networks today are.

Today we are unable to integrate router logs, system logs and firewall logs, host based IDS alerts with alerts from an IDS. Analyst have to flip from device logs to device logs to correlate scans and alerts or even worse have to contact the firewall administrator and contact the system administrator to get the logs which would consume even more time.

A major challenge with IDS today is being able to set up and query the database information to find relevant data. Companies with large Networks accumulate large amounts of data. Filtering through large amounts of data quickly is a challenge. Most companies have their IDS record all of their data to large databases for keeping. Having that database cost money especially with the likes of an Oracle database and having to hire a DBA to maintain them. It would much easier if there was a device that came with as part of the IDS that would be easy to setup and use. The storage device needs to be secure and be able to respond to queries quickly.

The last main challenge is having a skilled IDS analyst. It seems that most security professionals who are in charge of the IDS are also responsible for setting up the IDS and maintaining it. This process usually consists of installing and tweaking the IDS software and rule list, which can become quite complex. Setting up a database for the IDS to alert to as well as create scripts and queries to be able to get intelligent data from the database is also not a very easy task (gets even more complex if the analyst has to write scripts that can try to import other alert formats into the same database such as syslog alerts, router logs etc. Monitoring and evaluating the alerts forces, the analyst to stay on top of all the newest attacks, worms, virus and network changes on the internal network (to keep rule list accurate). The IDS analyst needs to be skilled with many different OS's. The analyst is also usually on (or is) the incident response team. This can become overwhelming on the IDS analyst.

The Future of IDS

"In God we trust….all others we monitor"
-**United States Navy**

How are IDS going to change in the Future? I don't have a special crystal ball to read from and since Ms. Cleo is out of business we will have to rely on the directions we see IDS vendors moving toward such as ISS http://documents.iss.net/gartner_response.pdf. The strong source for the information that is covered in this upcoming section comes from Marty Rosech the CEO of Sourcefire (an IDS vendor) and the creator of the open source IDS Snort. Sourcefire held a conference on the future of IDS at which Mr. Rosech was the keynote speaker.

THE FUTURE

INTEROPERATIVE:

Vendors today have heard the cries of security professionals challenges with IDS and are taking action. Vendors (such as Sourcefire and ISS) today are working to make the future IDS, firewalls, and routers to be able to interoperate with each other. Having firewall and router functionality incorporated with the IDS will provide extreme flexibility to secure a network. Getting all devices to send logs to a database that can incorporate the formats into one that is viewable to the IDS management console will greatly aid in correlating alerts.

RESPONSIVE:
With IDS, router, and firewall capabilities being integrated (possibly all on the save device) users can have a firewall that will be able to dynamically adjust its rule set or a router that can change its routes based on traffic that the IDS sensor detects. The responsiveness will be specified in the IDS filters. This will provide true wire speed detection and prevention. A firewall will have the abilities to drop the connections that the IDS would detect, that normally would be allowed through the firewall. An example of this is networks that have port 80 open in the firewall for users to connect to the internet but the firewall is not smart enough to see if the traffic coming to port 80 is the Code Red worm or not. The IDS is smart enough. When the IDS see the traffic coming into port 80 with the Code Red signature the IDS will inform the firewall to block such traffic. The interoperability of IDS and routers can move in to the realms of Quality of Service (QOS) as well as changes in the routing table. With routers being able to interoperate with IDS we can have the router change where traffic is routed based on what the IDS detects. One may simply have the router route the malicious traffic to a black hole (a dead IP address) or make better use of malicious traffic by sending it to a honeypot or honeynet. Honeypots are either a PC or a software program that simulates what the Hacker would see given the commands that he enters. Honeypots can be used to help learn more about hacker techniques and motives. Honeynets is a small network of honeypots and is used for the same purpose as honeypots but more at the network level.

To touch upon the importance of achieving wire speeds that comes from having the IDS, firewall, and router all in one device is that it will allow us to stop the traffic before it gets to where its going thus resolving the problem that current IDS have that try to use responsiveness to alerts such as Snort's Flex response. What Snort's Flex response does is allow the user to specify in his filter a particular action that it would like Snort to perform to knock down a connection. There are two ways to knock down the connections one for each protocol TCP or UDP

For TCP connections to be torn down by sending a reset packet to either the source host, destination host, or both. For UDP connections to be torn down Snort can send a different ICMP messages such as ICMP NET unreachable, ICMP HOST unreachable, and ICMP PORT unreachable. Snort can send any combination or all three types of ICMP messages. Note that these ICMP messages will always be directed at the Source host of the filter that triggered the alert. Even though IDS such as snort have the capabilities to tear down a connection, it does not always work because the IDS are in a race to send the reset packets before the destination host responds back to the source.

A hacker could very easily set up his firewall to block the Reset and ICMP packets that the IDS would send in trying to knockdown the connection (if they something like flex response was being used). With wire speed the malicious traffic will not even get to the destination or it can be blocked or rerouted thus greatly reducing the ability of a hacker to cause damage.

DATABASE:

Databases will end up being part of the management console. The main reason is to simplify IDS by not having to have 3rd party hardware and software thus reducing the cost associated with the total IDS solution. We have already seen where Sourcefire (IDS vendor) has already done so. Their management console can perform a query through 1 million entries and pull 7 responses in 1 second. Which is the rapid responsiveness that intrusion analyst want and need. The database will also need to be able to take logs from other devices (even from independent vendors….easier said than done but we are talking about the future here. ☺) such as application, firewall, system, router, and IDS logs (both network based and host based) and be able to "normalize" the input date into a format that can be queried for useful output.

NETWORK DISCOVER VIA PASSIVE OS FINGERPRINTING:

What is passive operating system (OS) fingerprinting? Describing the details of passive OS fingerprinting is beyond the scope of this paper but we can give a quick summary of OS fingerprinting. Each OS has by default unique characteristics to how it communicates and functions. The characteristics that vary from OS to OS are Time to Live (TTL), the initial TCP window size, Max Segment Size (MSS), and even the Type of Service (TOS) options may be used to identify OS. A good research paper on passive fingerprinting based on a options field of a SYN packet was done by Toby Miller and can read at <http://www.incidents.org/papers/OSfingerprinting.php>
IDS can monitor the traffic that it sees and can "discover" what the operating system is and keep a dynamic "mapping" of the internal network. So what does this have to do with making life easier and better for the intrusion analyst? It is the key to greatly reducing one of the biggest problems in intrusion detection…false positives. How can passive OS fingerprinting help to reduce false positives? By the IDS knowing what each host is on the network it can decipher if the malicious traffic is destined for a potentially vulnerably OS. Lets go back to our earlier scenario.

You saw syn packets being sent to the 10.10.10.0 /24 subnet on port 135 (RPC service that runs on Microsoft operating systems). Lets assume you have both Solaris and Windows operating systems on your subnet but, the only PCs on that particular subnet that received the syn packets had the Solaris operating system on the PCs thus the attack is not nearly as critical had there been Windows operating systems on those PCs. The analyst would have to see the alert and would have to investigate all the operating systems on all the PCs that received the syn packets destined for port 135 to verify if this traffic could potentially exploit a vulnerability

How would this scenario report differently if passive OS fingerprinting were implemented? The IDS would have known that the OS at the destination of the inbound syn packets going to port 135 were Solaris operating systems that don't run the

vulnerable RPC service on port 135 that does run on Microsoft operating systems. Depending on the security policy the IDS could simple either simple ignore the suspicious traffic or still alert but only alert at a low priority for instance a priority 4 on a Scale of 1-5 (5 being low 1 being highest).

Now if the same scenario were to happen but the syn packets went to hosts that had the Microsoft OS that run the RPC service on port 135 (NT,win98/95/2000) then the IDS would alert with a priority level 2 given the serious nature of the traffic. Now when the intrusion analyst looks at the alerts he will immediately know which alerts are serious and which alerts are false positives. Less critical alerts would greatly reduce the amount of false positives that an analyst would have to review and investigate. As the analyst is reviewing the alert for this scenario since it is a priority 2 alert he knows that a serious event has occurred. After the analyst has seen the alert describing syn packets was sent to the RPC service (port 135) on Windows machines running the daemon he can simply check the router logs (from the same management console) and see that the router simply routed the traffic to the appropriate honeypot to which the analyst will review and share his findings with the Honeypot project team.

Brief word on Host based IDS. The focus of this paper was on future of Network based IDS (NIDS) but we should at least mention briefly how Host based IDS may change in the future. Stephen Northcutt (Director of Training for Sans Institute) mentioned in the book (Network Intrusion Detection co-authored with Judy Novak) that he believed that the Anti-virus companies should take a serious look at implementing host based IDS and gave his reasons for his belief. As of today Anti-virus companies haven't made any movement to implement it into their software. I personally believe that it will not happen anytime soon. The reason is that as Network based IDS develop as spoken about in the proceeding pages that those developments will increase the security enough that network based IDS will remain the focus of the industry. I just don't foresee the market pushing the anti-virus companies hard enough to take on the endeavors of host based IDS.

So are host based IDS going to be a thing of the past? Absolutely not. Host based IDS play a very important part in the overall picture of security. It is very hard for Network IDS to detect new attacks (if you don't have a signature for a unknown attack how can you set a rule to catch it?) but a host based IDS can have better chances to detect the unknown attack. Though usually it will be after the attack has happen. Host based IDS can detect if files have been changed or altered (a popular product is TripWire that accomplish this by creating MD5 checksums on files) or if people log in after odd hours, people try to access information that they don't have rights to, monitor system resources for anomalies…these are just a few examples of all the different ways to set up rules to watch for abnormal behavior on host based IDS. Entercept is sort of a "next generation" host based IDS in that it sits between the user and the OS and will know what commands can be executed even inside the program while it is running. Host based IDS have the ability to log alerts in different formats including syslog and snmp so in the future look to see host based IDS alerts going to the same database that the NIDS, firewalls, router alert logs go to.

CONCLUSION: In comparing how the IDS of today function compared to the future is drastic. Given the Scenarios described above, the IDS analyst would have to spend great amounts of time going through all alerts investigating and researching the host to see if it is vulnerable or not (whether the host is a Microsoft OS vs. Solaris in this instance) to discover if the alert is an event of interest (EOI) or was the alert a false positive. In future IDS the analyst will (most of the time) know immediately what alerts are false positives and which alerts are true EOI.

Large networks will have the most to gain with the IDS of the future. Security personal at the larger networks wont have to spend as much time investigating all the false positives and will be able to make changes in the network without having to rewrite large portions of the rule set because of the passive OS fingerprinting. For example if a company has a few users who roam around the campus and plug into the network at different places (especially in DHCP environments) the IDS will detect them (via passive OS fingerprinting) and will update its mapping of the network. With IDS, firewalls, and routers all now operating together as one unit (at wire speeds once all devices are on the same appliance) the possibility of malicious traffic getting to the destination will be greatly reduced thus providing greater security.

So what will be the drawbacks? Properly configuring and customizing an IDS has never been an easy task and with the added functionality that is coming the challenge will be even more intense (hopefully this will keep up the salaries for IDS engineers) The IDS of the future will require more hardware power than IDS today but as technology keeps rapidly advancing this should be able to be overcome without making the price too outrageous for a complete perimeter defense solution. There will need to be serious forethought into how to react to alerts…its not a good idea to block traffic from a IP address of a business partner or customer that a hacker could have used to spoof an attack. Discretion will be needed.

The benefits fully outweigh the drawbacks. Learning how to handle and implement all of the new functionality of the future IDS should be one heck of a learning experience for security professionals but, that is the fun part.

References:

Ptacek, Thomas Newsham, Timothy "Insertion, Evasion, and Denial of service: Eluding Network Intrusion Detection"
http://www.snort.org/docs/idspaper/

Miller, Toby "passive OS fingerprinting: details and techniques"
http://www.incidents.org/papers/OSfingerprinting.php

ISS "Gartner Claims IDS is Dead and Validates ISS' Strategy"
http://documents.iss.net/gartner_response.pdf

Northcutt, Stephen and Novak, JudyNetwork Intrusion Detection (3rd edition) 2002

Cothers, Tim Implementing Intrusion Detection Systems: A Hands-On Guide for Securing the Network Wiley 2003

Marty Rosech: CEO of Sourcefire, Seminar on the Future of IDS Washington DC 4/30/2003

George Bakos: Senior security expert at Dartmouth College, SANS instructor for intrusion detection. Baltimore Conference April 2003

Cohen, Fred "50 ways to defeat your IDS" http://www.darksouls.net/archives/ids/50_Ways_to_Defeate_Your_IDS.txt

ISS (Internet Security Systems) "The evolution of intrustion detection technology Aug 29, 2001 http://documents.iss.net/whitepapers/TheEvolutionofIntrusionDetectionTechnology.pdf

Graham, Robert "Network Intrusion Detection Systems" http://www.robertgraham.com/pubs/network-intrusion-detection.html

Sourcefire "Real-Time Network Awareness" June 2003 http://www.sourcefire.com/technology/whitepapers/sourcefire_RNA_0603.pdf

Assignment 2 Network Detects

Detect #1 Version Bind request

1. Source of Trace

This Trace came from <http://www.incidents.org/logs/Raw/2002.5.10>
The Raw file was further analyzed using Windump and Snort
Both IP and TCP/UDP checksums are invalid, due to obfuscation of the original IP addresses of the monitored network. The data made available for these detects contains only the raw data captured by Snort as the result of signatures being triggered. This limits us as far as analysis of possible response to stimuli is concerned, as the responses would only be logged if they too triggered Snort signatures. As well as there are only 2 Mac address involved 0:3:E3:D9:26:C0 and 0:0:C:4:B2:33. Most likely the edge router of the internal network and the internet both are Cisco devices (this was discovered using Ethereal). This detect is focused on a version bind request.

2. Detect generated by

This detect was discovered using the Snort IDS 1-9-1 using the default Snort rule list

Further analysis was done using Windump 3.0. Both are current as of Early April 2003

Snort detects

The command I used to detect the alerts was

C:\Snort\etc>snort -r 2002.5.10 -c snort.conf -l 510may22.log -Xde

A breakdown of the command used

r -read file (2002.5.10)
c -tells snort to act as an IDS with the configuration file to use (snort.conf)
l -logs the alerts to a file (510may22.log)
X -displays hex information
D -dumps application layer data
E -displays mac address information

Interpretation of the alert (field identifiers will be above each field)

[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]

TimeStamp Source Mac Destination Mac encapsulation protocol (IP)
06/10-19:09:50.084488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800

Datagram length
len:0x48

Source IP Destination IP protocol IP Time to live
210.195.43.28:2809 -> 46.5.76.175:53 UDP TTL:45

IP type of service IP ID value
TOS:0x0 ID:43103

Length of IP header Total length of datagram
IpLen:20 DgmLen:58

Length of data (protocol and payload)
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]
Snort Alerts

[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/10-19:09:50.084488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.28:2809 -> 46.5.76.175:53 UDP TTL:45 TOS:0x0 ID:43103 IpLen:20

DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]

[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/10-19:12:11.134488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.28:1252 -> 46.5.46.240:53 UDP TTL:45 TOS:0x0 ID:45451 IpLen:20
DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]

[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/10-19:49:20.654488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.28:4040 -> 46.5.150.2:53 UDP TTL:45 TOS:0x0 ID:17369 IpLen:20
DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]

WINDUMP

C:\Snort\etc\510may22.log>windump -r 2002.5.10 -nvX host 210.195.43.28

A breakdown of the command

-r read from a file (2002.5.10)
-n do not resolves IP address to domain names
-X show the Hex and the ASCII value
host selects the record if the source/destination host matches this IP (210.195.43.28)

Windump could be used to see if any of our hosts responded to the attackers request and verify the Snort alert in a real life scenario but, the file we have is only of snort alerts. This file does not contain all network data so we can't analyze it for a response. We use Windump to show the payload of the packet thus revealing the "version.bind" text that the Snort rule is looking for.

See rule below
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt"; flow:to_server,established; content:"|07|version"; nocase; offset:12; content:"|04|bind"; nocase; nocase; offset:12;
***Analisys of Hex output**
Green lettering is the IP header
Blue lettering is the Protocol header
Red lettering is the payload

19:09:50.084488 210.195.43.28.2809 > 46.5.76.175.53: [bad udp cksum faf7!] 4660 [b2&3=0x80] TXT CHAOS)? version.bind. [|domain] (ttl 45, id 43103, len 58, cksu74c5!

```
0x0000   4500 003a a85f 0000 2d11 74c5 d2c3 2b1c       E..:._..-.t...+.
0x0010   2e05 4caf 0af9 0035 0026 4f8c 1234 0080       ..L....5.&O..4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....
```

*The remainder of records

19:12:11.134488 210.195.43.28.1252 > 46.5.46.240.53: [bad udp cksum faf7!] 4660 [b2&3=0x80] TXT CHAOS)? version.bind. [|domain] (ttl 45, id 45451, len 58, bad cksum 8958!

```
0x0000   4500 003a b18b 0000 2d11 8958 d2c3 2b1c       E..:....-..X..+.
0x0010   2e05 2ef0 04e4 0035 0026 7360 1234 0080       .......5.&s`.4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....
```

19:49:20.654488 210.195.43.28.4040 > 46.5.150.2.53: [bad udp cksum f7fa!] 4660 [b2&3=0x80] TXT CHAOS)? version.bind. [|domain] (ttl 45, id 17369, len 58, bad cksum 8cfb!)

```
0x0000   4500 003a 43d9 0000 2d11 8cfb d2c3 2b1c       E..:C...-.....+.
0x0010   2e05 9602 0fc8 0035 0026 fe6c 1234 0080       .......5.&.l.4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....
```

3. Probability the source address was spoofed:
The source address does not appear to be spoofed. For the bind version information to be used the attacker would need to be able to get the response back thus he would not be able to do so using a spoof address (unless the attacker has the ability to sniff the traffic that would be sent to the spoofed address)

4.Description of the attack:
This is a recon attempt to find what version of bind is running on a DNS server. Doing a search for bind on cve.mitre.org shows there are 31 entries. Unfortunately we do not know what version of Bind runs on the internal network. Even if we did have the version number, we do not want outsiders knowing. There is a chance that they may know of a unknown exploit and can be looking for targets.

5. Attack mechanism:
Over the years bind has been plagued with many different exploits such as buffer overflow attacks that result in the attacker having Root level access. If a malicious attacker can find what version of bind is running on the DNS server they can match the version number up with known exploits for that particular version.

Why would an attacker want to compromise a DNS server?
DNS servers are used to map domain names to IP address. Thus if an attacker has seized control of a company's DNS server, he can use it for further resonance such as possibly looking for the IP address's of computers used by Senior officers of a company or changing the mapping of the domain names and IP address or poisoning the cache of the DNS servers. Bind versions less than 8.1.1 could have the DNS cache poisoned by providing extra information in a DNS reply packet that would be cached by the daemon. This could happen because a query and response use the same packet layout. This would allow an attacker to inject false information into the DNS cache for a network, allowing them to perform man-in-the-middle attacks or other mayhem.
This is what is believed to have happened when the controversial news company Al-Jazeera's website was compromised, by American hackers. The Websites Arabic homepage was directed to a porn site and the English version of the website was directed to a pro-USA website. Al-Jazeera had recently showed videotape of inapproiate video of US soldiers during the recent war in Iraq.
http://www.rense.com/general36/haxkced.htm

http://www.bizreport.com/article.php?art_id=4503&PHPSESSID=3030485d0c97adf0deba22b398e85aee

Another similar incident happened when Hillary Clinton was in process of running for Senate office of New York. When internet surfers typed in her homepage URL they were sent to a Website that supported (at the time) an opponent to Hillary for the Senate seat (Network Intrusion Detection 3rd edition pg 121 by Northcutt and Novak).

6 Correlations:
There are numerous cert advisories in regards to bind exploits

<http://www.securityfocus.com/guest/17905>

<http://www.cert.org/advisories/CA-2001-02.html>

<http://www.iss.net/security_center/advice/Intrusions/2000415/default.htm>

<http://www.cert.org/advisories/CA-1998-05.html>

http://icat.nist.gov/icat.cfm?cvename=cve-1999-0009

A popular cert advisory that Sans references when discussing cache poisoning is
http://www.cert.org/advisories/CA-1997-22.html

Ricky Smith covered this same alert in his GCIA practical
http://www.giac.org/practical/GCIA/Ricky_Smith_GCIA.pdf

A search on Dshield.org showed that the offending IP had not been reported for any type of abuse..

7 Evidence of active targeting:
I believe this was NOT actively targeted. I do believe the network was targeted by way of a very slow and stealth like scan. Slow as that there were only 3 packets from the offending IP address and stealthy in that each packet went to a separate destination IP address that was not in any chronological order. So why is this a slow and stealthy scan vs. being specifically targeted? If the attacker knew what the DNS's IP address was why would he try different IP's, which would draw more attention to himself. Unless these IP address used to belong to DNS servers but were recently changed due to the attacker trying to find the bind version, I believe this is a slow and stealthy scan vs. deliberately targeted.

8 Severity:
Critically - 5
DNS servers can be used to help map networks, corrupt the mapping of domain names to IP address, and exploit trust relationship thus exposing most of a site's systems.

Lethality - 4
Even though I don't believe that the bind request went to DNS servers (assuming that the IP addresses targeted weren't recently change away from DNS servers) it is still a big concern given the fact that this attacker seems to have no chronological order in which he is mapping the network and that he is slowly searching for the DNS server.

System Countermeasures -3
Obviously I cannot say with full accuracy what preventivness has been done on the systems attacked. Let's hope system administrators have been diligent in keeping up with current software and patches.

Network Countermeasures - 4
Since most networks today have a firewall, we assume this location has a firewall as well (sites that know the value of having an IDS would know the value of having a firewall as well).
We assume the Firewall is properly administrated and kept updated.

Severity = 2
(criticality 5 + lethality 4) -(system countermeasures 3 + network countermeasures 4) = 2

9 Defense Recommendations:
A Keep snort updated
B Keep firewall updated
C Block the offending IP address at firewall
D Send a letter to the ISP of the offender if the mapping continues (Though the ISP probably could care less)
E. Make sure the DNS server does not give out the Bind version information. This can

be accomplished by not putting the bind version or system information in the HINFO records or by putting an incorrect version number in HINFO records. Also in Bind versions 8.2 and later the configuration file has the ability to respond to with a message instead of the version number.
F If possible implement Split-Split DNS. Split-Spilt DNS is having one DNS server (preferably in the DMZ) to handle public request (and does not do recursive queries) and have a separate DNS server for internal users (allow recursive querying). This arrangement is similar to Split-DNS.
http://www.isaserver.org/tutorials/You_Need_to_Create_a_Split_DNS.html
G. Make sure the DNS server has the most current patches.
H Run in chroot mode. Do not run the DNS in root mode. This will limit the access of the account if it were to be compromised

10 Multiple choice test question:
Look at the UDP packet below

[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/10-19:49:20.654488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.28:4040 -> 46.5.150.2:53 UDP TTL:45 TOS:0x0 ID:17369 IpLen:20 DgmLen:58
Len: 38
[Xref => arachnids 278][Xref => nessus 10028]

Assume there is no filtering firewall
If the targeted system was up but not running anything on port 53 what should be the response?

A. TCP reset packet sent
B. ICMP host unreachable
C. ICMP port unreachable
D. UDP port unreachable

The answer is C

A. This is not a TCP packet
B. ICMP host would be sent if the system were not alive
D. There is no such thing as a UDP port unreachable response

NETWORK DETECT 2: Port 0

***This detect was submitted to the incidents mailing list on Thursday, July 17, 2003 10:50 PM. Responses to three questions are included**

throughout the analysis of this detect

1.Source of Trace:
<http://www.incidents.org/logs/Raw/2002.5.18>
The Raw file was analyzed with Snort
Both IP and TCP/UDP checksums are invalid, due to obfuscation of the original IP addresses of the monitored network. The data made available for these detects contained only the raw data captured by Snort as the result of signatures being triggered. This limited us as far as analysis of possible response to stimuli is concerned, as the responses would only be logged if they too triggered Snort signatures. I believe that the IDS is in front of a firewall because we see retransmitting packets (will be covered in more detail below).

2. Detect generated by
This detect was discovered using the Snort IDS 1-9-1 using the default Snort Rule list current as of early April 2003

SNORT DETECTS

Command used to discover alerts

C:\Snort\etc\may29_518.log>snort -r 2002.5.18 -c snort.conf -l may29_518.log -Xd

* for reference of Command variables please see network detect # 1. (page 14)

The rule was tripped because the Home Net port was zero. See rule with highlight below.

alert tcp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD-TRAFFIC tcp port 0)

notice destination port (bold) in alerts below

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:30:28.214488 211.47.255.23:52216 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5D6E5492 Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:30:31.204488 211.47.255.23:52216 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5D6E5492 Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:30:37.204488 211.47.255.23:52216 -> 46.5.179.136**:0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5D6E5492 Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:30:49.204488 211.47.255.23:52216 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5D6E5492 Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:00.214488 211.47.255.23:52608 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5F5B09FB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:03.204488 211.47.255.23:52608 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5F5B09FB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:09.204488 211.47.255.23:52608 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5F5B09FB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:21.204488 211.47.255.23:52608 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x5F5B09FB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:32.204488 211.47.255.23:52995 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF

******S* Seq: 0x60C074DB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:35.204488 211.47.255.23:52995 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x60C074DB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:41.204488 211.47.255.23:52995 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x60C074DB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:31:53.204488 211.47.255.23:52995 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x60C074DB Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:32:04.204488 211.47.255.23:53314 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6332253F Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:32:07.204488 211.47.255.23:53314 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6332253F Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:32:13.204488 211.47.255.23:53314 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6332253F Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]

[Classification: Misc activity] [Priority: 3]
06/17/02-20:32:25.214488 211.47.255.23:53314 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6332253F Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

3 Probability the source address was spoofed:
I don't believe the address was spoofed (using a spoof definition of: hiding behind a fake IP address because there is no need to receive any information back from the destination host). The sender is looking for a response so he can't spoof the IP address.

4 Description of the attack:
I believe this packet was indeed crafted. Port zero is an invalid port. The only way to have a packet sent to port zero is to craft the packet. You won't see normal traffic on port 0. This packet is most likely looking to recover some recon information
This is a Syn packet that is being sent to a destination host's port 0 which is invalid.
Please visit the URL below

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

Port 0 is officially a reserved port in TCP/IP networking, meaning that it should not be used for any TCP or UDP network communications.

However, port 0 sometimes takes on a special meaning in network programming, particularly Unix socket programming. In this environment, port 0 is a programming technique for specifying system-allocated (dynamic) ports.

Instead of "hard-coding" a particular port number, or writing code that searches for an open port, the programmer simply specifies port 0 as a connection parameter. That triggers the operating system to automatically search for and return the next available port in the dynamic port number range.

This programming technique does not work the same way in Microsoft Windows as it does in Unix.

Another reason I think the packets have been crafted because normally an OS will reassign port 0 to a random ephemeral port number.
Please visit URL below for reference

http://www.securiteam.com/securityreviews/5XP0Q2AAKS.html

Port 0's Normal usage
As many of you programmers will know, when you specify the source port of 0 when you connect to a host, the OS automatically reassigns the port number to high numbered ephemeral port. The same happens if you try to bind a listening socket to port 0.

The code below forces the OS to change the listening source port (my_addr.sin_port = 0) to another random ephemeral port.

```
//probably ripped from beej's guide to network programming
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <errno.h>
#define BACKLOG 1 // how many pending connections queue will hold

void main()
{
	int sockfd, new_fd; // listen on sock_fd, new connection on new_fd
	struct sockaddr_in my_addr; // my address information
	struct sockaddr_in their_addr; // connector's address information
	int sin_size;

	sockfd = socket(AF_INET, SOCK_STREAM, 0); //opps no checking

	my_addr.sin_family = AF_INET; // host byte order
	my_addr.sin_port = 0; // port 0 is reassigned
	my_addr.sin_addr.s_addr = INADDR_ANY; // auto-fill with my IP
	memset(&(my_addr.sin_zero), '\0', 8); // zero the rest of the struct

	if((bind(sockfd, (struct sockaddr *)&my_addr, sizeof(struct sockaddr))) !=0){
		printf("opps: bind error as %s\n",strerror(errno));
		exit(1);
	}

	//no checking oops
	listen(sockfd, BACKLOG);

	sin_size = sizeof(struct sockaddr_in);
	new_fd = accept(sockfd, (struct sockaddr *)&their_addr, &sin_size);
	printf("woop woop got a connection\n");
}
```

In response to my post on the incidents list I was asked by "rocker atschool" starplanet1000@yahoo.com.hk what was the attackers OS

We should never see a valid IP ID number of 0 but, this is what we see. The IP ID number is set to 0.

[**] [1:524:5] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/17/02-20:32:25.214488 211.47.255.23:53314 -> 46.5.179.136:**0**
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6332253F Ack: 0x0 Win: 0x16D0 TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

The only exception is some versions of Linux will send small packets with the DF flag set (which is set in the alerts detected). I believe the offenders OS is Linux
Please see URL's below for reference on Linux using IP ID of 0 with the DF flag set.
http://marc.theaimsgroup.com/?1=bugtraq&m=98992536801625&w=2

http://marc.theaimsgroup.com/?1=bugtraq&m=101709117512191&w=2

When datagrams are created they need a unique ID 0-65,535 (the IP ID field is a 16 bit field that is why the max is 65,535) the sending TCP/IP stack generally increment IP ID values by 1 or 256. The only case when a IP ID number would be 0 is if the IP ID has wrapped around (IP ID hit 65,535 then start over at 0) but this is not the case because all the packets came within 2 minutes which is not enough time to fully wrap around.

Please see the timestamps on the Windump output below that show the time frame as well as doubling back off timer for retransmitting the packet

20:30:28.214488 IP 211.47.255.23.52216 > 46.5.179.136.0: S
20:30:31.204488 IP 211.47.255.23.52216 > 46.5.179.136.0: S
20:30:37.204488 IP 211.47.255.23.52216 > 46.5.179.136.0: S
20:30:49.204488 IP 211.47.255.23.52216 > 46.5.179.136.0: S

20:31:00.214488 IP 211.47.255.23.52608 > 46.5.179.136.0: S
20:31:03.204488 IP 211.47.255.23.52608 > 46.5.179.136.0: S
20:31:09.204488 IP 211.47.255.23.52608 > 46.5.179.136.0: S
20:31:21.204488 IP 211.47.255.23.52608 > 46.5.179.136.0: S

20:31:32.204488 IP 211.47.255.23.52995 > 46.5.179.136.0: S
20:31:35.204488 IP 211.47.255.23.52995 > 46.5.179.136.0: S
20:31:41.204488 IP 211.47.255.23.52995 > 46.5.179.136.0: S
20:31:53.204488 IP 211.47.255.23.52995 > 46.5.179.136.0: S

20:32:04.204488 IP 211.47.255.23.53314 > 46.5.179.136.0: S
20:32:07.204488 IP 211.47.255.23.53314 > 46.5.179.136.0: S
20:32:13.204488 IP 211.47.255.23.53314 > 46.5.179.136.0: S
20:32:25.214488 IP 211.47.255.23.53314 > 46.5.179.136.0: S

We see that there are a total of 16 packets sent that are grouped in groups of 4. The initial packet then we see another packet with the same IP ID, port number, and the same TCP options set 3 seconds after the initial packet. Then 6 seconds later we see another identical packet followed 12 seconds later by another identical packet. This is a very common retransmission attempt (doubling back off times) that many TCP/IP stacks use. We see the same scenario played out in all 4 groups of packets. These packets are the stimulus. The sender is hoping to get some type of response from the host. Since we see the retransmission packets it is safe to assume the packets never arrived to the destination (most likely blocked at the firewall. This answers another question from Rocker at school which was how I could conclude if the packets were dropped at the firewall.

Port 0 has been known in the past to be associated with OS fingerprinting.

5 Attack Mechanism:

I am not sure what the sender of the crafted packet is trying to accomplish exactly. Maybe a recon effort. By sending packets to port 0 maybe the sender was trying to elicit some kind of a response that could be used to fingerprint the operating system.
I don't believe the offender is attempting to map the network since the sender only sends traffic to the same host every time (hard to map a network that way). Possibly this destination has been specifically targeted and the attacker attempted to connect a few times before giving up. This is why we see four separate sessions with the retransmitting of SYN packets with the double back timer.

Observations:
After reading other students previous analysis on a similar detect
http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00405.html
http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00087.html

It was mentioned that possibly the port scanning utility Hping2 could have been used to generate these packets. The main reason for the suspicion was because Hping2 will default to port 0 unless specifically stated to a different port number. I posted this to the incidents list to which I was asked if I could regenerate the packets

From: "Holger van Lengerich (Telefónica Deutschland)" <hvl@telefonica.de>

> Hi,Can you reproduce the TCP back off timing pattern with any version of hping

> easily? Given the answer to the former question: Is it still likely that hping

> crafted the packages you analyzed?

> Good luck,

> Holger

At this point I really hadn't had a lot of experience with any particular version of *nix (always had been a Windows person) but hping could not run on Win32 platform. I bought a used p2 266 PC just for putting Linux on. After getting hping from www.hping.org and going through the documentation and playing with the utility I could not figure out a way to recreate the packets (mainly the double back off timing as common in retransmission packets). At this point I consulted with a couple high-end security professionals. None had much experience using hping and could not provide any assistance. At this point I emailed the author of Hping2 Salvatore Sanfilippo explaining what I was trying to recreate as well as how I had made my attempts already to recreate the pattern and was unsuccessful. I asked if it was possible for hping to generate the packets in the described fashion and he responded saying NO. Hping2 could not send packets with the double back off timing in its native form unless you made a script to send a syn packet then sleep for 3 seconds, send another syn packet, sleep 6 seconds but, to do such would be purposeless and any kind of response such as a Reset-Ack would throw a wrench in things (See actual email below).

Hello,

not in the native way, but you can just wrap it in a shell script:

hping -S <dest> <your options> -c 1
sleep 3
hping -S <dest> <your options> -c 1
sleep 6
...

and so on. A problem with this solution is that if the SYN/ACK
reply doesn't come in short time, hping doesn't just exist
but wait, so there is some kind of time shifting.

Regards,
Salvatore

--
Salvatore Sanfilippo <antirez at invece dot org> s/at/@/ s/dot/./
finger antirez@tella.alicom.org for PGP key
"Universities and research labs force hackers to be
scientists, and companies force them to be engineers." (Paul Graham)

After all of this I am confident that these are not packets from Hping. So what caused these packets? Let continue to look at some other possibilities.

Could this be a response to someone else packets who spoofed using our address? When we look at the packets we do not see the Ack flag set. We only see the initial syn packet being sent. Had this been a response to someone spoofing our IP address we would see the packet with the syn and ack flags set (the 2nd step in the 3way handshake). Again this is a stimulus not a response.

Can the DF flag tell us anything? With the DF flag set, the attacker could hope to get back an ICMP need to fragment packet in hopes of discovering MTU and mapping purposes. However the datagram was only 52 bytes and even the smallest of networks are large enough to move that size without needing to fragment the traffic. Again some versions of Linux do set the DF flag on small packets which most likely is the case

Jason Thompson had put a lot of leg work into a similar detect that he posted to the incidents list on 7/17/2003 (see URL http://cert.uni-stuttgart.de/archive/intrusions/2003/07/msg00183.html).
Jason came across some information about Back Orifice looking for port 0. He experimented with the Trojan and was able to generate traffic that was very similar

except for the IP ID of 0.
I don't think we will ever figure out exactly the what or why. The best thing to do is to block this offenders IP range.

6 Correlations:
A search on Cert.com for port 0 did not produce any results

We checked out the offending IP on Dshield.org

IP Info

Check another IP Address: 211.47.255.23 Submit **IP Address:** 211.47.255.23 **HostName:** 211.47.255.23 **DShield Profile:** Country: KR Contact E-mail: ip@saeroun.co.kr Total Records against IP: not processed Number of targets: select update below Date Range: to Summary was recently updated. ***Top 10 Ports hit by this source:*** **Port Attacks Start End Last Fightback Sent:** sent to ip@saeroun.co.kr on 2003-04-25 13:37:19

*** Notice a Fightback was sent in April of 2003**

And on APNIC

inetnum: 211.42.0.0 - 211.51.255.255
netname: KRNIC-KR
descr: KRNIC
descr: Korea Network Information Center
country: KR
admin-c: HM127-AP <http://www.apnic.org/apnic-bin/whois.pl?searchtext=HM127-AP&form_type=advanced>
tech-c: HM127-AP <http://www.apnic.org/apnic-bin/whois.pl?searchtext=HM127-AP&form_type=advanced>
remarks: **
remarks: KRNIC is the National Internet Registry
remarks: in Korea under APNIC. If you would like to
remarks: find assignment information in detail
remarks: please refer to the KRNIC Whois DB
remarks: http://whois.nic.or.kr/english/index.html
remarks: **
mnt-by: APNIC-HM <http://www.apnic.org/apnic-bin/whois.pl?searchtext=APNIC-HM&form_type=advanced>
mnt-lower: MNT-KRNIC-AP <http://www.apnic.org/apnic-bin/whois.pl?searchtext=MNT-KRNIC-AP&form_type=advanced>
changed: hostmaster@apnic.net 19991118
changed: hostmaster@apnic.net 20010606
status: ALLOCATED PORTABLE
source: APNIC
person: Host Master
address: 11F, KTF B/D, 1321-11, Seocho2-Dong, Seocho-Gu,
address: Seoul, Korea, 137-857

country: KR
phone: +82-2-2186-4500
fax-no: +82-2-2186-4496
e-mail: hostmaster@nic.or.kr
nic-hdl: HM127-AP
mnt-by: MNT-KRNIC-AP <http://www.apnic.org/apnic-bin/whois.pl?searchtext=MNT-KRNIC-AP&form_type=advanced>
changed: hostmaster@nic.or.kr 20020507
source: APNIC

Also the GCIA practical by Susan Kovacevich (detect # 3) was used for correlation. Susan's alert came from the same subnet (211.47.255.22…who I believe is the same person) and the offender had used the exact same packets towards a different host. In her practical she mentioned that in October of 2002 that an email to incidents.org showed similar alerts from offending IP 211.47.255.21. Susan had discovered that in May of 2002 a fightback email had been sent.
In another GCIA practical by Brian Cahoon (detect #2) and Ewen Fung (detect #3) saw the same alert but from 211.47.255.24. The alerts were seen as early at 6/14/02 and Susan saw the alerts at late as 7/7/02

http://www.giac.org/practical/GCIA/Brian_Cahoon_GCIA.pdf
http://www.giac.org/practical/GCIA/Susan_Kovacevich_GCIA.pdf
http://www.giac.org/practical/GCIA/Ewen_Fung_GCIA.pdf

As mentioned above I also reviewed two other analysis that I found on the incidents list via google
http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00405.html
http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00087.html

7. Evidence of active targeting:
I think this was actively targeted. This offender is using packets that are crafted possibly for OS fingerprinting or Trojan activity. There was one specific IP that he attempted 4 separate times.

8 Severity:
Critically-2
We assume that a workstation was the destination

Lethality-4
No system process runs on port 0. However the worst case is that this is a Trojan that if the destination host responded it could lead to drastic consequences.

System Countermeasure-4
Again assuming operating system is fully patched, with updated Anti-Virus software running.

Network Countermeasure-4

The destination host did not respond with a reset packet so its safe to assume that the packet was dropped by the firewall thus explains why we see the retransmit packets

Severity = -2

(criticality 2 + lethality 4) - (system countermeasures 4 + network countermeasures 4)

9 Defense Recommendations:
A Keep snort updated
B Keep firewall updated
C Block his IP range (211.47.255.X range)
D. Add the IP range to MYNETWATCHMAN
E. Run anti-Trojan software on the destination host
F. Report this activity to Dshield.org

10 Multiple Choice Question:
Why is the Maximum number of port numbers 65,535?
A Because the port fields are 2 byte fields
B Because the port fields are 4 byte fields
C Because the port fields are 16 byte fields
D Because the port fields are 32 byte fields

Answer is A

Network Detect 3: Gnutella

1.Source of Trace
<http://www.incidents.org/logs/Raw/2002.5.18>
Both IP and TCP/UDP checksums are invalid, due to obfuscation of the original IP addresses of the monitored network. The data made available for these detects contained only the raw data captured by Snort as the result of signatures being triggered. This limits us as far as analysis of possible response to stimuli is concerned, as the responses would only be logged if they too triggered Snort signatures.

2. Detect generated by
This detect was discovered using the Snort IDS 1-9-1
Further analysis was done using Windump 3.0

SNORT DETECTS

Rule used:
alert tcp any any -> any any (msg:"P2P GNUTella"; content:"GNUTELLA CONNECT"; depth:40;)

In this rule we see some variables that we haven't discussed yet (good example of showing the flexibility of IDS used today). We have the variable ***content*** that is used to look for a string in this case "GNUTELLA CONNECT". This will be case sensitive thus the string "Gnutella Connect" will not be detected. We can add the option called ***nocase*** that will negate the case sensitive issue. We also see the option ***depth*** this option tells snort how many bytes into the packet before it begins searching for a content string….in this case 40 bytes. Depth options are used to help speed up snort's processing ability (in this case it keeps snort from searching the first 40 bytes of the packet) thus improving Snort's performance

* for reference to snort flags see page 14

```
[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:26:54.244488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:28371 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:27:58.244488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:15747 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:29:02.254488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:58457 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:30:06.254488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:40849 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:31:10.254488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:7981 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:32:14.264488 148.63.153.51:2466 -> 46.5.80.149:6346
```

```
TCP TTL:111 TOS:0x0 ID:25427 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:33:18.264488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:46353 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:34:22.264488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:5227 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20

[**] [1:0:0] P2P GNUTella [**]
[Priority: 0]
06/28-00:35:26.264488 148.63.153.51:2466 -> 46.5.80.149:6346
TCP TTL:111 TOS:0x0 ID:32085 IpLen:20 DgmLen:174 DF
****P*** Seq: 0xEE51C20A  Ack: 0x0  Win: 0x2000  TcpLen: 20
```

WINDUMP

This filter was used to verify the Snort Alert

```
C:\Snort\etc>windump -r 2002.5.28 -nvvX host 148.63.153.51
```

RULE:

```
alert tcp any any -> any any (msg:"P2P GNUTella"; content:"GNUTELLA CONNECT"; depth:40;)
```

We see the string in the Hex output
(see highlight)

```
00:26:54.244488 148.63.153.51.2466 > 46.5.80.149.6346: P [bad tcp cksum faf7!]
3998335498:3998335632(134) win 8192 (DF) (ttl 111, id 28371, len 174, bad cksum
f86e!)
0x0000   4500 00ae 6ed3 4000 6f06 f86e 943f 9933        E...n.@.o..n.?.3
0x0010   2e05 5095 09a2 18ca ee51 c20a 0000 0000        ..P......Q......
0x0020   5e08 2000 7c9c 0000 474e 5554 454c 4c41        ^...|...GNUTELLA
0x0030   2043 4f4e 4e45 4354 2f30 2e36 0d0a 5573        .CONNECT/0.6..Us
0x0040   6572 2d41 6765 6e74 3a20 4d6f 7270 6865        er-Agent:.Morphe
0x0050   7573 4f53 2031 2e39 2e31 2e30 0d0a 582d        usOS.1.9.1.0..X-
0x0060   556c 7472 6170 6565 723a 2054 7275 650d        Ultrapeer:.True.
0x0070   0a58 2d4c 6561 662d 4d61 783a 2034 3030        .X-Leaf-Max:.400
0x0080   0d0a 582d 5175 6572 792d 526f 7574 696e        ..X-Query-Routin
0x0090   673a 2030 2e31 0d0a 5570 7469 6d65 3a20        g:.0.1..Uptime:.
0x00a0   3044 2030 3548 2033 304d 0d0a 0d0a            0D.05H.30M....
```

**** For brevity we will not include the packet dump for all the alerts. The payload is the same as the packet dump above.

3 Probability the source address was spoofed:
I do not think the address was spoofed. Gnutella is a program that is used for sharing files (mostly mp3s and videos but could be anything from docs, games, and pictures) between users. They would need to connect with other host to exchange files which could not be done with spoofed address. Though I DON'T believe this is spoofed but, it is possible. Gnutella has the ability built into the GUI that would allow a user to spoof their IP address.

4 Description of the Attack:
I don't believe this is an attack. Gnutella users are searching for files they want to download. By looking at the time stamps we see exactly every 1 min and 4 seconds we see TCP traffic with the Push flag set and inside the payload we see information about hops, uptimes. I believe this is the packets Gnutella host use to keep up with who is available on the "Gnutella network". We see the Push flag is set in all the alerts. Generally the Push flag is set when one host wants the destination host to push the data up the TCP stack because the sender is now waiting for a response. An example is when the destination host's TCP window is larger than the data sent to it.

Is there any downside to Gnutella? Due the ability of Gnutella to spoof a users IP address, a particular host could advertise that he is sharing a popular file (for instance the new Terminator part 3 movie before it hits the movie theaters) but spoof the IP address of someone else thus causing traffic to be directed to the Spoofed IP address of his choice (for instance whitehouse.org). However the major threat though is the possibilities of a virus or worms being spread via p2p users. Virus maybe contained in the files down loaded upon which if executed can cause trouble. In February 2001, the Mandragore worm spread rapidly throughout the Gnutella network as well as Nomad and Nimda. Many p2p applications are loaded with Trojans and spyware that can monitor everything from keystrokes to search engine queries even personal information used to fill out online forms and lets not forget the awesome pop up ads that everyone loves to get while online. The other major threat is when un-careful users of p2p programs share files that should not be shared. A simply search on Kazaa for the word "finance" can produce some interesting corporate financial information that probably should not be shared over the internet. Gnutella has also been known to consume large amounts of bandwidth.
http://infosecuritymag.techtarget.com/articles/february01/cover.shtml

5 Attack Mechanism:
Gnutella works by what is known as peer to peer file sharing without a main server. A host does a search for a particular file. The search begins by the host querying 10 other host that it knows about. If the file is not found by any of those 10 host all 10 host will

turn around and will query 10 more host each. This process continues until the file is found. The damaging potential lies in that a file maybe downloaded into the network that has a virus, unprotected file sharing, as well as bandwidth consumption.

6 Correlations:
A search for Gnutella on www.cert.org <http://www.cert.org> did not produce any results.
A search on google found this URL that discuss how some virus/worms can spread via p2p programs. <http://www.unwantedlinks.com/Guntella-alert.htm>
One can ask any college administrator about how much bandwidth Gnutella and other p2p users consume.

A look up on Dshield.org has no reports of malicious activity from this Source IP.

7 Evidence of active targeting:
This is not active targeting. Not active targeting in the traditional sense (an attacker appears to be in possession of sufficient information about your site to be able to go right after a certain component). Per the time stamps and the information in the payload this traffic is from the Gnutella program from it being used for its intended purpose.

8 Severity:
Critically 1
We assume this is a user at their personal computer who is running an updated anti-virus program.

Lethality 1
This is a single packet that comes every 1 minute and 4 seconds that is used by the Gnutella program to function.

System Countermeasures 4
Again we assume the workstation has all current patches and running anti-virus software.

Network Countermeasures 4
We assume the company has a firewall.

Severity = -6

(criticality 1 + lethality 1) - (system countermeasures 4 + network countermeasures 4)

9 Defensive Recommendations:
A. Make sure anti-virus software is current
B. Investigate site policy. If p2p software is allowed then ignore this traffic. If the software is not allowed then remove the software and set up rules in the IDS to detect Gnutella by searching for “GNUTELLA CONNECT” string in the payload. It is to easy for Gnutella users to change ports and IP address thus it would be too difficult to detect the

Gnutella traffic using IP's and ports for the signature. The campus can attempt to use Snort's Flex response to send RST packets to both hosts anytime it detects the string. It should be noted that much of the activity that occurs on p2p networks (sharing of mp3's) is currently considered Illegal in the United States.

10 Multiple Choice Question:
What ports below are associated with Peer to Peer (p2p) programs
A 6699
B 1214
C 6346
D all of the above

Answer is D

A 6699 - Napster
B 1214 - Kazaa
C 6346 - Gnutella

Assignment 3: Analysis This

SUMMARY:
This section is where we analyzed five days worth of logs for a campus network. We have used logs from 7/27/03 to 7/31/03. We have broken this assignment down into five Main parts:

Alert Analysis
Scan Analysis
OOS Analysis
Registration information on five offending external host
Conclusions and final recommendations

Most of the number counts mentioned in the below analysis is accurate however there may be a small difference for some alerts. There were a number of different "malformed" alert formats in the logs used. These malformed alerts were removed from the logs prior to analysis. Overall the counts represent a very accurate picture of the state of the network.
.
Please see chart below for log files used for this assignment

Alert Logs	Scan Logs	OOS Logs
alert.030727	scans.030727	OOS_Report_2003_07_27_18859.txt
alert.030728	scans.030728	OOS_Report_2003_07_28_29050.txt
alert.030729	scans.030729	OOS_Report_2003_07_29_23718.txt
alert.030730	scans.030730	OOS_Report_2003_07_30_29913.txt

alert.030731	scans.030731	OOS_Report_2003_07_31_11092.txt

Alert Analysis

SUMMARY:
The Alert logs were analyzed and sorted out by volume.. We will cover the top 13 alerts based on volume (I stopped at 13 because they made up the bulk of all the alert logs) but, there are other alerts that weren't triggered as much but, due to the severity of the alert itself we will cover these as well in the section Priority Alerts. These section is divided into four Main parts:

Top Alerts. These are the top 13 alerts that were triggered the most. We will cover each alert individually including a brief explanation of the alert itself, possible recommendations or compromises, and show some of the top offenders.

Priority Alerts. These alerts carry a high severity due to the nature of the alert We will cover an explanation of these alerts as well as possible compromise, recommendations, and top offenders.
Top 10 Talkers This is a list of the IP addresses that triggered the most alerts.

LINK Graph This is a graph that will help show correlation between alerts and host/s involved in the compromise of internal host MY.NET.60.16

TOP ALERTS

The chart below shows the top 13 alert signatures based on total number of alerts. I did not include the scan attempts (except Null scan) that were in the alert logs since I cover the scans in the scan logs.

Alert Name	27th	28th	29th	30th	31st	total #
CS Webserver	12409	24488	24402	23676	25104	110079
SMB Wildcard	6043	14069	12736	10545	7317	50710
IIS unicode attack	3855	7234	9704	7547	4199	32539
Queso fingerprint	1457	3499	2208	1479	1772	10415
CGI NULL Byte	5242	1190	1238	1228	549	9447
Exploit x86 NOOP	2648	618	687	1298	970	6221
TCP High port 65535	726	611	355	196	156	2044
UDP High port 65535	635	230	231	17	137	1250
tiny fragments	21	706	209	507		1443
connect to 515 from outside		261	286	358	162	1067

SUNRPC highport access	128	20	828	11	31	1018
IDS552/web-iis_IIS ISAPI	159	233	230	157	116	895
Null scan	127	137	111	269	71	715

**** TCP and UDP high port 65535 alerts were combined as one signature**

CS Webserver: 110,079 alerts.
This is traffic that is directed at a web server. I believe this rule is set for only one IP address. All Source IP's went to the same destination host MY.NET.100.165

Source IP	# of alerts	FQDN
216.39.48.2	15489	trek21.sv.av.com
66.77.73.164	1391	cr005r01-3.sac2.fastsearch.net
216.88.158.142	1251	crawlers.looksmart.com

Recommendations: Review why this rule was setup. IF this host is not suppose to be reached from the internet then the traffic needs to be blocked with ACL's on the border router

SMB Wildcard: 50,710 alerts.
Server Message Block (SMB) This is a protocol that is used for file and print sharing that runs on Netbios. UDP port 137 is used for file sharing on Microsoft machines as well as Netbios name to ip address look ups. This is normal traffic for the internal network but should not be allowed into the internal network from the internet. There is an old worm called network.vbs that would run on UDP port 137. People may scan for open udp 137 ports and then will try to connect on tcp port 139 to access a shared resource.

Top offenders

Source IP	**# alerts**	**FQDN**
169.254.45.176	5561	*Unable to resolve address*
64.228.212.245	1977	HSE-Montreal-ppp143096.sympatico.ca
64.228.213.12	1624	HSE-Montreal-ppp143117.sympatico.ca
64.228.214.41	1421	HSE-Montreal-ppp143400.sympatico.ca

Recommendations: Block UDP 137,138 and TCP 139 as well as port 445 TCP (SMB runs directly over TCP/IP on port 445) at perimeter firewall.

IIS Unicode attack: 32539 alerts.
This is an alert that is based on a vulnerability in Microsoft's IIS 4 and 5 web server. The attack works by using Unicode in a URL string to do a directory transversal allowing access to other files and folders on the web server. There are worms that have created

havoc in the past based on this type of attack such as Code Red, Nimda, and sadmind. Snort has a preprocessor that will try to convert Unicode back to ASCII. However there are many websites that use an escape character "/" in normal URL strings. This preprocessor can hog Snort's resources so it does have the ability to be turned off as well. Patches have been issued out for all of the above mentioned worms. Looking at the graph below, all the alerts are internal but are only going to 1 destination IP. This suggests that these alerts (for the internal host listed below) are most likely false positives. The nature of the worms mentioned is to scan many different hosts looking for other victims to replicate itself to.

Top Internal IP

Source IP	# alerts	dst hosts
MY.NET.153.153	1737	1
MY. NET.97.183	1503	1
MY.NET.97.16	1319	1

Top Offending IP's

Source IP	FQDN	# Alerts	dst hosts
203.172.26.89	*Unable to resolve address*	148	29
211.93.108.180	*Unable to resolve address*	118	32
211.90.183.252	*Unable to resolve address*	76	16
211.93.108.180	*Unable to resolve address*	144	30
202.96.193.106	*Unable to resolve address*	144	31
202.96.193.106	*Unable to resolve address*	107	34

Looking at the number of alerts by the different destination IP's it's probably safe to assume these external host are infected with one of the above-mentioned worms.

Recommendations: Make sure all webservers running IIS have current patches. Make sure all users have their PC's running current patches as well. Make sure any Microsoft machines that are not acting as web servers do not have IIS loaded.

Queso fingerprint: 10415 alerts

Queso is a utility used for fingerprinting OS's. It works by sending out packets with various TCP flags set to try to get a particular response that can help Identify the OS.
See URL http://www.digitaltrust.it/arachnids/IDS29/event.html
Also see CVE can-1999-0454

Top offenders

Source IP	FQDN
216.95.201.21	smtp11.dbhits.com
216.95.201.22	smtp12.dbhits.com
209.47.197.16	smtp6.amermail.com
216.95.201.15	smtp5.dbhits.com
216.95.201.18	smtp8.dbhits.com
209.47.197.18	smtp8.amermail.com

209.47.197.14	smtp4.amermail.com
216.95.201.11	smtp1.dbhits.com
209.47.197.17	smtp7.amermail.com
216.95.201.20	smtp10.dbhits.com
209.47.197.15	smtp5.amermail.com
209.47.197.13	smtp3.amermail.com
216.95.201.19	smtp9.dbhits.com

Recommendations: I believe almost all of the queso alerts are false positives The offending IP's belong to email management companies. These companies also set off the majority of the OOS alerts as well. I believe the routers are ECN/CWR aware and there was network congestion which turned on the ECN/CWR flags that set off the queso signature in Snort. Please see URL http://www.securityfocus.com/infocus/1205

There were only a few alerts from outside the networks above. These alerts are more likely to be genuine alerts of OS fingerprint attempts. See below for the Source IP's for the likely OS fingerprint attempts

Source IP	**FQDN**
81.52.246.220	*Unable to resolve address*
66.48.78.13	smtp3.dcswx.com
217.9.225.6	block54-ibgc-int.interbgc.com

CGI Null Byte: 9447 alerts

This alert is triggered when the snort preprocessor detects a %00 in a URL string (this can be turned off by putting –cginull in the preprocessor line to save on snort resources). This is very similar to how the alert for the IIS Unicode exploit gets detected in that it's looking for Unicode with the escape character. This %00 is used to cause a Perl script to error and show source code to the remote attacker for a particular file (such as password etc). See URL http://archives.neohapsis.com/archives/bugtraq/2002-01/0065.html These may be mostly false positives particularly on port 443 which is used for SSL (encryption protocol).
See URL http://archives.neohapsis.com/archives/snort/2000-11/0244.html for reference. To verify one would need to look at the traffic with ethereal, tcpdump etc etc. Nearly all alerts came from internal hosts.

Top Offenders

Source IP	**# Alerts**
MY.NET.97.133	4737
MY.NET.81.58	454
MY.NET.53.32	548
MY.NET.70.17	324

Recommendations: Most of the alerts are probably false positives. Nearly all alerts are outbound most likely to sites that have Unicode "%00" in the URL strings.

Exploit x86 NOOP: 6221 alerts
This alert is triggered when a large number of NOOP bytes (Ox90 or Ox61) are detected. NOOP bytes don't do anything they are used to help pad code (assembler language) but, an excessive number of NOOP bytes can be used to cause buffer overflow attacks. Usually buffer overflow attacks happen by overloading a memory buffer that doesn't do error checking on input and causing the execution of some code. This alert has been known to false positive particularly with http and ftp (binary traffic). Most buffer overflow attacks have patches available to render the exploit useless. For an interesting paper on how buffer overflow attacks work please see URL http://www.phrack.org/show.php?p=49&a=14

Recommendations: Check to see if offending IP's are part of the "safe list" (see chart below). Make sure systems are running current patches.

Source IP	FQDN
195.185.187.102 ***	ns2.unix-admin.net
131.118.254.130	news.ums.edu
129.237.39.5	seagull.cc.ku.edu
217.231.231.250 ***	pD9E7E7FA.dip.t-dialin.net
200.171.147.23	200-171-147-23.speedyterra.com.br
212.202.13.104	port-212-202-13-104.reverse.qsc.de
217.85.146.27	pD955921B.dip.t-dialin.net
213.93.84.190	e84190.upc-e.chello.nl
193.253.221.183	APuteaux-102-1-2-183.w193-253.abo.wanadoo.fr

***The campus should consider blocking IP 195.185.187.102 triggered 1192 attempts to 29 unique destinations as well as IP 217.231.231.50 he triggered 2413 alerts to 20 unique destinations.

TCP/UDP High Port Traffic 65535 –Possible Red Worm: 3294 alerts
(Both TCP and UDP alerts were combined)
Red Worm aka Adore worm works similar to the Lion and Ramen Worms. The Worm attacks Linux boxes that have LPRng, rpc-statd, wu-ftpd and BIND services running. Adore installs a Trojan that upon activation (an icmp packet of a certain length it would then open a root shell to allow a remote user to connect. Adore also scans random B class networks looking for other host to infect. Patches are available to fix this issue. See URL http://www.sans.org/y2k/adore.htm
Also a utility exist to detect and remove the worm. See URL http://www.ists.dartmouth.edu/IRIA/knowledge_base/tools/adorefind.htm

This chart shows host that should be investigated

Source IP	# alerts	unique dst IP's
MY.NET.153.223	255	28
MY.NET.152.45	39	5
MY.NET.84.216	47	11
MY.NET.12.4	251	2
MY.NET.86.110	36	9

Recommendations: Make sure all Linux boxes are currently patched. Use the utility mentioned above to remove any cases of infection. Host MY.NET.153.223 should be investigated immediately. This host alerted 255 times to 28 different destination IP's.

Tiny Fragments: 1443 alerts

This alert is when we see IP packet that is too small. In Snort2.0 the threshold is set to alert if the fragments are smaller than 25 bytes. There is no reason why any normal traffic would get fragmented that small unless the hardware is going bad or the software has a major bug in it. Traffic can become fragmented when it has to travel across a MTU (maximum transmission unit) that is smaller than the packet itself but the smallest MTU is 576. Anytime traffic is below 25 bytes something is suspicious. Nmap and fragrouter can fragment traffic to 24 bytes and 8 bytes. Nmap is used to fingerprint OS and Fragrouter is used for evading IDS via insertion and evasion tactics.
There are 2 main offenders for this alert. Both targeted the same destination.
MY.NET.113.4

Source IP	FQDN	# alerts	dst
24.34.64.248	h00095b1f728a.ne.client2.attbi.com	912	1
24.240.149.34	24-240-149-34.charter.com	491	1

Recommendations: This traffic needs to be dropped at the perimeter firewall. I also recommend blocking the two IP address above at the firewall as well as follow up with an email to their ISP.

Connect to port 515 from outside: 1067 alerts

This alert gets triggered when it external traffic is going to port 515. Port 515 is the LPD. There has been an exploit on this port that gave users root (because you had to be Root to run the service per RFC 1179). See URL http://www.lprng.com/LPRng-HOWTO-Multipart/setuid.htm
See also CVE-2000-0917.

We have one offending IP attacking the same victim IP

Source IP	FQDN	# alerts	Destination IP
131.118.229.7	wizards.usmsc.edu	1067	65.40.24.15:515

Recommendations: Immediately pull host 65.40.24.15 off line to make sure it has the most current version of LPD (LPRng 3.6.24 is the vulnerable version)
Block traffic inbound to port 515 at the firewall (no need to share campus printers with the world).

SunRPC high port access: 1018 alerts
RPC are services that run on both Unix and Windows OS. This daemons allow for remote PCs to interact with each other. RPC daemons have had a number of known vulnerabilities over the years. The main reason is RPC daemons do not perform strong error checking on input which makes the services very susceptible to buffer-overflow attacks (putting too much data into the buffers memory which can allow for malicious code to be ran). Most of the RPC services run as root so many of the exploits will allow a root compromise. Some of the CVE's are

CVE-1999-0002	CVE-1999-0019	CVE-1999-0170
CVE-1999-0003	CVE-1999 -0166	CVE-1999-0208
CVE-1999-0008	CVE-1999-0167	CVE-1999-0211
CVE-1999-0018	CVE-1999-0168	CVE-1999-0493

IP 131.118.254.130:119 FQDN: news.ums.edu alerted over 823 times on the 29th to MY.NET.24.8:32771. Then the offender set off the exploit x86 setuid0 alert going to MY.NET.24.8 on port 119 (the SKA Trojan/Happy Trojan work on this port)

Recommendations: Pull host MY.NET.24.8 offline and investigate for compromise. Turn off all RPC services that are not needed and keep tight control on those who do. Block traffic to ports 111 (Unix) and 135 (windows) These are port-mapper services that would show an attacker what RPC services are running. Make sure all the most current patches are running. Block the RPC "loopback" ports, 32770-32789 (TCP and UDP). Please see URL http://www.sans.org/top20/

IDS552/web-iis_IIS_ISAPI Overflow ida INTERNAL nosize: 895 alerts
This alert goes with a vulnerability in Microsoft's IIS webserver. IIS use ISAPI to link other extensions to dll's to further its functionality. These links can be vulnerable. The idq.dll has insufficient error checking of input of URL's which allows for a buffer overflow attack that can lead to executing an attackers code thus creating a compromised host. The Code Red (1 and 2) used this type of attack to propagate. Our main concern with this alert is looking for any internal host that are triggering this alert while attempting to connect to multiple other hosts. We only have one such host On the 30th host MY.NET. 97.81 alerted on this 385 times while going to 244 unique destination IP's. It is clear that this Host has been compromised.

Recommendations: Pull host MY.NET.97.81 off the network and apply current level patches. I also recommend using Microsoft's "IIS Lockdown Wizard" to help harden the configuration.

Null Scan: 715 alerts
This alert is triggered when a TCP with no flags set is detected. These packets are used for scanning. Open ports don't respond to this type of packet but a closed port will send

a RST/ACK. Null packets are also used as part of OS fingerprinting.

The top offender is

Source IP	FQDN	# alerts
67.119.237.120	adsl-67-119-237-120.dsl.sndg02.pacbell.net	275

Recommendations: Check to see if this offending IP address is on a safe-list. If not then block his IP at the perimeter firewall.

Priority Alerts

Although these alerts may have a low number of triggered alerts, they are severe enough that attention needs to be given to them. We will cover how attacks work and give some examples of host that may need immediate attention.

Trojan Activity: Subseven and Back Orifice

Subseven: These alerts are from internal host trying to connect to remote machines on port 27374 or external host on port 27374 trying to connect to the internal network. Port 27374 is the port that a popular Trojan called Subseven works on. Subseven allows remote host to have total control of an infected pc. There can be false positives on this alert. Normal web traffic may go to port 27374 or any low port to ephemeral port. A big clue is when we see ephemeral ports used to connect to port 27374. Hosts below need to be investigated for possible infection.

Subseven Trojan activity	
Source IP's	**Destination IP**
MY.NET.24.58:443	68.50.137.44:27374
MY.NET.29.11:443	68.64.32.32:27374
MY.NET.69.27:3456	65.40.147.144/64.171.5.172:27374
MY.NET.60.16:53268	68.64.32.32:27374

Recommendations: Investigate host for infection Please see URL for removal tools
http://www.symantec.com/avcenter/venc/data/backdoor.subseven.html

Back Orifice: This is another Trojan that works very similar to Subseven and works on port 31337. It allows for a remote user to control a host.
These hosts should be investigated.

Back Orifice	
Source IP	**Destination IP**
66.250.188.10:27525	MY.NET.69.192:31337
12.129.72.202:1430	MY.NET.84.145:31337
65.25.161.66:1547	MY.NET.135.228:31337
65.25.161.66:1547	MY.NET.135.108:31337

65.25.161.66:1547	MY.NET.134.238:31337
65.25.161.66:1547	MY.NET.134.169:31337
65.25.161.66:1547	MY.NET.134.56:31337
65.25.161.66:1547	MY.NET.133.233:31337
65.25.161.66:1547	MY.NET.133.110:31337
65.25.161.66:1547	MY.NET.133.48:31337

Recommendations: Have a system admin investigate the host mentioned above and remove Trojan if found. Please see URL on how to remove Back Orifice.
http://www.nwinternet.com/~pchelp/bo/removingBO.htm

SMB C$ access attempt
This alert is generated when a remote user tries to access the C drive on Windows host. There were many instances of this on the internal network.
This alert is a high priority. Please see URL
http://www.pantek.com/library/general/lists/snort.org/snort-sigs/msg00656.html
There were 153 alerts triggered to many different hosts on the internal network.

Recommendations: block port 139 and port 445 at the border router

NIMDA
Nimda is a vicious worm that spreads multiple ways taking advantage of a number of different Microsoft IIS exploits. The worm will scan looking for other host to infect. Host MY.NET.97.81 appears to be infected with Nimda. This host triggered this alert 144 times to 129 different destinations.

Recommendation: Pull this host off the network immediately and apply the patch.
Please see URL for patch
http://www.microsoft.com/technet/security/bulletin/MS01-020.asp

IRC evil –running XDCC
This is a file sharing server/bot running on IRC that is associated with underground activities such as share pirated software, mp3s, movies etc. There are possibilities of Trojans being installed on the bot that could look for other host to infect.
See URL for details http://www.security.duke.edu/cleaning/xdcc.html
There are only 2 host associated with this alert
My.NET.74.216 and MY.NET.98.221

Recommendation: Investigate these hosts for Trojans as well vital file changes. Consider blocking and banning all IRC activity

FTP Passwd Attempt
Host 212.74.226.145 triggered this alert 74 times in one day. If a hacker is successful in

downloading the passwd file from the FTP server he can use a passwd cracking utility such as john the ripper or LC4. Please see URL http://www.snort.org/snort-db/sid.html?sid=356

Recommendation: Block this IP address at the perimeter and report this activity to Dshield.org

Top 10 Talkers For the Alert Logs

The chart below shows the most active IP address base on total number of alerts.

Source IP	FQDN	total #	Unique alerts	Unique DST
68.48.217.68	pcp04613030pcs.gambrl01.md.comcast.net	15916	16	2
216.39.48.2	trek21.sv.av.com	15489	4	2
68.54.93.211	pcp01781322pcs.howard01.md.comcast.net	11454	3	1
68.18.29.200	adsl-18-29-200.rdu.bellsouth.net	5542	2	2
65.40.97.133	user133.net412.tx.sprint-hsd.net	4738	2	1
66.82.245.45	dpc6682245045.direcpc.com	2554	4	2554
217.231.231.250	pD9E7E7FA.dip.t-dialin.net	2413	1	20
65.40.153.153	user153.net468.lv.sprint-hsd.net	2262	1	1
64.228.212.245	HSE-Montreal-ppp143096.sympatico.ca	1978	2	2
24.35.42.249	cmu-24-35-42-249.mivlmd.cablespeed.com	1962	2	2

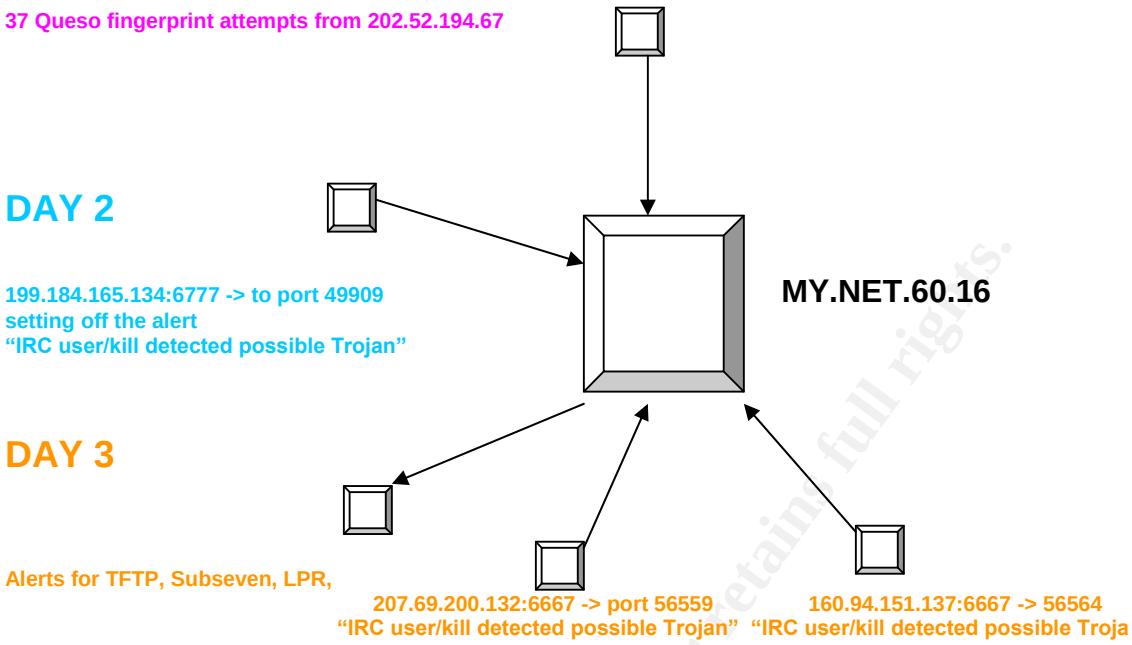
Link Graph:

The link graph below shows the recon and compromise of internal host MY.NET.60.16

Day 1
Shows host 202.52.194.67 triggering 37 queso OS fingerprint alerts

Day 2
Shows host 199.184.165.134:6667 connecting to the internal host MY.NET.60.16:49909 (notice ephemeral to ephemeral) The alert “IRC user/kill detected, possible Trojan”. At this stage I believe the Trojan has been installed and the compromise has happened

Day 3
First we see MY.NET.60.16 trigger the TFTP alert going to 68.64.32.32
After the last alert for TFTP the Subseven alert is triggered going to 68.64.32.32, then followed up with connection to port 515 to 68.64.32.32. We have one more alert for TFTP to 68.64.32.32.
Within minutes of the last triggered alerts going to 68.64.32.32, two new connections MY.NET.60.16 start. Both trigger the alert “IRC user/kill detected, possible Trojan”. (notice the high port numbers)
207.69.200.132:6667 -> MY.NET.60.16:56559
160.94.151.137:6667 -> MY.NET.60.16:56564
I believe these hosts are under control of the same hacker a group of hackers working together.



SCAN Analysis

In this section we will cover the scanning activity that is happening on the network. We will cover the most targeted ports along with the service associated with those ports. We will also show the most active scanning source IP's and the most targeted destination IP's

See graph below for the port numbers that were most scanned as well as recommendations for each port scanned.

port	attempts	Service	Recommendations:
53	2227855	DNS	Ignore
137	665555	Netbios	block at Perimeter
80	545039	http	Ignore
6257	269450	Winmx (p2p file sharing program)	block at Perimeter/
21	77074	FTP	Ignore
445	64650	SMB	block at Perimeter
25	62731	SMTP	Ignore
6346	60375	Gnutella (p2p file sharing program)*	block at Perimeter
443	38912	SSL	Ignore
17300	38348	backdoor-kuang2v	block at Perimeter
7674	34454	IMQ SSL	Ignore
134	30393	INGRES-NET	Ignore
22321	23654	Backdoor.Dobol	block at Perimeter

4000	23398	video game Command and Conquer	Ignore
139	21513	Netbios	block at Perimeter

* Gnutella should be blocked using Snort's flex response vs. attempting to block by port 6346.

See graph below for the Top Talker's. These Source IP's had the most attempts scanning other host.

Source IP	Total #
130.85.1.3	1942362
130.85.1.4	284940
130.85.153.223	268725
130.85.82.2	183893
130.85.97.83	110012
130.85.97.53	87890
130.85.114.88	76668
130.85.97.68	75666
130.85.97.52	73536
130.85.97.88	70526
130.85.97.42	67540
217.84.34.106	57093
130.85.97.11	56673
63.250.195.10	55234
130.85.100.230	54687

Almost all the top scanners are from the internal network.

See graph below. These IP address were the highest scanned Destinations.

Destination IP	Total #
192.26.92.30	63355
205.231.29.244	55263
130.85.198.221	54330
192.148.252.171	43669
130.94.6.10	39430
192.52.178.30	37053
205.231.29.243	32947
192.5.6.30	28620
216.109.116.17	24288
66.33.98.17	23336
65.120.116.6	20120
209.208.92.254	19790
212.100.230.160	18834
213.130.63.232	18198
194.109.6.154	16622

OOS Analysis

SUMMARY:

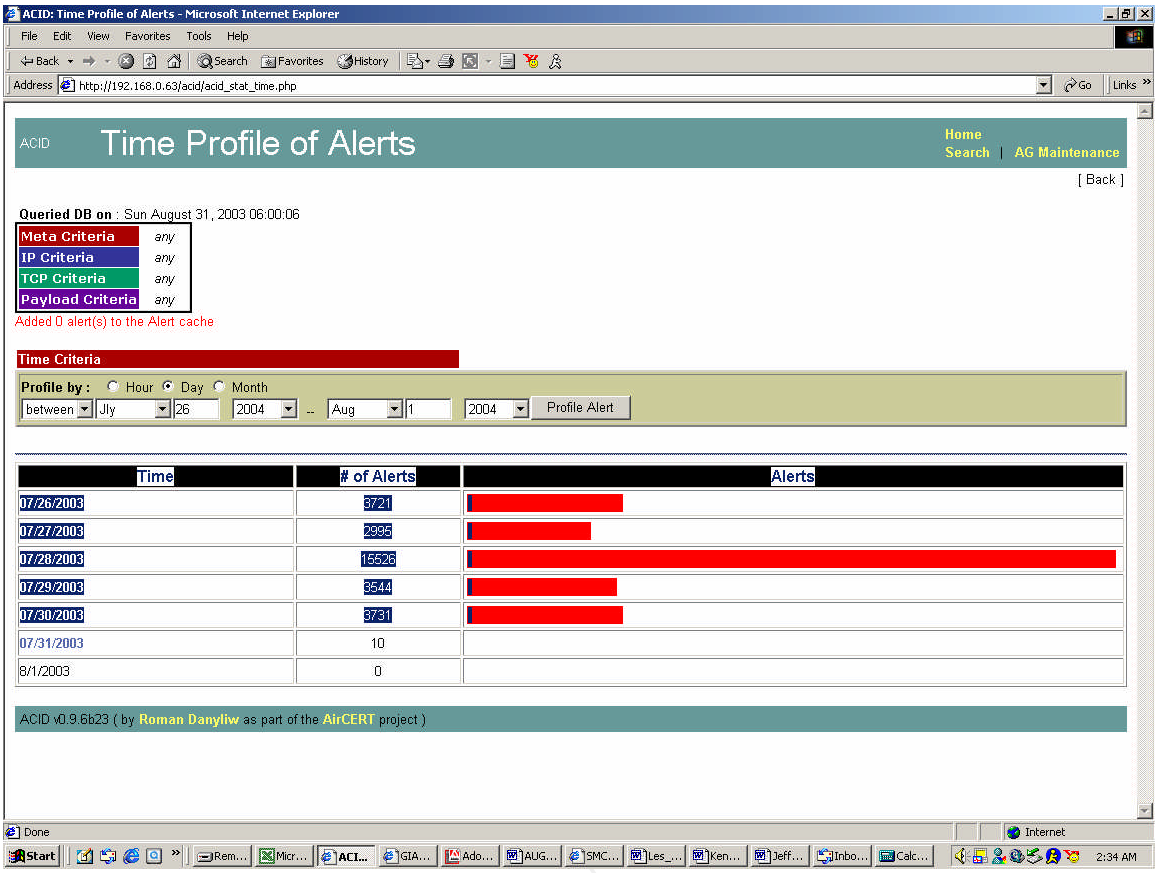
Out of Spec packets are packets that don't fit within RFC guidelines.
These packets are usually are cause from two things

1) Packets that have become corrupted either due to a broken application or corrupted while in transport (For example if the T1 circuit is malfunctioning then packets can easily become errored).
2) Packets that are purposely crafted in such a way to get certain type of response. This is what is used for OS fingerprinting and port scanning, IDS evasion, exploits (ex. Cisco's recent vulnerability http://www.cisco.com/warp/public/707/cisco-sa-20030717-blocked.shtml)

Here is a chart on the total number of OOS alerts

Total # of OOS alerts	Total # of Source IP's	Total # of Destination IP's
29527	442	7752

One of the first things I noticed was that there was a large swell of OOS alerts that came in on 7/28/03. It was about 5 times more than the other days. Below is snapshot of an Acid bar graph that shows the drastic rise in alerts.



I was curious as to why such a large swell. I discovered that Source IP 66.82.245.45 attempted to scan 7,644 IP address looking to possible fingerprint FTP servers (all attempts were to port 21 from port 21) The same day I also discovered 193.41.64.2 sent 2641 syn packets and 217.9.225.6 sent 2283 syn/fin packets to my.net.29.66 IP address to port 80 in attempted syn flood DOS attack. This accounts for 12568 alerts. The remaining 2958 alerts is very similar to the amount of alerts we see normally see on the four other days.

Here is a chart of the top 25 talker's source IP's based on number of alerts for the OSS logs. I choose the top 25 because most of the Source IP address belonged to the same subnet. Using the top 25 gives a better view of the OOS activity on the campus network.

Source IP	FQDN	# OOS alerts	Unique DST IP's
66.82.245.45	dpc6682245045.direcpc.com	7644	7644
193.41.64.2	proxy.bgnet.bg	2641	1
217.9.225.6	block54-ibgc-int.interbgc.com	2283	1
216.95.201.15	smtp5.dbhits.com	797	15
216.95.201.22	smtp12.dbhits.com	716	14
168.226.117.207	168-226-117-207.speedy.com.ar	692	3

216.95.201.18	smtp8.dbhits.com	606	13
216.95.201.17	smtp7.dbhits.com	598	14
216.95.201.20	smtp10.dbhits.com	588	14
216.95.201.12	smtp2.dbhits.com	568	15
216.95.201.23	smtp13.dbhits.com	530	13
216.95.201.16	smtp6.dbhits.com	528	14
67.119.237.120	adsl-67-119-237-120.dsl.sndg02.pacbell.net	509	5
216.95.201.11	smtp1.dbhits.com	432	15
213.186.35.9	ns336.ovh.net	430	8
216.95.201.28	smtp18.dbhits.com	417	12
216.95.201.21	smtp11.dbhits.com	411	14
202.52.194.67	*Unable to resolve address*	394	6
216.95.201.19	smtp9.dbhits.com	373	14
216.95.201.13	smtp3.dbhits.com	358	13
209.47.197.16	smtp6.amermail.com	345	9
209.47.197.15	smtp5.amermail.com	296	12
209.47.197.14	smtp4.amermail.com	292	11
216.95.201.27	smtp17.dbhits.com	280	13

Below is a detail summary of some of the unique sources from the above chart

Source IP 66.82.245.45
This host was mentioned above. This host launched a massive scan of 7,644 attempts looking for port 21 (FTP) with a source port of 21 as well. The scan started out scanning every 35th address. I am not sure why the scan makes such an attempt. If it was looking for broadcast IP address/mapping the network I could understand attempting to find the Network IP address of a subnet vs. attempting every single IP address inside the IP block. Here is sample of how the scan looked.

My.net.2.12
My.net.2.47
My.net.2.48
My.net.2.83
My.net.2.84

…. For brevity

My.net.2.192

Then we see the scan suddenly turn to incrementing by 1

My.net.3.9
My.net.3.10
My.net.3.11
My.net.3.12

…. For brevity

Then the scan would become random by changing octets while half way through the previous octet.

All these attempts were with crafted packets. Both the SYN and FIN flags are set (Syn to start a connection and Fin to end) This type of packet is used for OS fingerprinting. Some versions of Linux (2.2.x) would respond to this type of packet with a syn ack and windows boxes would send a reset ack packet back.
See URL http://archives.neohapsis.com/archives/snort/2000-07/0229.html
This is most likely traffic generated by a scanner called Synscan. Synscan uses syn/fin packets and the DST port equals the SRC port. Please see URL http://www.dshield.org/pipermail/list/2003-July/009146.php

A lookup on dshield.org shows that there have been 110 records file against this source IP.

Recommendations: Block this IP at the firewall and at the router with ACLs and report the activity to dshield.org

SOURCE IP 193.41.64.2 And SOURCE IP 217.9.225.6
These hosts appear to have launched a SYN flood against a possible web server (my.net.29.66). IP address 217.9.225.6 sent 2,641packets and IP address 193.41.64.2 and 2,283 packets from 217.9.225.6 were sent to port 80 in a rapid session. SYN floods work by sending a packet with just the SYN flag on (to start a connection) but the intention here is not to complete the 3-way handshake but to open up "half connections to a victim" When the victim receives the SYN packet is allots a portion of memory for the upcoming TCP session. If enough SYN packets are sent, it can tie up enough memory to cause a denial of service attack on the victim via lack of memory resources. The attacker will usually spoof a dead source IP address to cover his Identity. If the victim sent a syn ack to a live spoofed IP address that host would respond with a Reset Ack and thus that particular TCP connection would end and would free up the memory it was using. The reason OOS rules caught these packets were because we see the ECN flag (Explicit Congestion Notification and the CWR (Congestion Window Reduced) flag was turned on for legit reasons. It is possible that these flags are turned on legit ably. Routers that are ECN aware will notify each other (in the IP header TOS field) that it is capable and if there is congestion on the network. However these flags have been used for OS fingerprinting

Recommendations: Limit the amount of memory that can be used by TCP connections or by using TCP cookies see this URL http://cr.yp.to/syncookies.html
I recommend blocking the IP's at the firewall (though these IP's are most likely spoofed)

SOURCE IP 216.95.201.15, 216.95.201.17, 216.95.201.18, 216.95.201.22, 216.95.201.20
216.95.201.12, 216.95.201.23, 216.95.201.16, 216.95.201.11, 216.95.201.28, 216.95.201.21,
216.95.201.19, 216.95.201.13, 216.95.201.27
These hosts sent packets with the SYN flag set to port 25 on the destination host. Port 25 is Simple Mail Transport Protocol (SMTP) SMTP is the protocol used between mail client and mail server communications. Spammers have been known to compromise a pc that has port 25 open and use it as a relay to send out their spam (unsolicited email…not the meat ☺). These hosts scanned the exact same 15 different IP address with a packet that had the SYN flag set as well as the ECN/CWR flags. The primary destination was host MY.NET.12.6. There were over 3010 attempts to this destination almost half of this source IP's activity (7202 total attempts)
There have been a number of Trojans over the years that listen on port 25. Some of them are Ajan, Antigen, Email Password Sender, Gip, Haebu Coceda (=Naebi), Happy 99, I Love You, Kaung2, Pro Mail Trojan, Shtrilitz, Stealth, Tapiras, Terminator, WinPC, and WinSpy ù. For a full listing go to http://www.treachery.net/security_tools/ports/ and search on port 25. Also there are a number of buffer overflow exploits www.giac.org/practical/GCIH/stephanie_alarcon_GCIH.pdf has a good explanation of the buffer overflow attack associated with CVE CAN-2002-1337.
See URL http://isc.incidents.org/port_details.html?port=25 for more CVEs associated with port 25.

These Source IP's belong to a company called Sender Base. Sender Base is a service that mail administrators can use to help identify known spammers and their spam mail. Please see URL http://www.senderbase.org/?page=help It is most likely that this is legitimate traffic into the university.

RECOMMENDATIONS: Confirm the use of Sender Base. Implement rules in the IDS that if the Source IP matches the above IP address from SenderBase on port 25 to "pass". This will keep this traffic from alerting in the IDS.

SOURCE IP 168.226.117.207
This Host is sending packets to 3 separate IP address with the syn flag set.
The traffic break down is
MY.NET.84.235 -----287 attempts
MY.NET.112.196 ----404 attempts
MY.NET.80.105 -----1
All packets are sent to port 4662. Which is the default port for Edonkey
See http://www.edonkey2000.com/faq.html and
http://www.seifried.org/security/ports/4000/4662.html for more info on Edonkey.
Edonkey is a peer-to-peer file-sharing program. Most likely mp3's are the files that are being shared and destination host obviously are "sharing" their files. The Source IP belongs to Speedy.com which is a car service business. Most likely this is an employee of Speedy who wants to get some music to listen to while at work.

RECOMMENDATIONS: Block traffic for this port at the firewall. Remove the file sharing software from the destination hosts and remind users of the copyright laws. The RIAA is currently on a relentless rampage pursuing legal actions against those “sharing” mp3’s “illegally” even poor under-aged 12 year old girls. Please see URL http://www.slyck.com/news.php?story=234

SOURCE IP 67.119.237.120
This host sent 509 packets to 5 unique destination IP’s on port 110 (pop3 mail) with most (255) of those packets going to MY.NET.12.4. These are TCP packets that do not have any flags set are also known as Null packet. Null packets are associated with scanning. This is one of several packets used with Nmap while attempting to do OS fingerprinting. See URL for more info on NULL packets work with OS fingerprinting http://honeynet.hackers.nl/scans/scan23/sol/Vivek.html When the Null packet goes to a closed port a RST/ACK is sent if the port is open there is no response. There is a Trojan that listens on port 110 (promail Trojan see URL http://www.symantec.com/avcenter/venc/data/promail.trojan.html)

RECOMMENDATIONS: All Null packets need to be stopped at the Firewall Perimeter. Null packets are not used for communication but, are used for OS fingerprinting and thus should be blocked and not allowed into the network.

INTERNAL HOSTS
HOST MY.NET.12.4 and MY.NET.12.6

These two hosts were the only internal IP’s that set off any OOS alerts. There were only a total of 46 alerts between both of them for all 5 days. I investigated each individual packet and all packets were TCP with the RST flag set along with the ECN/CWR. These packets could be the response to OS fingerprinting or just a packet with the RST/ over a routing path that is ECN/CWR aware with current network congestion (This was picked up by OOS because the ECN/CWR flags were set). Both hosts I believe are mail servers due to the volume of traffic to ports 110 and 25 which explains why they may have network congestion problems to these two hosts.

Registration Information on External Hosts.

We have included the registration information on a few selected Hosts.

66.82.245.45
This host was selected because he scanned our internal network 6,644 times looking for FTP servers on the 28th.
OrgName: Hughes Network Systems
OrgID: HNS

Address: 11717 Exploration Lane
Address: DirecWAY Network Management Center
Address: attn: Network Security Manager
City: Germantown
StateProv: MD
PostalCode: 20876
Country: US
Comment:
RegDate: 1986-08-26
Updated: 2002-11-08
AdminHandle: EG264-ARIN
AdminName: Gillon, Ed
AdminPhone: +1-301-212-7897
AdminEmail: egillon@hns.com
TechHandle: NSM5-ARIN
TechName: Network Security Manager
TechPhone: +1-301-601-7205
TechEmail: abuse@direcpc.com

217.231.231.250
This host triggered the Exploit x86 NOOP alert 2,413 on the 27th
inetnum: 217.224.0.0 - 217.237.161.47
netname: DTAG-DIAL15
descr: Deutsche Telekom AG
country: DE
admin-c: DTIP
tech-c: DTST
status: ASSIGNED PA
remarks: **********
remarks: * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
remarks: * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC. *
remarks: **********
mnt-by: DTAG-NIC
changed: ripe.dtip@telekom.de 20010404
changed: ripe.dtip@telekom.de 20030211
source: RIPE
route: 217.224.0.0/11
descr: Deutsche Telekom AG, Internet service provider
origin: AS3320
mnt-by: DTAG-RR
changed: bp@nic.dtag.de 20010405
source: RIPE
person: DTAG Global IP-Addressing

address: Deutsche Telekom AG
address: D-90449 Nuernberg
address: Germany
phone: +49 180 5334332
fax-no: +49 180 5334252
e-mail: ripe.dtip@telekom.de
nic-hdl: DTIP
mnt-by: DTAG-NIC
changed: ripe.dtip@telekom.de 20030210
source: RIPE
person: Security Team
address: Deutsche Telekom AG
address: Germany
phone: +49 180 5334332
fax-no: +49 180 5334252
e-mail: abuse@t-ipnet.de
nic-hdl: DTST
mnt-by: DTAG-NIC
changed: abuse@t-ipnet.de 20030210
source: RIPE

131.118.254.130
This host was select he attempted 823 time to RPC port on MY.NET.24.8:32771 and later saw this host connect again on port 119 possibly infecting the internal host with the SKA Trojan
OrgName: University of Maryland
OrgID: UNIVER-270
Address: System Administration
Address: 3300 Metzerott Road
City: Adelphi
StateProv: MD
PostalCode: 20783
Country: US
NetRange: 131.118.0.0 - 131.118.255.255
CIDR: 131.118.0.0/16
NetName: MINCNET
NetHandle: NET-131-118-0-0-1
Parent: NET-131-0-0-0-0
NetType: Direct Assignment
NameServer: NS.USMD.EDU
NameServer: UMCPNOC.UMS.EDU
NameServer: NOC.USMD.EDU
NameServer: TRANTOR.UMD.EDU
Comment:

RegDate: 1988-11-15
Updated: 1998-11-24
TechHandle: NM162-ARIN
TechName: Malmberg, Norwin
TechPhone: +1-301-445-2758
TechEmail: malmberg@usmh.usmd.edu

68.64.32.32
This host connected to MY.NET.60.16 on both port 515 (possibly exploiting a LPD vulnerability) and later on port 27374 possibly infecting the internal host with Subseven
CustName: Adelphia
Address: 1 North Main Street
City: Coudersport
StateProv: PA
PostalCode: 16915
Country: US
RegDate: 2002-10-24
Updated: 2002-10-24
NetRange: 68.64.32.0 - 68.64.47.255
CIDR: 68.64.32.0/20
NetName: 6864320-Z5
NetHandle: NET-68-64-32-0-1
Parent: NET-68-64-0-0-1
NetType: Reassigned
Comment:
RegDate: 2002-10-24
Updated: 2002-10-24
TechHandle: AH102-ARIN
TechName: Hostmaster, Adelphia
TechPhone: +1-814-274-0638
TechEmail: ipadmin@adelphia.net
OrgTechHandle: CKI8-ARIN
OrgTechName: Kio, Carolyn
OrgTechPhone: +1-888-512-5111
OrgTechEmail: arin@adelphiacom.net

212.74.226.145
This Host was picked because he attempted 74 times on the 29th to retrieve the passwd file from the FTP server

inetnum: 212.74.224.0 - 212.74.233.63
netname: ITERANET
descr: ITERANET Ltd

descr: ISP in Moscow and Cypris
country: RU
admin-c: MIV5-RIPE
tech-c: IT548-RIPE
status: ASSIGNED PA
notify: noc@iteranet.net
mnt-by: ITERANET-MNT
mnt-lower: ITERANET-MNT
changed: igort@iteranet.ru 20030218
source: RIPE
route: 212.74.224.0/19
descr: ISP ITERANET
origin: AS15682
mnt-by: ITERANET-MNT
changed: igort@iteranet.net 20000914
source: RIPE
person: Igor Matskevich
address: ITERANET Limited
address: Sevostopolsky st. 28-1
address: 113209, Moscow, Russian Federation
phone: +7 095 7255878
fax-no: +7 095 7211447
e-mail: miv@iteranet.net
nic-hdl: MIV5-RIPE
changed: igort@iteranet.net 20000209
source: RIPE
person: Igor Tumkin
address: ITERANET Limited
address: Sevastopolsky av. 28-1
address: Moscow, Russia
phone: +7 095 7265544
fax-no: +7 095 7265522
e-mail: igort@iteranet.net
nic-hdl: IT548-RIPE
notify: noc@iteranet.ru
mnt-by: ITERANET-MNT
changed: igort@iteranet.net 20020212

Internal compromised hosts

I have made reference through out the third assignment on host that may be compromised and should be investigated. This is a consolidated list of the hosts that need immediate attention: MY.NET.60.16, MY.NET.69.27, MY.NET.29.11, MY.NET.24.58, MY.NET.84.145, MY.NET.24.8, MY.NET.24.15, MY.NET.153.223

Recommendations Summary

We have been giving recommendations through out this assignment so we will conclude in this section to bring it all together. The first thing the campus admin need to do is take offline the potentially compromised hosts referred to in the above sections particularly those mention in the Priority Alert section and the Internal compromised hosts section. Ports 135,137,138,139, and 445 need to be blocked at the perimeter firewall both inbound and outbound. If any Microsoft hosts need to communicate in from the outside world strict access should be used and should be under tight monitor. In most cases these ports have no reason to be opened to the world to which they can become easy targets. Port 515 should also be blocked at the firewall both inbound and outbound. All users need to install latest patches and keep Antivirus software updated. There is a large amount of p2p file sharing taking place. The campus should clearly define its policy on this issue (currently sharing mp3's etc is consider illegal in the USA and the RIAA is pursuing legal actions against those who do) and begin enforcement of that policy. There is also a large amount of activity going to the DNS servers. Port 53 was the most scanned port very likely for appropiate reasons however DNS is a big target for hackers. It is important to follow the recommendations made in assignment 2 on the first detect on how to protect the DNS servers (split DNS or split-split DNS if possible). The Sun RPC services should be turned off on any host that does not require them. Windows machines that aren't being used as web servers need to make sure they are not running IIS software. Host based IDS such as Tripwire should be used on "mission critical" hosts such as MY.NET.12.4 and MY.NET.12.6. I saw large amounts of OS fingerprinting to these hosts (I believe these are the campus mail servers). The campus admin should also consider turning off certain ICMP unreachable messages from internal host to help cut down on the potential recon information that external scanners can gather. Keep Snort rules updated. Use Microsoft IIS Wizard to lock down the IIS servers. The campus should keep a updated list of what ports are suppose to be open on particular hosts as well as devices. These hosts and devices should at least on a weekly basis scan for any "new" additional open ports from externally. A good tool to use that is free is Nessus. www.nessus.org

Process used to for Assignment 3

After a while of not knowing how to begin on this assignment I looked to other student's practicals as well as seeking advice from the local Snort users group meetings. I initially thought I would used Snortsnarf. I followed the install procedures I found on www.Silicondefense.com <http://www.Silicondefense.com> for Windows/Snort/Apache/Snarf but I could not get the programs to work. Next I went back again to www.Silicondefense.com and tried to use their procedures to install Acid. I used the windows/snort/mysql/apcache/Acid pdfs but again I was unable to get the

programs to work. So I returned to reading other students practicals on how they processed the logs and discovered that most students were using *nix tools and databases and scripts, most stuff I had never heard of (what is sed, awk, grep, ??). At that point I made a major decision that I would have to learn more on how to use Unix/Linux as well as Perl. This opened a whole new challenge. I immediately starting studying how to use Linux (online materials, friends, and books). To make a long story short I ended up using Redhat 9 with Snort, Mysql, Apache, and Acid. Using a Perl scripts created by Ryan Johnson http://listserv.secport.com/security/index.html to import the data logs into the database. Using Acid, I was able to break down the number of alerts and top destination and source IP's. I was well on my way to sorting through the logs being able to make good sense of it. This worked well for the OOS logs because they were small enough to put all five days worth of logs into ACID. The Alert logs I had to handle slightly different because of the size of the logs. I had to put one day of the alert logs into acid at a time. I would load the first day, perform my analysis then I would have to flush the database and load up the next day (which took about 30 minutes to load each day worth of alert logs) Sometimes the queries would take up to 4 minutes to complete. 1 day worth of Scan logs was too much for acid to handle (especially on a p2 266mhz laptop). So I used some Perl scripts that would breakout both source IP, destination IP, and port with the number of attempts to each. The results were stored in a text file. From there I copied the text to a windows machine and used MS Access and Excel to help organize and sort the IP and port information.

References:

www.google.com

http://www.dshield.org/ipinfo.php

http://www.apnic.net/

http://www.arin.net/whois/index.html

www.whitehats.org

Sourcefire "Real-Time Network Awareness" June 2003
http://www.sourcefire.com/technology/whitepapers/sourcefire_RNA_0603.pdf

Washington DC Snort user group. <http://mccammon.org/snort/>

www.snort.org <http://www.snort.org>

"passive fingerprinting" whitehats.com
<http://www.whitehats.com/library/passive/index.html>

Miller, Toby “passive OS fingerprinting: details and techniques”
<http://www.incidents.org/papers/OSfingerprinting.php>

ports and descriptions
<http://www.neohapsis.com/neolabs/neo-ports/>

incidents mailing list: port 0 detect
<http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00087.html>

incidents mailing list: port 0 detect
<http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00405.html>

hping2 manuel
<http://www.hping.org/manpage.html>

support documentation: installing ACID and SnortSnarf
<http://www.silicondefense.com/support/windows/documentation.php>

Till, David ebook “Learning Perl in 21 days”

Kay, Trevor Linux + Certification Bible Hungry Minds
copyright 2002

<http://www.giac.org/practical/GCIA/Ewen_Fung_GCIA.pdf>

Northcutt, Stephen and Novak, JudyNetwork Intrusion Detection (3rd edition)
2002

Cothers, Tim Implementing Intrusion Detection Systems: A Hands-On Guide for Securing the Network Wiley 2003

Scarborough, Matt “Gnews aboutGnutella
http://www.sans.org/y2k/gnutella.htm

ISS “Gartner Claims IDS is Dead and Validates ISS’ Strategy”
<http://documents.iss.net/gartner_response.pdf>

Gartner Press release june 11 2003
http://www3.gartner.com/5_about/press_releases/pr11june2003c.jsp

Ptacek, Thomas Newsham, Timothy “Insertion,Evasion, and Denial of service: Eluding Network Intrusion Detection”
http://www.snort.org/docs/idspaper/

Marty Rosech: CEO of Sourcefire, Seminar on the Future of IDS
Washington DC 4/30/2003

George Bakos: senior security expert at Dartmouth College's, Sans instructor for intrusion detection. Baltimore Conference April 2003

Cohen, Fred “50 ways to defeat your IDS”
http://www.darksouls.net/archives/ids/50_Ways_to_Defeate_Your_IDS.txt

ISS (Internet Security Systems) “The evolution of intrustion detection technology Aug 29, 2001
http://documents.iss.net/whitepapers/TheEvolutionofIntrusionDetectionTechnology.pdf

Graham, Robert “Network Intrusion Detection Systems”
http://www.robertgraham.com/pubs/network-intrusion-detection.html

Wu, Marcus GCIA practical
http://www.giac.org/practical/GCIA/Marcus_Wu_GCIA.pdf

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

Cahoon, Brian GCIA practical
http://www.giac.org/practical/GCIA/Brian_Cahoon_GCIA.pdf

Ray, Edward GCIA practical
http://www.giac.org/practical/GCIA/Edward_Ray_GCIA.pdf

Kite, Doug GCIA practical
http://www.giac.org/practical/GCIA/Doug_Kite_GCIA.pdf

Smith, Ricky GCIA practical
http://www.giac.org/practical/GCIA/Ricky_Smith_GCIA.pdf

Acid program
http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html

Shinder, Thomas “You need to create a Split DNS” Sept 19 2002
http://www.isaserver.org/tutorials/You_Need_to_Create_a_Split_DNS.html

Shadow IDS
http://www.nswc.navy.mil/ISSEC/CID/

Al-Jazeera hacked again , 3-30-03
http://www.rense.com/general36/haxkced.htm

Bizreport “American hacker pleads guilty in Al-Jazeera Hacking”
http://www.bizreport.com/article.php?art_id=4503&PHPSESSID=3030485d0c97adf0deba22b398e85aee

Mitchell, Bradley port 0
http://compnetworking.about.com/library/ports/blports_0.htm

Port 0 normal usage
http://www.securiteam.com/securityreviews/5XP0Q2AAKS.html

Linux IPID 0 with DF flag set (both URLs below)
http://marc.theaimsgroup.com/?1=bugtraq&m=98992536801625&w=2
http://marc.theaimsgroup.com/?1=bugtraq&m=101709117512191&w=2

Thompson, Jason port 0 detect
http://cert.uni-stuttgart.de/archive/intrusions/2003/07/msg00183.html

Infosecurity P2P, OR NOT P2P? Feb 2001
http://infosecuritymag.techtarget.com/articles/february01/cover.shtml

CVE, Queso OS fingerprint
http://www.digitaltrust.it/arachnids/IDS29/event.html

Alarcon, Stephanie (GCIH pratical)
www.giac.org/practical/GCIH/stephanie_alarcon_GCIH.pdf

Smashing The Stack For Fun And Profit, Aleph One
http://www.phrack.org/show.php?p=49&a=14

Symantec, Promail Trojan
http://www.symantec.com/avcenter/venc/data/promail.trojan.html

Cisco Ipv4 DOS vulnerability July 2003
http://www.cisco.com/warp/public/707/cisco-sa-20030717-blocked.shtml

TCP cookies
http://cr.yp.to/syncookies.html

Edonkey
http://www.edonkey2000.com/faq.html

Honeynet project, Null packets
http://honeynet.hackers.nl/scans/scan23/sol/Vivek.html

IRC XDCC running
http://www.security.duke.edu/cleaning/xdcc.html

Slyck New, RIAA sues 12 year old girl
http://www.slyck.com/news.php?story=234

Miller, Toby ECN and its impact on IDS, Sept 2000
http://www.securityfocus.com/infocus/1205

Senderbase, email management company
http://www.senderbase.org/?page=help

Perl scripts used in third assignment, Ryan Johnson
http://listserv.secport.com/security/index.html