Spoofed Traffic & Distributed DoS Attack

GIAC Certified Intrusion Analyst (GCIA)
Practical Assignment
Version 3.3 (revised August 19, 2002)

Mohammad Al-Kurbi

December 7, 2003

Abstract/Summary

Welcome to my Practical Assignment in GIAC Certified Intrusion Analyst (GCIA) version 3.3. This Practical is divided into three (3) parts:

- Part-1: that includes an "IP Spoofing in DoS attack" article which talks about protection and trace back methods.
- Part-2: that includes a three different network detects selected by me, with full and detailed analysis information.
- Part-3: that includes a scenario based security audit for a University, given a three different log types.

I tried to organise topics as I could. Expression was reformed many times, to make it as simple and accurate as possible. I referred to as much references as I could, and tried to cite them following the right format as best as I can. If needed or necessary, I listed the references at the end of each separate topic/section/part. Thank you for your time to read my practical, and wish all of us a good luck.

Part 1. Describe the State of Intrusion Detection – IP Spoofing in DoS Attack

Introduction

Denial of Service (DoS) plays a major part of today attack techniques, which add more pressure to stop or at least eliminate it. In presence of IP spoofing, Predicting and denying DoS becomes more and more complicated. In this assignment, I am going to focus on methods that help to protect your network against IP spoofing, and how to trace back such attack to its true source, followed by suggestions to solve this issue forever. For knowledge demonstration, I used the Cisco router as a sample to implement the techniques, but the important is the concept, which can be further transferred/converted to any router vendor format.

Denial of Service Attack

Definition

Denial of Service (DoS) attack is any kind of traffic targeting a host/network in purpose to take it out of service. DoS attacks accomplish this by exhausting the limited resources of network bandwidth by packet flooding or exhausting host resources by consumption of CPU cycles, random memory, and static memory or data structures [1].

Not all service outages, even those that result from malicious activity, are necessarily denial-of-service attacks. Other types of attack may include a denial of service as a component, but the denial of service may be part of a larger attack [2].

There are vary kinds of DoS attacks, as well as levels of complexity, but the last, and most difficult one is to have a Distributed DoS (DDoS) attack using spoofed IP.

The followings are some DoS Techniques that would be mentioned in this document:

- ☐ Ping to Death: is to send so many large ICMP packets to a victim. The intention is to consume the bandwidth, or to trouble the host when receiving these large numbers and sizes of ICMP packets.
- ☐ Smurf attack: is a special kind of the previous Technique, where the victim IP address is spoofed and used as a source IP and the destination is a network broadcast. We call this intermediate network an amplifier or a reflector if it accepts to process and deliver “directed broadcast” packets. All hosts that receive that broadcast ping will reply to the spoofed IP (i.e. the Victim host).
- ☐ Fraggle Attack: is the UDP version of the “Smurf” attack.
- ☐ TCP SYN-Flood: is to send a lot of SYN packets pretending to initiate a TCP connection, but never complete the handshaking process. Your intention is to fill the victim host TCP stack, which will deny that host from accepting the other real connections.

Distributed Denial of Service Attack

In DoS, the attacker launches the attack on the final target network/host, through intermediate compromised hosts. Figure-1 illustrates a Distributed DoS scenario where there could be more than one layer of compromised hosts. First level –which is called the MASTER-, is used to place and trigger the DoS tool in the Second level –which is called the SLAVE-. The SLAVE layer launches the attack on the final victim network/host.

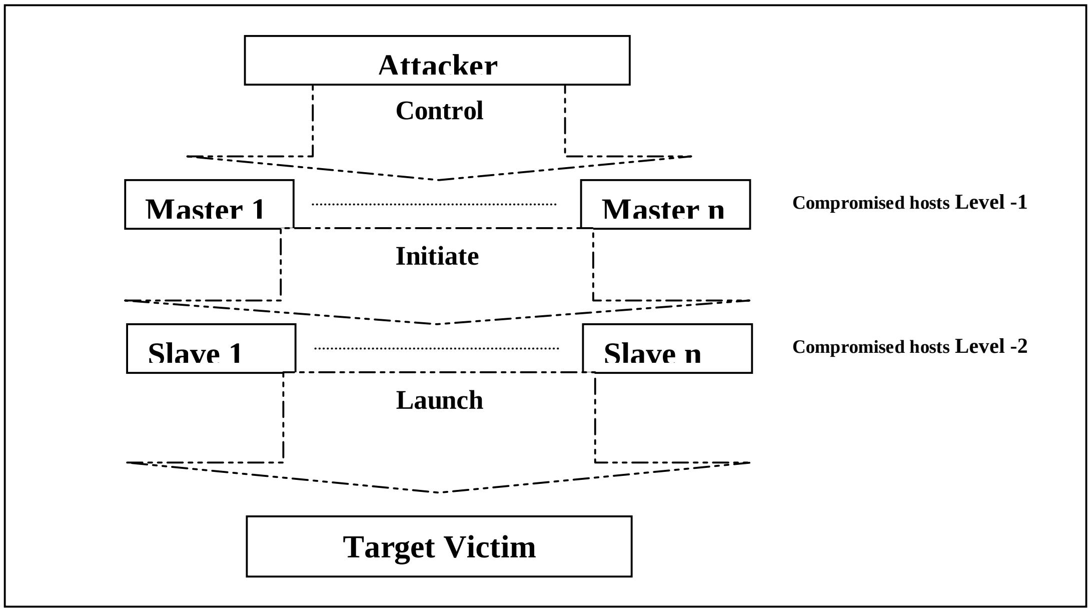


Figure-1 : Sample of DDoS Scenario

DDoS is able to generate a very huge amount of traffic, and is used for many reasons: to protect/hide the real attack source, add more power to it, and complicate the investigation process. It was not easy to control DoS 100%, but now with such complexity, it becomes more difficult. It is a true headache.

▪ How do you detect such Attack?!

Detection of DoS attack depends on its impact and information availability. If you are under a huge attack, it will be so clear to feel the network slowness, where you will receive a lot of complains about unavailable service or service slowness. Nevertheless, it is required to have monitoring tools.

There are punches of monitoring tools that provide any security and network administrator with –so important- updated network statistics information. Examples of these tools are as following:

- IDS Systems: Are used to monitor the network, visualize the traffic, and alarm you when needed. Many IDS systems are available; commercially such as: ISS Real Secure® Network Gigabit …etc), or freely as snort.
- Cricket / HP Open View: Both tools use SNMP to provide Visual graphs that can be customized and configured to keep track of Links load, CPU load and other System resources information. For further information, refer to the following two web sites [http://cricket.sourceforge.net/], [http://www.openview.hp.com/]
- Sniffer: TCPDUMP and SNORT are sniffers, and can be used to figure out any attack destination, as well as to have more information about the attack type, and application layer payloads.
- Net Flow: is an application that can collect exported data from a Net Flow agent or feature that is integrated in the Cisco IOS Router. It collect useful information about a traffic, such as : source/destination address, source/destination ports, Type of service (ToS), Packet and byte counts, timestamps, Input/output interface, TCP flags, Routing information (next-hop address, source autonomous system (AS) number, destination AS number, source prefix mask, destination prefix mask) [22]. Net Flow can be helpful in any attack by figuring out the target victims, the source attack, and other info.
- Router ACL: Can help in different ways. If there is an attack, Router ACL gives an indication by showing hits amount on an access list entry. "log" option helps in finding the source/destination of the attack. "Log-input" option tells from which router interface that traffic comes from.

IP Spoofing

Each packet has a Source and Destination IP. To route a packet, you need the destination address. Source IP is not used, until the destination host respond to that packet. When a packet carries a fake IP (IP of another host, or a private one) as a source, we call it then a spoofed packet.

IP spoofing might lead to unauthorized access to systems or networks, such as intercepting a connection. To do that Attacker has to disable any return traffic to that trusted source, redirect it to his machine, and resume the connection successfully with the destination host. There are many ways of doing that, but one way is to keep the trusted source silent via DoS attack against it, and then play its part with the destination host.

IP spoofing might be used as a part of DoS attack, where the attacker does not need to get any responses from the victim system. It is called "Blind Spoofing", because the attacker works blindly, and he will not get immediate feed back.

Techniques to Protect the Network

It is impossible to protect a network for 100%. This is not a secret to tell, it is the truth, but you have to secure as much as you can, and eliminate the damage as possible. Regular (not spoofed) DoS techniques –like other attacks- are unable to control. To stop such attacks, you need a quick response as soon as an event is discovered from your Incident Response Team (IRT). You most likely need cooperation from your upstream internet provider, to block any traffic coming from a specific source, or going to a specific destination.

Since our concern is on spoofed DoS attack, we are going to walk through and show how to prevent most resources that such attack relies on. Techniques have been classified into. Our theory is based on closing every possible door one by one.

I used Cisco router for demonstration, and I am assuming that there is an explicit deny/permit all at the end of each router Access List (ACL), so be careful.

▪ Anti Spoofing Techniques

I list here those techniques that rely on blocking any spoofed traffic from passing through. As a result any attack that relies on spoofed packets will be blocked as well.

Source Routing

Source routing means that the packet will follow the same path that it came with, even if the source IP requires different route path. Attacker might use that to instruct a router to return a spoofed IP packet in the same path it came through, and so it must be disabled. To disable the source routing on a router, type the following in the global configuration mode:

```
Router (config) #no ip source route
```

IP Directed Broadcast

Packets that directed to broadcast address from outside network are called "directed broadcast" packets, and are currently disabled by default in Cisco IOS version 12.0 and further. Disabling this feature is recommended by "RFC 2644", and will save your network from being a smurf/fraggle amplifier/reflector. To disable this feature you have to access the configuration for each router interface, and specify the following command:

```
Interface XY
        no ip directed-broadcast
```

Special Networks

According to RFC1918 [3] and RFC3330 [23], there are different kinds of network blocks that have been reserved for special purposes, and should not cross over your network boarders. Even though that routing issues have been already resolved, or should be, you need to assure that these networks would not pass by your network in either way to/from external networks. On the following, I list most of them:

- <u>Private Networks:</u> We have different network blocks reserved for private internets: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16.

- <u>Special Use Networks:</u> 0.0.0.0/8 ("This" Network), 127.0.0.0/8 (Loop Back), 169.254.0.0/16 (Link Local), 192.0.2.0/24 (Test-Net), 192.88.99.0/24 (6to4 Relay any cast), 198.18.0.0/15 (Network Interconnect Device Benchmark Testing), 224.0.0.0/4 (Multicast), 255.255.255.255/32 (Broad cast).
- <u>Reserved but subject to allocation:</u> Some of network blocks in this category were reserved –before- for special use, but they returned now returned to the pool of addresses reserved for future allocation or assignment as the other some. This category is part of the address pool, but not yet allocated or used, which means that you should not receive –yet- any traffic from it, so it could be treated as the previous two on the sense of filtering, but requires more knowledge and updated information. It needs more attention. The following are sample of these networks: 24.0.0.0/8 (was for Cable Television Networks), 39.0.0.0/8, 128.0.0.0/16, 191.255.0.0/16, 192.0.0.0/24, 223.255.255.0/24, 240.0.0.0/4, 248.0.0.0/5.

Any coming in/going out traffic moves using one of above network blocks is illegal, and should be filtered at all incoming and outgoing router interfaces that connect your network to upstream internet providers, and your customers. The following two router access lists ("101" incoming ACL and "102" outgoing ACL) are configured to deny such traffic:

```
interface XY
       ip access-group 101 in
       ip access-group 102 out
!
access-list 101 deny ip 0.0.0.0          0.255.255.255     any
access-list 101 deny ip 10.0.0.0         0.255.255.255     any
access-list 101 deny ip 127.0.0.0        0.255.255.255     any
access-list 101 deny ip 169.254.0.0      0.0.255.255       any
access-list 101 deny ip 172.16.0.0       0.15.255.255      any
access-list 101 deny ip 192.0.2.0        0.0.0.255         any
access-list 101 deny ip 198.18.0.0       0.1.255.255       any
access-list 101 deny ip 198.88.99.0      0.0.0.255         any
access-list 101 deny ip 192.168.0.0      0.0.255.255       any
access-list 101 deny ip 224.0.0.0        15.255.255.255    any
access-list 101 deny ip 255.255.255.255  0.0.0.0           any
!
access-list 102 deny ip any  0.0.0.0         0.255.255.255
access-list 102 deny ip any  10.0.0.0         0.255.255.255
access-list 102 deny ip any  127.0.0.0        0.255.255.255
access-list 102 deny ip any  169.254.0.0      0.0.255.255
access-list 102 deny ip any  172.16.0.0       0.15.255.255
access-list 102 deny ip any  192.0.2.0        0.0.0.255
access-list 102 deny ip any  198.18.0.0       0.1.255.255
access-list 102 deny ip any 198.88.99.0       0.0.0.255
access-list 102 deny ip any 192.168.0.0      0.0.255.255
access-list 102 deny ip any  224.0.0.0        15.255.255.255
access-list 102 deny ip any  255.255.255.255  0.0.0.0
```

Home Network

Your Home Networks are your official registered networks, and you are identified by them. RFC 2827 tells us no packet should leave a network if the source address doesn't belong to its address space [6]. Regular Traffic that you should pass it through your network is the one which destined to you, or to one of customer's networks, and also the one that is generated by you, or by your customer. That means the following:

- For in coming traffic, the destination address must be within your or customer's registered address space.
- For out going traffic, the source address must be from your or customer's registered address space.

Any thing else is not legal, and must not pass. The following router access lists configuration shows how to stop such illegal traffic:

- For each network you or your customer has, write the following, followed by explicit deny else at the end of the access-list. Assign that access-list to each outgoing router interface:

```
access-list 102 permit ip Home-Network          Net-Mask  any
access-list 102 permit ip Customer-Network      Net-Mask  any
…
access-list 102 deny ip any any
!
interface XY
         ip access-group 102 out
```

- For each network you or your customer has, write the following, followed by explicit deny else at the end of the access-list. Assign that access-list to each incoming router interface, :

```
! Refuse spoofed packet
access-list 101 deny ip Home-Network            Net-Mask  any
access-list 101 deny ip Customer-Network        Net-Mask  any
! Accept only legal traffic
access-list 101 permit ip any Home-Network      Net-Mask
access-list 101 permit ip any Customer-Network  Net-Mask
!
access-list 101 deny ip any any
!
interface XY
          ip access-group 101 in
```

Unicast Reverse Path Forwarding (Unicast RPF)

Dynamic routing protocols such as (BGP) advertise and receive networks to or from neighbors, which allow Routers that use those protocols to learn routes. A router receives advertisements through a specific interface. If you get a packet through an interface, but you learned about its source IP network from another one, this means you have a spoofed IP. Unicast Reverse Path Forwarding (Unicast RPF) feature helps to do that, and to enable it you have to enable first the "CEF switching" or "CEF distributed switching" mechanism on global configuration mode. Cisco Express Forwarding (CEF) is a switching technique designed to route packets faster [11]. By enabling URPF, the router checks the packet and forwards it if the source IP address has a route in the CEF tables that points back to the same interface on which the packet arrived, or drop it otherwise. The following command shows how to use CEF routing table to check reverse path:

```
! Enable CEF or distributed CEF
ip cef distributed
! Enables Unicast RPF on the interface
interface XY
        ip verify unicast reverse-path
```

Monitoring Techniques

The following are techniques that help in monitoring the traffic and interrupt or intercept it to keep the required services safe.

Packets Rate Limit

ICMP packets and SYN flood are used to create vary kinds of DoS attack. One of the methods that help to prevent DoS attacks is measuring the unusual traffic load, and put it as a measurement to draw a line between the usual and unusual load. Any ICMP/SYN traffics exceed that rate will be dropped. You have to be careful as much as you can to avoid dropping a legitimate traffic. To configure the router you may define one ACL for each service/connection you want to protect, and then use the rate-limit command in interface configuration mode to monitor that ACL on that interface:

```
access-list 102 permit tcp any host eq www
!
Interface XY
          rate-limit input rate-limit access-group 102 20000000 24000 32000  conform-action transmit
exceed-action drop
```

Where “20000000” is average rate, in bits per second (bps). “24000” is a normal burst = configured rate * (1 byte)/(8 bits) * 1.5 seconds. “32000” is extended burst = 2 * normal burst. [14]

TCP Intercept

This Cisco IOS security feature helps in protecting servers from TCP SYN-Flood attack, and needs to be enabled first. Any connections that match an access-list are intercepted in Default TCP Intercept mode. Router stands transparently in the middle between the client and the server. Two connections will be established. First one is between the client and the router, to validate the connection. Once we verify that, the router will establish the second connection with the server, then passes through the traffic between the two connections. To configure the router do the following in global configuration mode:

```
! Access list 102 entries go here
access-list 102 …
…
! Enable TCP Intercept
ip tcp intercept list 102
```

How to trace back any spoofed traffic?!

Since there is no relation between the forged source IP and the real attacker, then the only reliable way to identify the source of an attack is to trace it back hop-by-hop through the networks. This process requires cooperation between all network operators along the path from the victim to the attacker network. Tracing process should be done while the attack is going on.

To trace back a source, we need first to know the boarder gateway interface that such traffic come through. There are many ways to do that such as: Router ACL, Net Flow and Source Tracker.

Router ACL

In a Router Access List, you can use "log-input" option on an ACL entry, to get the source interface that generates the logs:

```
access-list 101  permit/deny  ip   any   host   IP-Address  log-input
```

Net Flow

With Net Flow, you have so many options, but you have to enable it first. By specifying the victim address as an example, you can show detailed information about packets, and source interface will be among them:

```
show ip cache flow | include VICTIM
```

Source Tracker

It is a feature that allows you to gather information about a traffic flowing to a host [21]. You have to enable it first - for a while- to track a flow. To track a flow to a victim address, configure the router as following in the global configuration mode:

```
! enable tracking for the destination address
Router(config)#ip source-track VICTIM
!
show ip source-track [VICTIM|summary]
```

Once you found out the source boarder gateway interface that attack come through, you move to the next step, and find out who is the next hop for that source interface –It should be strait forward-. Then, you have to contact its responsible network administrator, to carry on, and continue doing the same procedure, until you reach the real attack source network. Without the cooperation at each hop, you will not succeed.

Suggestions

To stop spoofed DOS attack, a global support, and cooperative works are needed between network operators all over the world. If we succeed on this mission, then there will be no hide place for any bad behave. I would recommend the following points to help in stopping any Spoofed Traffic:

- Global Security Standard/Policy should exists, and includes current and further Security recommendations such as what we explore in "Techniques to Protect the Network" topic in this document , and then network operators must followe it.
- A Global Community should exist, as similar as "anti-spamming" and "anti-smurf-amplifier" communities/groups, or bigger than them. This group/community would track any network that does not follow the Security Standard/Policy. Report it, and follow it up as much power as it can be. It is preferred to have a power to disconnect that network functionally –Could be a Temp situation- from the internet.
- Law enforcement agencies have o be strongly involved.
- Alternatively, if all or part of previous is hard to implement, or need time, so then I would like to suggest including new and brief Information about the REAL source network as an option on the IP packet header. To avoid fake information, this option could be added by the ISPs boarder routers that connect them to other networks. Such information –such as: Source Autonomous System (AS) - if included, will simply provide us with attack source network.

References (For Part1)

[1] Brindley, Adrian. "Denial of Service Attacks and the Emergence of "Intrusion Prevention Systems"". November 1, 2002. URL: http://www.sans.org/rr/firewall/prevention.php

[2] CERT® Coordination Center. "Denial of Service Attacks". June 4, 2001. URL: http://www.cert.org/tech_tips/denial_of_service.html

[3] Rekhter Yakov, Moskowitz Robert, Karrenberg Daniel, Groot Geert, Lear Eliot. "Address Allocation for Private Internets". RFC 1918. February 1996. URL: http://www.ietf.org/rfc/rfc1918.txt

[4] Troll, Ryan. "Internet Draft: DHC-IPV4-AUTOCONFIG". October 1998. URL: http://www.ietf.org/proceedings/98dec/I-D/draft-ietf-dhc-ipv4-autoconfig-01.txt

[5] Bradner S., McQuaid J. "Benchmarking Methodology for Network Interconnect Devices". RFC 2544. March 1999. URL: http://www.ietf.org/rfc/rfc2544.txt

[6] Lenssen, Klaus. "Security on Routers", Networkers Security Conference (2002): Session SEC-211.

[7] Rice, Russel. "Network Security: Design and Attack Mitigation". Networkers Security Conference (2002): Session SEC-201.

[8] Todd, Bennett. "Distributed Denial of Service Attacks". 18 February 2000. URL: http://www.opensourcefirewall.com/ddos_whitepaper_copy.html

[9] "Strategies to Protect Against Distributed Denial of Service (DDoS) Attacks". Cisco White Papers: Document ID: 13634 .Apr 29, 2003. URL: http://www.cisco.com/warp/public/707/newsflash.html

[10] Huegen, Craig A. "THE LATEST IN DENIAL OF SERVICE ATTACKS: "SMURFING" DESCRIPTION AND INFORMATION TO MINIMIZE EFFECTS". Feb 8 2000. URL: http://www.pentics.net/denial-of-service/white-papers/smurf.txt

[11] "How-To Troubleshoot Unicast IP Routing CEF on Catalyst 6000s with a Supervisor 2 in Hybrid Mode". Cisco IP switching: Document ID: 20626. May 29, 2002. URL**:** **http://www.cisco.com/en/US/tech/tk827/tk831/technologies_tech_note09186a0080094b27.shtml**

[12] Behringer, Michael. "Surviving a DoS Attack". Networkers Security Conference (2002): Session SEC-301.

[13] Cisco Systems. "Cisco ISP Essentials". Version 2.9. June 06, 2001. URL: http://www.cisco.com/public/cons/isp/documents/IOSEssentialsPDF.zip

[14] Cisco Systems. "Quality of Service Commands". Jan 17, 2003. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_r/qrcmdr.htm

[15] SANS Institue. "Help Defeat Denial of Service Attacks: Step-by-Step". Revision: 1.4- March 23, 2000. URL: http://www.sans.org/dosstep/index.php

[16] "Characterizing and Tracing Packet Floods Using Cisco Routers". Cisco Tech Notes: Document ID: 13609. Feb 20, 2003. URL: http://www.cisco.com/warp/public/707/22.html

[17] "Configuring TCP Intercept (Preventing Denial-of-Service Attacks)". Cisco IOS Security Configuration Guide. May 3, 2002. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fsecur_c/ftrafwl/scfdenl.htm

[18] Senie Daniel. "Changing the Default for Directed Broadcasts in Routers". RFC 2644. August 1999. URL: http://www.ietf.org/rfc/rfc2644.txt

[19] "Quality of Service Commands Q through R". Cisco IOS Quality of Service Solutions Command Reference, Release 12.3. Jun 23, 2003. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios123/123cgcr/qos_r/qos_q1g.htm

[20] "Configuring Unicast Reverse Path Forwarding". Cisco IOS Security Configuration Guide, Release 12.2. May 3, 2002. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fsecur_c/fothersf/scfrpf.htm

[21]" IP Source Tracker". Cisco New Features in Release 12.0(21)S. May 13, 2003. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s21/ipst.htm

[22] “Cisco IOS NetFlow *Technology”. Cisco* Data *Sheet. Jul 15, 2002. URL:*
http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/iosnf_ds.htm

[23] IANA. “Special-Use IPv4 Addresses”. RFC 3330. September 2002. URL: http://www.ietf.org/rfc/rfc3330.txt

Part 2 – Network Detects

Detect (1) – W32/Blaster worm | RPC DCOM BUFFER OVERFLOW

- **Snort Logs:**

Part (a) : Launch Attack to port (135)

```
08/14-10:19:51.117583 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3050 -> 172.16.117.148:135 TCP TTL:124 TOS:0xA0 ID:31398 IpLen:20
DgmLen:48 DF
******S* Seq: 0x9409397  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:19:52.919740 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3051 -> 172.16.117.149:135 TCP TTL:124 TOS:0xA0 ID:31405 IpLen:20
DgmLen:48 DF
******S* Seq: 0x9484F2D  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:19:56.547861 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3101 -> 172.16.117.194:135 TCP TTL:124 TOS:0xA0 ID:31460 IpLen:20
DgmLen:48 DF
******S* Seq: 0x977C6B4  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:19:58.365416 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3136 -> 172.16.117.228:135 TCP TTL:124 TOS:0xA0 ID:31502 IpLen:20
DgmLen:48 DF
******S* Seq: 0x9970F32  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:00.174372 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3146 -> 172.16.117.236:135 TCP TTL:124 TOS:0xA0 ID:31518 IpLen:20
DgmLen:48 DF
******S* Seq: 0x9A38EB6  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:14.677153 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3323 -> 172.16.118.144:135 TCP TTL:124 TOS:0xA0 ID:31690 IpLen:20
DgmLen:48 DF
******S* Seq: 0xA55782F  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:20.111692 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3386 -> 172.16.118.200:135 TCP TTL:124 TOS:0xA0 ID:31746 IpLen:20
DgmLen:48 DF
```

```
******S* Seq: 0xA91FDF4  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:20.112216 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3387 -> 172.16.118.201:135 TCP TTL:124 TOS:0xA0 ID:31747 IpLen:20
DgmLen:48 DF
******S* Seq: 0xA92E89D  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:21.927805 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3415 -> 172.16.118.229:135 TCP TTL:124 TOS:0xA0 ID:31775 IpLen:20
DgmLen:48 DF
******S* Seq: 0xAAF8491  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:25.548274 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3444 -> 172.16.119.0:135 TCP TTL:124 TOS:0xA0 ID:31801 IpLen:20
DgmLen:48 DF
******S* Seq: 0xAD0BC12  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:25.548362 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3445 -> 172.16.119.1:135 TCP TTL:124 TOS:0xA0 ID:31802 IpLen:20
DgmLen:48 DF
******S* Seq: 0xAD15A51  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:20:25.549564 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3446 -> 172.16.119.2:135 TCP TTL:124 TOS:0xA0 ID:31803 IpLen:20
DgmLen:48 DF
******S* Seq: 0xAD22D23  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:21:06.075218 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3676 -> 172.16.119.209:135 TCP TTL:124 TOS:0xA0 ID:32137 IpLen:20
DgmLen:48 DF
******S* Seq: 0xC0D7F53  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:21:06.076464 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3680 -> 172.16.119.212:135 TCP TTL:124 TOS:0xA0 ID:32140 IpLen:20
DgmLen:48 DF
******S* Seq: 0xC1021B6  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:21:15.146422 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3779 -> 172.16.120.44:135 TCP TTL:124 TOS:0xA0 ID:32238 IpLen:20
DgmLen:48 DF
******S* Seq: 0xC74CF30  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

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

08/14-10:21:16.950300 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3807 -> 172.16.120.70:135 TCP TTL:124 TOS:0xA0 ID:32264 IpLen:20
DgmLen:48 DF
******S* Seq: 0xC8EF076  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:25:12.277106 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:1547 -> 172.16.126.79:135 TCP TTL:124 TOS:0xA0 ID:34707 IpLen:20
DgmLen:48 DF
******S* Seq: 0x14B42C2C  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:25:14.080455 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x1B2
192.168.194.40:1540 -> 172.16.126.72:135 TCP TTL:124 TOS:0xA0 ID:34741 IpLen:20
DgmLen:420 DF
***A**** Seq: 0x14AED579  Ack: 0xBAFCD9CC  Win: 0x4150  TcpLen: 20
05 00 00 03 10 00 00 00 A8 06 00 00 E5 00 00 00  ................
90 06 00 00 01 00 04 00 05 00 06 00 01 00 00 00  ................
00 00 00 00 32 24 58 FD CC 45 64 49 B0 70 DD AE  ....2$X..EdI.p..
74 2C 96 D2 60 5E 0D 00 01 00 00 00 00 00 00 00  t,..`^..........
70 5E 0D 00 02 00 00 00 7C 5E 0D 00 00 00 00 00  p^......|^......
10 00 00 00 80 96 F1 F1 2A 4D CE 11 A6 6A 00 20  ........*M...j. 
AF 6E 72 F4 0C 00 00 00 4D 41 52 42 01 00 00 00  .nr.....MARB....
00 00 00 00 0D F0 AD BA 00 00 00 00 A8 F4 0B 00  ................
20 06 00 00 20 06 00 00 4D 45 4F 57 04 00 00 00   ... ...MEOW....
A2 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  ...............F
38 03 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  8..............F
00 00 00 00 F0 05 00 00 E8 05 00 00 00 00 00 00  ................
01 10 08 00 CC CC CC CC C8 00 00 00 4D 45 4F 57  ............MEOW
E8 05 00 00 D8 00 00 00 00 00 00 00 02 00 00 00  ................
07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 C4 28 CD 00 64 29 CD 00 00 00 00 00  .....(..d)......
07 00 00 00 B9 01 00 00 00 00 00 00 C0 00 00 00  ................
00 00 00 46 AB 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 A5 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 A6 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 A4 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 AD 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 AA 01 00 00 00 00 00 00 C0 00 00 00  ...F............
00 00 00 46 07 00 00 00 60 00 00 00              ...F....`...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:39:16.797894 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:3170 -> 172.16.161.1:135 TCP TTL:124 TOS:0xA0 ID:43833 IpLen:20
DgmLen:48 DF
******S* Seq: 0x3B31ADDB  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:39:54.111755 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x7E
192.168.194.40:3507 -> 172.16.162.54:135 TCP TTL:124 TOS:0xA0 ID:44218 IpLen:20
DgmLen:112 DF
***AP*** Seq: 0x3CA401C1  Ack: 0x7A6256DF  Win: 0x4470  TcpLen: 20
05 00 0B 03 10 00 00 00 48 00 00 00 7F 00 00 00  ........H.......
D0 16 D0 16 00 00 00 00 01 00 00 00 01 00 01 00  ................
A0 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46  ...............F
00 00 00 00 04 5D 88 8A EB 1C C9 11 9F E8 08 00  .....]..........
```

```
2B 10 48 60 02 00 00 00                          +.H`....

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

08/14-10:39:54.129482 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x12A
192.168.194.40:3507 -> 172.16.162.54:135 TCP TTL:124 TOS:0xA0 ID:44220 IpLen:20
DgmLen:284 DF
***AP*** Seq: 0x3CA407BD  Ack: 0x7A6256DF  Win: 0x4470  TcpLen: 20
93 CD C2 94 EA 64 F0 21 8F 32 94 80 3A F2 EC 8C  .....d.!.2..:...
34 72 98 0B CF 2E 39 0B D7 3A 7F 89 34 72 A0 0B  4r....9..:..4r..
17 8A 94 80 BF B9 51 DE E2 F0 90 80 EC 67 C2 D7  ......Q......g..
34 5E B0 98 34 77 A8 0B EB 37 EC 83 6A B9 DE 98  4^..4w...7..j...
34 68 B4 83 62 D1 A6 C9 34 06 1F 83 4A 01 6B 7C  4h..b...4...J.k|
8C F2 38 BA 7B 46 93 41 70 3F 97 78 54 C0 AF FC  ..8.{F.Ap?.xT...
9B 26 E1 61 34 68 B0 83 62 54 1F 8C F4 B9 CE 9C  .&.a4h..bT......
BC EF 1F 84 34 31 51 6B BD 01 54 0B 6A 6D CA DD  ....41Qk..T.jm..
E4 F0 90 80 2F A2 04 00 5C 00 43 00 24 00 5C 00  ..../...\.C.$.\.
31 00 32 00 33 00 34 00 35 00 36 00 31 00 31 00  1.2.3.4.5.6.1.1.
31 00 31 00 31 00 31 00 31 00 31 00 31 00 31 00  1.1.1.1.1.1.1.1.
31 00 31 00 31 00 31 00 31 00 2E 00 64 00 6F 00  1.1.1.1.1...d.o.
63 00 00 00 01 10 08 00 CC CC CC CC 20 00 00 00  c....... .......
30 00 2D 00 00 00 00 00 88 2A 0C 00 02 00 00 00  0.-......*......
01 00 00 00 28 8C 0C 00 01 00 00 00 07 00 00 00  ....(...........
00 00 00 00                                      ....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:47:56.091845 0:7:D:F:73:FC -> 0:7:D:F:AF:FC type:0x800 len:0x3E
192.168.194.40:2814 -> 172.16.174.2:135 TCP TTL:124 TOS:0xA0 ID:47864 IpLen:20
DgmLen:48 DF
******S* Seq: 0x4CB706D5  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

Part (b): Hosts Reply

```
08/14-10:19:51.245671 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x3E
172.16.117.148:135 -> 192.168.194.40:3050 TCP TTL:104 TOS:0xA0 ID:31398 IpLen:20
DgmLen:48 DF
***A*R** Seq: 0x0  Ack: 0x9409398  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:19:53.726742 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x3E
172.16.117.149:135 -> 192.168.194.40:3051 TCP TTL:104 TOS:0xA0 ID:31427 IpLen:20
DgmLen:48 DF
***A*R** Seq: 0x0  Ack: 0x9484F2E  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

08/14-10:19:56.674033 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x3C
172.16.117.194:135 -> 192.168.194.40:3101 TCP TTL:245 TOS:0x0 ID:8544 IpLen:20
DgmLen:40
***A*R** Seq: 0x0  Ack: 0x977C6B5  Win: 0x0  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:20:48.590855 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x3C
172.16.119.157:135 -> 192.168.194.40:3620 TCP TTL:116 TOS:0x0 ID:13870 IpLen:20
DgmLen:40 DF
```

```
***A**** Seq: 0xDB85D1EF  Ack: 0xB9C2831  Win: 0x4470  TcpLen: 20

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

08/14-10:21:02.408154 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x5E
172.16.119.158:135 -> 192.168.194.40:3621 TCP TTL:116 TOS:0x0 ID:14073 IpLen:20
DgmLen:80 DF
***AP*** Seq: 0xDB86A511  Ack: 0xB9D1902  Win: 0x4470  TcpLen: 20
05 00 02 03 10 00 00 00 28 00 00 00 E5 00 00 00  ........(.......
10 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 04 00 08 80                          ........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:25:14.205649 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x72
172.16.126.72:135 -> 192.168.194.40:1540 TCP TTL:119 TOS:0x0 ID:45487 IpLen:20
DgmLen:100 DF
***AP*** Seq: 0xBAFCD9CC  Ack: 0x14AED579  Win: 0x4108  TcpLen: 20
05 00 0C 03 10 00 00 00 3C 00 00 00 7F 00 00 00  ........<.......
D0 16 D0 16 4F DC 00 00 04 00 31 33 35 00 00 00  ....O.....135...
01 00 00 00 00 00 00 00 04 5D 88 8A EB 1C C9 11  .........]......
9F E8 08 00 2B 10 48 60 02 00 00 00              ....+.H`....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:39:55.053818 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x3C
172.16.162.54:135 -> 192.168.194.40:3507 TCP TTL:108 TOS:0x0 ID:194 IpLen:20 DgmLen:40
DF
***A***F Seq: 0x7A62571B  Ack: 0x3CA408B2  Win: 0x4470  TcpLen: 20

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

Part (c): ICMP Reply

```
08/14-10:19:49.437290 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.96.17 -> 192.168.194.40 ICMP TTL:245 TOS:0x0 ID:0 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 A0 00 30 7A 88 40 00 01 06 1E 79  ....E..0z.@....y
D4 64 C2 28 D4 23 75 76 0B CA 00 87 09 24 50 FB  .d.(.#uv.....$P.

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

08/14-10:20:29.299467 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x5A
172.16.119.38 -> 192.168.194.40 ICMP TTL:246 TOS:0xC0 ID:44613 IpLen:20 DgmLen:76
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.194.40:3483 -> 172.16.119.38:135 TCP TTL:116 TOS:0xA0 ID:31847 IpLen:20
DgmLen:48 DF
******S* Seq: 0xAFB2FA0  Ack: 0x0  Win: 0x4000  TcpLen: 28
** END OF DUMP
00 00 00 00 45 A0 00 30 7C 67 40 00 74 06 A7 E9  ....E..0|g@.t...
D4 64 C2 28 D4 23 77 26 0D 9B 00 87 0A FB 2F A0  .d.(.#w&....../.
00 00 00 00 70 02 40 00 18 8B 00 00 02 04 05 B4  ....p.@.........
01 01 04 02                                      ....

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

08/14-10:24:36.157769 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.124.180 -> 192.168.194.40 ICMP TTL:53 TOS:0x0 ID:24725 IpLen:20 DgmLen:56
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.194.40:1104 -> 172.16.124.180:135 TCP TTL:115 TOS:0xA0 ID:34238 IpLen:20
DgmLen:48 DF
```

```
Seq: 0x12F460A0  Ack: 0x1010101
** END OF DUMP
00 00 00 00 45 A0 00 30 85 BE 40 00 73 06 9A 04  ....E..0..@.s...
D4 64 C2 28 D4 23 7C B4 04 50 00 87 12 F4 60 A0  .d.(.#|..P....`.

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

08/14-10:24:36.169351 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.124.190 -> 192.168.194.40 ICMP TTL:53 TOS:0x0 ID:24730 IpLen:20 DgmLen:56
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.194.40:1115 -> 172.16.124.190:135 TCP TTL:115 TOS:0xA0 ID:34248 IpLen:20
DgmLen:48 DF
Seq: 0x12FC35CA  Ack: 0x1030300
** END OF DUMP
00 00 00 00 45 A0 00 30 85 C8 40 00 73 06 99 F0  ....E..0..@.s...
D4 64 C2 28 D4 23 7C BE 04 5B 00 87 12 FC 35 CA  .d.(.#|..[....5.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:39:22.111105 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11748 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 6F 40 00 01 06 C2 71  ....E..0.o@....q
D4 64 C2 28 D4 23 A1 36 0C 9A 00 87 3B 6D 6B 5F  .d.(.#.6....;mk_

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

08/14-10:39:22.112034 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11750 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 72 40 00 01 06 C2 6B  ....E..0.r@....k
D4 64 C2 28 D4 23 A1 39 0C 9D 00 87 3B 6F 9A CA  .d.(.#.9....;o..

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

08/14-10:39:22.112034 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11752 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 73 40 00 01 06 C2 69  ....E..0.s@....i
D4 64 C2 28 D4 23 A1 3A 0C 9E 00 87 3B 70 69 20  .d.(.#.:....;pi

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

08/14-10:39:22.112034 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11752 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 73 40 00 01 06 C2 69  ....E..0.s@....i
D4 64 C2 28 D4 23 A1 3A 0C 9E 00 87 3B 70 69 20  .d.(.#.:....;pi

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

08/14-10:39:22.116686 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11759 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 6C 40 00 01 06 C2 77  ....E..0.l@....w
D4 64 C2 28 D4 23 A1 33 0C 97 00 87 3B 6A DC CE  .d.(.#.3....;j..

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

08/14-10:39:22.116686 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11760 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 6B 40 00 01 06 C2 79  ....E..0.k@....y
D4 64 C2 28 D4 23 A1 32 0C 96 00 87 3B 69 DF 27  .d.(.#.2....;i.'
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/14-10:39:22.117161 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.161.1 -> 192.168.194.40 ICMP TTL:236 TOS:0xC0 ID:11761 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 AB 6D 40 00 01 06 C2 75  ....E..0.m@....u
D4 64 C2 28 D4 23 A1 34 0C 98 00 87 3B 6B 8B 0A  .d.(.#.4....;k..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…

08/14-10:43:51.123956 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x5A
172.16.174.2 -> 192.168.194.40 ICMP TTL:232 TOS:0xC0 ID:55452 IpLen:20 DgmLen:76
Type:3  Code:1  DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.194.40:4096 -> 172.16.164.66:135 TCP TTL:102 TOS:0x0 ID:45076 IpLen:20
DgmLen:48
******S* Seq: 0x41B015F8  Ack: 0x0  Win: 0x4000  TcpLen: 28
** END OF DUMP
00 00 00 00 45 00 00 30 B0 14 00 00 66 06 95 C0  ....E..0....f...
D4 64 C2 28 D4 23 A4 42 10 00 00 87 41 B0 15 F8  .d.(.#.B....A...
00 00 00 00 70 02 40 00 CB FC 00 00 02 04 05 B4  ....p.@.........
01 01 04 02                                      ....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
…
…
08/14-10:45:16.068548 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.162.129 -> 192.168.194.40 ICMP TTL:233 TOS:0xC0 ID:41582 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 B3 A1 00 00 01 06 F3 B4  ....E..0........
D4 64 C2 28 D4 23 A7 C1 04 4E 00 87 45 99 6C 46  .d.(.#...N..E.lF

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

08/14-10:45:16.090603 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.162.129 -> 192.168.194.40 ICMP TTL:233 TOS:0xC0 ID:41583 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 B3 A2 00 00 01 06 F3 B2  ....E..0........
D4 64 C2 28 D4 23 A7 C2 04 4F 00 87 45 99 F3 DD  .d.(.#...O..E...

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

08/14-10:45:16.091239 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.162.129 -> 192.168.194.40 ICMP TTL:233 TOS:0xC0 ID:41584 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 B3 A3 00 00 01 06 F3 B0  ....E..0........
D4 64 C2 28 D4 23 A7 C3 04 50 00 87 45 9A E6 84  .d.(.#...P..E...

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

08/14-10:45:16.092247 0:7:D:F:AF:FC -> 0:7:D:F:73:FC type:0x800 len:0x46
172.16.162.129 -> 192.168.194.40 ICMP TTL:233 TOS:0xC0 ID:41588 IpLen:20 DgmLen:56
Type:11  Code:0  TTL EXCEEDED IN TRANSIT
00 00 00 00 45 00 00 30 B3 A6 00 00 01 06 F3 AA  ....E..0........
D4 64 C2 28 D4 23 A7 C6 04 53 00 87 45 9C DC 4B  .d.(.#...S..E..K

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

- **Source of Trace**

This is an ISP network, where you have two routers connected to a switch. By saying two routers, I do not mean only two physical routers, but I mean two classes. First class is the boarders which are connected to Internet providers. Second class connects customers' networks to ISP network. Each class may have more than one router. Sniffer is connected to the switch, and listens to all going in & out traffic. Data has been sanitized. Figure-2 draws a picture of this network topology.

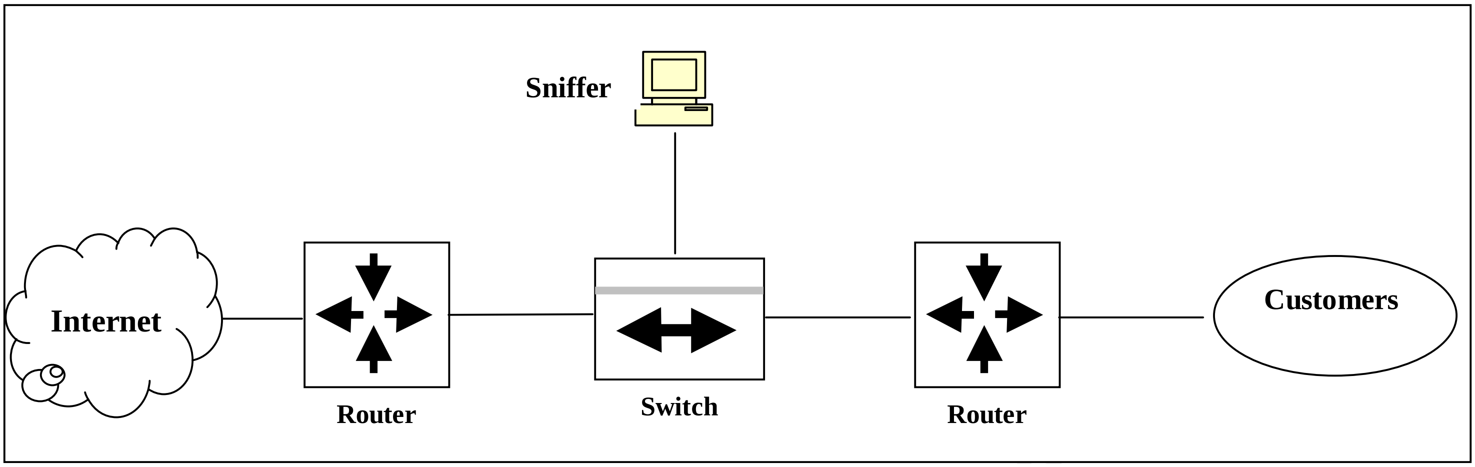


Figure-2 : Network Topology

▪ Detect was generated by

Logs are generated with Snort version (1.9.0), and default rule set, on a Linux box. First I captured the data into binary format (TDCPDUMP Format) using the following command: (snort –b -i eth1 –l logs). I took then the binary file and run it against the Snort standard rule set with the following command:

```
snort –de –c snort.conf –r snort.bin –l logs
```

When I checked the output alert file I saw a huge amount of ICMP alert messages with different types, and codes: ICMP Destination Unreachable (Port Unreachable), ICMP Destination Unreachable (Host Unreachable), ICMP Time-To-Live Exceeded in Transit. I returned back to the binary captured data file, to analyse it using this command:

```
snort –dev –r snort.bin
```

Later on I found in the first place a huge amount of traffic scanning or launching an attack on port (135), so I picked all that traffic, and wrote it to a file using this command:

```
snort –dev –r snort.bin "dst port 135" > attack.log
```

I Combined outputs from the last three commands. I picked up –then- one of the attack sources, and analyse it further with the following commands:

```
# Monitor any traffic from one specific source. Part(a) of the Detects.
snort -dev -r snort.bin "src host 192.168.194.40 and dst net 172.16" > Attack.log

# Monitor all icmp replies from victims. Part(c) of the Detcts.
snort -dev -r snort.bin "dst host 192.168.194.40 and src net 172.16 and icmp" >
Response.icmp

# Do we have  any compromised hosts. Part(b) of the Detects.
snort -dev -r snort.bin "dst host 192.168.194.40 and src net 172.16 and not icmp" >
Host.Response
```

I found that any host from the victim network (172.16.x.x) either reply the connection back to "192.168.194.40", or send one of the following ICMP messages: ICMP Destination Unreachable (Port Unreachable), ICMP Destination Unreachable (Host Unreachable), ICMP Time-To-Live Exceeded in Transit.

Log Format

Log format is in a snort standard format as a Sequence of packets. Each packet consists of:

- Date & Time.
- Data Link layer headers: Source and Destination address, protocol, and packet length.
- For next Two mixed Layers: you need some knowledge of IP Header, ICMP, and Transport Layer Headers (TCP, UDP). We will explain some fields :
    - IP Header fields :
        - TTL: Time to Live.
        - TOS: Type of Service.
        - ID: packet identification.
        - IpLen: IP header Length.
        - DgmLen: IP packet Total Length.
        - DF : Don't Fragment bit is set to one.
    - TCP Header fields :
        - ******S* : SYN bit is set.
        - Seq: Sequence Number.
        - Ack: Acknowledge number.
        - Win: Windows size.
        - TcpLen: TCP header length.
        - TCP options
- Application Layer data in hex and ASCII.

Snort Rule

From "icmp-info.rules" file, I used the following rules:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Destination Unreachable
(Port Unreachable)"; itype: 3; icode: 3; sid:402;  classtype:misc-activity; rev:4;)

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Destination Unreachable
(Host Unreachable)"; itype: 3; icode: 1; sid:399;  classtype:misc-activity; rev:4;)

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Time-To-Live Exceeded in
Transit"; itype: 11; icode: 0; sid:449;  classtype:misc-activity; rev:4;)
```

▪ Probability the source address was spoofed

I think this is not a spoofed IP, but a real one for the following reasons:

- From *Part(a)* on "Snort Logs" section we see one source sends allot, but not a huge traffic towards what it seems to be a part of "B" Class, Which indicate that it is not a DoS attack.
- Any host has been attacked mostly once, which indicate again that it is not a DoS attack.
- From *Part(a*), and according to attack description, and attack mechanism, this should be a part of an attack initiation process that exploit vulnerabilities in the Microsoft Remote Procedure Call (RPC) Interface. It is a TCP connection that needs to be established, before the attack can carry on. Spoofed would not be useful in this case.
- Time-To-Live (TTL) value gives us some indication that each attack source came from different networks, because each one has a different TTL values when picket up by the sensor, when it passed through ISP network. One more thing is that the TTL value is always the same if it came from the same source, and this gives us another indication that the attack source is really what he claimed to be. Such sources are: `212.71.63.124 (TTL=125), 192.168.194.40 (TTL=124).`
- From attack sources we tell the attack was coming from customers' networks. Was it?! Attack hosts are Windows compromised machines –By the worm in the first place-. The current TTL value from sniffed packets are (124/125), which is reasonable if the attack really came from customers networks, since they are 3~4 hop a way, Assuming that Windows default TTL value is (128). That is another indication that attack source IP is not fake.

- Source IP is not a private one, nor unregistered address. By query Ripe data base we have the following Info :

Description of attack

This attack tries to get a full access (TCP connection) to a host and execute arbitrary code by exploiting buffer overflow vulnerability in a Windows Distributed Component Object Model (DCOM) Remote Procedure Call (RPC) interface on port (135). *Part(a)* on "Snort Logs" shows how does a compromised Windows host "192.168.194.40" by a prior worm life cycle infection, initiated an attack to a portion –as shown in *part(a)*- of Class B (172.16.x.x) on port (135). Packets were destined to {172.16.117.148, 172.16.117.149...etc} are sample of initiation process. Packets were destined to {172.16.126.72, 172.16.162.54} are sample of buffer overflow attempts, after establishing the connection. This attack has a Common Vulnerabilities and Exposures (CVE) record under name "CAN-2003-0352 (under review)" [2].

Attack mechanism

1. The machine -infected by the Worm- picks random sequential IP addresses from each "C" class in [0.0.0.0 - 255.255.255.0]. It does a TCP scan for open (135) port on those networks.
2. For those who accept the connection, it sends them a specific malformed crafted request to exploit a buffer overflow.
3. Infected machine has a full access right now on the remote target –compromised- machine. It can execute any code on it.
4. Before installing a copy of the worm, auto run entry is created on the remote target machine Under windows registry key to execute every time windows starts : [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run]. Depends on Worm variants, different entries will be created under the described registry key:
    - **"windows auto update" = MSBLAST.EXE (variant A)**
    - **"windows auto update" = PENIS32.EXE (variant B)**
    - **"Microsoft Inet xp.." = TEEKIDS.EXE (variant C)**
5. Infected machine instruct the remote target machine to download a copy of the worm via TFTP into the Windows System32 folder.
6. Infected machine instruct the remote target machine to run the worm, and another worm life cycle will start, but now via this remote target as the infected machine.
7. On specific days, and months as following [3]:
    - On 16 to the 31 of the following months: January, February, March, April, May, June, July, August.
    - Any day in the month of September to December.

    All infected machines launch a Distributed Denial of Service attack against a specific site such as "windowsupdate.com", or if the worm fails to resolve the web site, it launches the attack against the broad cast address "255.255.255.255". This DDoS attack is carried on by sending around 40 Bytes size SYN packet every 20 milliseconds, with spoofed IP.

Correlations

It is a wide spread and one of the top most attack according to Internet Storm Center [4] at the time of writing this document, and may be it is right now. It uses common buffer overflow vulnerability in Microsoft RPC, which is used by a couple of other viruses/worms that rely on the same vulnerability to launch the attack. In other words we have what we call it "RPC DCOM buffer overflow worm's family", and it is mentioned in most security sites, and I will just list four of them:

- Internet Storm Center web site show us up to date detailed reports about RPC DCOM attacks, URL: http://isc.incidents.org/port_details.html?port=135

- Cert web site announces this vulnerability many times, so Cert briefs it on this URL: [http://www.cert.org/current/current_activity.html#msrpc]. Cert announce some worms, and viruses that use this vulnerability, such as:
    - "W32/Blaster" or "W32/Lovsan", URL: [http://www.cert.org/current/current_activity.html#msrpcworm].
    - " **W32/Welchia Worm**" known alternately as 'W32/Welchia', 'W32/Nachi', or 'WORM_MS_BLAST.D', URL: [http://www.cert.org/current/current_activity.html#welchia]
- CVE recorded this vulnerability under “CAN-2003-0352 (under review)” name/ID [2].
- Trend Micro site [3] lists the following WORM_MSBLAST aliases: W32.Blaster.Worm, W32/Lovsan.worm, W32/Blaster-A, Worm.Win32.Lovesan, WORM_MSBLAST.A, WORM_MSBLAST.B, WORM_MSBLAST.C, and WORM_MSBLAST.D.

Evidence of active targeting

This attack is not destined to a specific network in particular, but it attacks random IPs in sequence networks that could be in this range [0.0.0.0 - 255.255.255.0]. Your network had fallen under the attack because it fitted within the network range that the attack is destined, and *Part (a)* of the detects shows how a class B (172.16.x.x) is under attack, and *"Attack Mechanism"* section explains how the attack works, which proof what we said earlier that This attack is not destined to a specific network in particular.

DDoS part of this attack –in some case- is launched against a particular specific networks, such as *Windows Update* web site, and other.

Severity

Severity is calculated using the following formula:

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

Criticality = 3

According to Attack mechanism, any Windows servers/desktops –within a network- are possible targets, they can be a web server, a mail exchanger, or most probably a PC. The probability to target a Windows desktop is much higher than targeting a Windows server, regardless of its criticality, which is usually low too. System criticality is vary from one target network to another, and the web server that it is not critical for some client, it is indeed very critical for others that have Online web trade. So, I would take the average via this formula: (4+1)/2=2.5, but since the compromised targets can launch another attack in a new life cycle on others, or being a part of a DDoS attack against specific sites, then I choose to rank it as (3).

Lethality = 5

If the attack succeeds, then Attacker gains root access to the target machine, and can run, or do any thing on it, what worse than that?!. So, I would rank it as (5), for its lethality.

System countermeasures = 2

Patch is available, but you can not guarantee that all systems have been patched. The worm hits are still high and dangerous. It is not yet far –at detect date- from its first existence. So, I would rank it as (2).

Network countermeasures = 2

Part of targets *(part b)* are not protected by/behind a network firewall, even though some of them seems to have a host firewall, other many targets do not, and they response to the attack. The other part’s (*part c)* targets are probably protected by/behind a firewall. The risk is high, so I would rank it as (2).

So the conclusion is:
Severity = (3+5)-(2+2) = 4

Defensive recommendation

We can brief the defense status by exploring Part (b) and (c) of the Detects:

- We can easily divide ***Part (b)*** into two sections:

    - Section one includes those hosts that acknowledged back the connection, which indicates that they are not protected by a firewall, and they accept such connection. Such hosts are: 172.16.119.157, 172.16.119.158, 172.16.126.72, 172.16.162.54
    - Section two includes those hosts that refused the connection by resetting it. Such responses could be carried out by IDS. A host in this section could be IDS, firewall, or has a host firewall. Such hosts are: 172.16.117.148, 172.16.117.149, 172.16.117.194.
- ***Part (c)*** includes ICMP replies. We have three kind of messages :
    - Destination Unreachable (Type 3): Original packet -that trigger ICMP replies- and portion of its payload is attached with the ICMP packet.
        - Host Unreachable (Code 1) means that the gateway that protects a target denied the access, and responded by this message. Such gateway is: 172.16.174.2 and such protected host is: 172.16.164.66.
        - Port Unreachable (Code 3) is most probably sent by the target host, because the required port (135) is not active/open. Such hosts -that might not be a Windows machine- are: 172.16.119.38, 172.16.124.180, 172.16.124.190.
    - Time to live exceeded in transit (Type 11, Code 0) means that the packet exceeded its time to live value, so the gateway sent back ICMP message to the sender. The strange here is the TTL value does not reach "0" value, but still the gateway send that ICMP message, which is not clear for me, unless it is a way to fool the attacker. Such gateways are: 172.16.96.17, 172.16.161.1, 172.16.162.129.

To defeat this attack, it is recommended to do the following:

- Apply patches to your system, by either using Windows Update site to update your system generally, or referring to Microsoft Security Bulletin MS03-026 [5], and pick the proper specific patch.
- Microsoft recommend to block -on the firewall- any traffic going to all ports used to initiate an RPC connection [5] :
    - UDP ports 135, 137, 138 and 445.
    - TCP ports 135, 139, 445, and 593.
- Block any traffic coming from outside to TFTP port (69).
- Disable DCOM interface, by either editing Windows registry, or running "DCOMCNFG.EXE". Microsoft does not recommend this option until you estimate the effect on your network. For further help refer to Microsoft Knowledge Base Article – 825750 [7].
- Disable RPC over HTTP, which listen on ports 80 and 443. "RPC over HTTP is controlled by two registry entries on the computer running IIS and the RPC proxy" [8].

Multiple choice test question

How does W32/Blaster worm act as a DDoS attack tool?!

a) Scan for vulnerable hosts.
b) Use TCP port (135) to exploit buffer overflow vulnerability and then execute arbitrary codes.
c) On specific dates, each compromised host –by this worm- launch a DoS attack against Windows Update web site.
d) All above answers.

Answer: d

References (Part-2, Detect-1)

[1] CERT® Coordination Center. “CERT Advisory CA-2003-20 W32/Blaster worm”. August 14, 2003. URL: http://www.cert.org/advisories/CA-2003-20.html

[2] Common Vulnerability and Exposures. "CAN-2003-0352 (under review)". May 28, 2003. URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0352

[3] *Gaerlan, Maria Joan.* “WORM_MSBLAST.GEN - Technical details”. Virus Encyclopedia, TrendMicro. Aug. 15, 2003. URL: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_MSBLAST.GEN&VSect=T

[4] SANS Institute. "Internet Storm Center". Spetember 29, 2003. URL: http://isc.incidents.org/

[5] "Buffer Overrun In RPC Interface Could Allow Code Execution (823980)". Microsoft Security Bulletin MS03-026. September 10, 2003. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-026.asp

[6] Postel J. "INTERNET CONTROL MESSAGE PROTOCOL". RFC 792. September 1981. URL: http://www.ietf.org/rfc/rfc0792.txt

[7] "How to Disable DCOM Support in Windows". Microsoft Knowledge Base Article – 825750. Aug 12, 2003. URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;825750

[8] "RPC over HTTP Security". Microsoft Remote Procedure Call. February 2003. URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/rpc/rpc/rpc_over_http_security.asp

Detect(2) – DNS named version attempt

```
[**] [1:1616:4] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/04-01:39:59.584488 210.195.43.32:1996 -> 46.5.253.75:53
UDP TTL:44 TOS:0x0 ID:53244 IpLen:20 DgmLen:58 Len:  38
[Xref => arachnids 278][Xref => nessus 10028]
```

- **Snort Logs: “2002.4.5”**

```
06/04-04:39:59.584488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:1996 -> 46.5.253.75:53 UDP TTL:44 TOS:0x0 ID:53244 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4..........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03     sion.bind.....

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

06/04-06:49:29.724488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:1556 -> 46.5.149.192:53 UDP TTL:44 TOS:0x0 ID:53027 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4..........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03     sion.bind.....

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

06/04-08:09:18.324488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:1090 -> 46.5.55.205:53 UDP TTL:46 TOS:0x0 ID:2572 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4..........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03     sion.bind.....

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

06/04-08:35:19.444488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4202 -> 46.5.37.23:53 UDP TTL:46 TOS:0x0 ID:28790 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4..........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03     sion.bind.....

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

06/04-09:18:14.324488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4919 -> 46.5.46.101:53 UDP TTL:44 TOS:0x0 ID:6424 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4..........ver
```

```
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-09:21:37.714488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4460 -> 46.5.10.2:53 UDP TTL:46 TOS:0x0 ID:9837 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-10:13:20.764488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:2556 -> 46.5.240.181:53 UDP TTL:44 TOS:0x0 ID:62068 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-10:38:17.174488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4573 -> 46.5.43.242:53 UDP TTL:46 TOS:0x0 ID:21705 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-10:38:34.184488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4863 -> 46.5.198.189:53 UDP TTL:44 TOS:0x0 ID:21986 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-10:43:28.564488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:2010 -> 46.5.88.70:53 UDP TTL:44 TOS:0x0 ID:26933 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-10:57:29.824488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4649 -> 46.5.13.133:53 UDP TTL:44 TOS:0x0 ID:41079 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-11:07:42.034488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:3309 -> 46.5.209.234:53 UDP TTL:44 TOS:0x0 ID:51363 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

06/04-11:08:44.974488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4409 -> 46.5.134.238:53 UDP TTL:44 TOS:0x0 ID:52431 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....

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

- ## Source of Trace

I picked file "2002.5.4" from raw TCP DUMP logs that are available on [incidents.org](http://www.incidents.org), at the following URL path [http://www.incidents.org/logs/Raw]. I did some analysis based on the log file, to come up with approximate network topology. Log packets have only two MAC addresses [`"0:0:C:4:B2:33"`, `"0:3:E3:D9:26:C0"`], with different IP addresses, which indicate that sniffer has been put between two gateways, and all other networks are behind those two gateways. I noticed –mostly- two sanitized networks, one is your internal network and includes all (46.5.x.x) addresses, and second is your public/DMZ network which includes all (64.154.x.x) addresses. Figure-3 draws a picture of the approximate network topology.

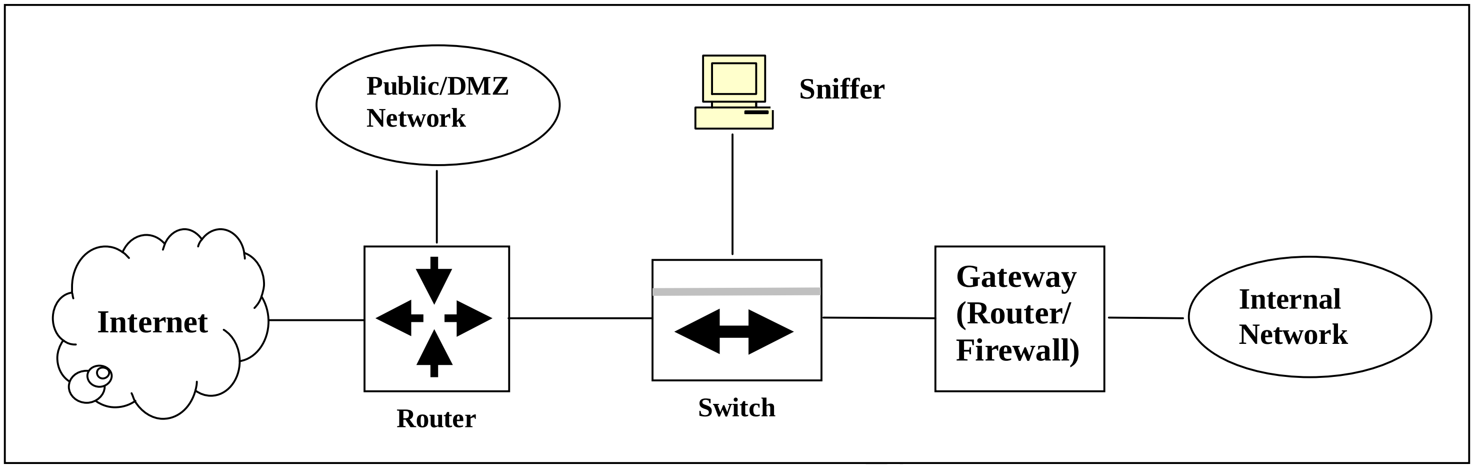


Figure-3: Network Topology

- ## Detect was generated by

Logs are generated using Snort version (1.9.0) with default rule set, on a Linux box. First I run snort with these options (snort -N -c snort.conf -r 2002.5.4) to get the alert file, I noticed –then- many "`DNS named version attempt`" alert messages. I discovered a DNS scan process, from different sources, and I picked the largest scan attack source “210.195.43.32” -for further analysis- by issuing this command:

```
snort -dev -r 2002.5.4 ”src host 210.195.43.32” > attack.log
```

Prior command -based on the current log- showed only UDP scan packets to port 53 –that ask for bind version-, but nothing else that “210.195.43.32” did. Current log do not show any response packets to all scan packets in *Snort Logs* section, and even no ICMP packets have been sent to the attack source “210.195.43.32”, which means nothing, since one of them –UDP reply / ICMP message- should had happened. I run the two following commands to find out:

```
# No response packets have been detected for all scan packets in Snort Logs section
snort -dev -r 2002.5.4 ”src port 53 and dst host 210.195.43.32 and udp”

# No ICMP messages.
snort -dev -r 2002.5.4 ”icmp and dst host 210.195.43.32”
```

Log Format

Log format is in a snort standard format as a Sequence of packets. Each packet consists of:

- Date & Time.
- Data Link layer headers: Source and Destination address, protocol, and packet length.
- For next Two mixed Layers: you need some knowledge of IP Header, ICMP, and Transport Layer Headers (TCP, UDP). We will explain some fields :
    - IP Header fields :
        - TTL: Time to Live.
        - TOS: Type of Service.

        - ID: packet identification.
        - IpLen: IP header Length.
        - DgmLen: IP packet Total Length.
    - UDP Header fields :
        - UDP: UDP header length.
- Application Layer data in hex and ASCII.

Snort Rule
From "dns.rules" file, I used the following rule:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt";
content:"|07|version"; nocase; offset:12; content:"|04|bind"; nocase; offset: 12;
reference:nessus,10028; reference:arachnids,278; classtype:attempted-recon;
sid:1616; rev:4;)
```

Probability the source address was spoofed

The source IP is not spoofed, because current detect is a UDP scan for DNS servers with a bind version query, and this is so clear from captured packets in *Snort Logs* section. Each scan process needs a response/reply to collect information, otherwise it would be useless. If source IPs were spoofed, then replies would not get back to them.

[whois -h whois.apnic.net] query shows the IP owner:

```
inetnum:        210.195.0.0 - 210.195.255.255
netname:        TMNET-MY
descr:          TMnet Telekom Malaysia
country:        MY
admin-c:        AS115-AP
admin-c:        EU3-AP
admin-c:        SM135-AP
tech-c:         AS115-AP
tech-c:         EU3-AP
tech-c:         SM135-AP
mnt-by:         APNIC-HM
mnt-lower: TM-NET-AP
changed:        hostmaster@apnic.net 20010820
status:         ALLOCATED PORTABLE
source:         APNIC
```

Description of attack

It is a UDP scan to port (53) looking for DNS servers, that includes a bind version query. Each captured packet from *Snort Logs* section is a simple UDP DNS query packet, that ask for bind version. Attack source is “210.195.43.32” and –as expected- came from the Internet. Scan targets are from the sanitized class B ”46.5.x.x” that most probably reside in the internal network. Destination port is the DNS default port (53). Packets payload contains the following pattern (.4...........version.bind.....), but the key string here is ” version.bind“ which tells that this packet is a bind version query. Bind version string appears on offset “12” in the UDP packet payload, so snort rule looks for that string starting from that offset.

Query a DNS server for its bind version is so easy and basic. It can be done by querying the CHAOS TXT record 'version.bind'. You can do it by such the following query that uses dig command [dig version.bind @dns.server –t txt –c chaos]. DNS servers were usually configured to answer such queries by default, but this is switched off on Bind version “9”.

To be able to launch an attack on a specific network, you need –mostly- first to gather information about that network. You may scan a network by simple ICMP ping, or advance TCP/UDP packets, that can not just say who is alive and who is not, but to tell the service version that listen on a specific port.

There is no direct Common Vulnerabilities and Exposures (CVE) record for the bind version query attempt, but there is indirect relation between it and buffer overflow vulnerability on specific bind versions. If the responded DNS server is vulnerable, then attacker can exploit it. Buffer overflow vulnerability has the following CVE record “CVE-1999-0009”.

▪ Attack mechanism

Attack is carried on by sending number of UDP packets that query each target IP about bind version. If we assumed that the log already captured all similar packets from the same source, within the start and end time of the log, then we would say that time between packets were random, which helped in slow down the scan process a little bit–It was not definitely a fast scan-. Targets IPs were random too and picked up from within (46.5.x.x) class B. Attacker hits only one IP from each class C every time, then –next- hits another IP from different class C, and so on. The log shows only one detected packet for each captured class C. Both random time and random target IPs were used to scatter the scan packets, in a hope that they would be unnoticed. TTL values were consistent → 44, but sometimes were not → 46. Route path is not always the same even for the same source, according to the network status along the possible paths, which may affect TTL values, so I just ignore this point.

Such scan is usually one step among initial steps to collect information about the target network. By knowing DNS Bind version, attackers can pick up proper tools that exploit this particular version, and launch attack against that particular DNS server, to gain root access, or to denial its service…etc.

▪ Correlations

Scanning is done every day, and scanning a network for DNS servers that includes a version.bind query is still active. I found two kinds of correlations: direct/indirect. I will talk about them, and I will list references:

- *Direct* Correlation: *DNS named version attempt* is recognized and recorded by
    - Snort rule set with two rules. One for UDP attempt as listed earlier in *Snort Rule*, and the other is to detect TCP version of DNS *named version attempt*. TCP Snort rule is written down:

```
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; reference:nessus,10028;
reference:arachnids,278; classtype:attempted-recon; sid:257; rev:6;)
```

    - *Whitehats.com* under the following name (IDS278 "*NAMED-PROBE-VERSION*"). URL is [http://www.whitehats.com/info/IDS278]

    - *Internet Security Systems*, under ID (2000417), name (DNS Bind Version request), and URL path [http://www.iss.net/security_center/advice/Intrusions/2000417/default.htm]
    - *Nessus.org* plug-in that help to determine which version of Bind name daemon is running. URL path is [http://cgi.nessus.org/plugins/dump.php3?id=10028]
    - *Incidents.org* archives in the following messages path :
        - http://www.incidents.org/archives/intrusions/msg00087.html
        - http://www.incidents.org/archives/intrusions/msg03827.html
    - Posted messages to intrusions@incidents.org , such as:
        - http://cert.uni-stuttgart.de/archive/intrusions/2003/07/msg00065.html
        - http://cert.uni-stuttgart.de/archive/intrusions/2003/02/msg00191.html
    - *Neohapsis.com* archives, such as [http://archives.neohapsis.com/archives/snort/2002-02/0345.html]

- *In Direct* Correlation: Most Security web sites correlate between *DNS named version attempt* and other DNS incidents. Such web sites are:
    - *Common Vulnerabilities and Exposures (CVE)* record an Inverse query buffer overflow in BIND 4.9 and BIND 8 Releases incident under following name “CVE-1999-0009”. URL path is [http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0009]

- *Internet Storm Center*, Where port (53) is among the top attacked ports. Port (53) Attacks include following: buffer overflow, TCP scan, UDP scan …etc. URL path is [http://isc.incidents.org/port_details.html?port=53]
- *Securityfocus.com* talked about Multiple Vendor BIND query buffer overflow Vulnerability, under Vulnerabilities section. URL path is [http://www.securityfocus.com/bid/134]
- *Internet Security Systems* list two vulnerabilities:
  - NXT overflow: A bug in version 8.2 and 8.2.1 that allow hackers to break into DNS system. ID is (2000415), name is "DNS NXT record overflow", URL path is [http://www.iss.net/security_center/advice/Intrusions/2000415/default.htm]
  - IQUERY overflow: A bug in version 4.9.6 that allows a hacker to break in. ID is (2000410), name is "DNS I-Query exploit", URL path is [http://www.iss.net/security_center/advice/Intrusions/2000410/default.htm]

Evidence of active targeting

It is a general scan for the entire class B network that has been sanitized with (46.5.x.x). This is so clear when you look to scan targets in Snort logs, read attack description and attack mechanism.

Severity

Severity is calculated using the following formula:
Severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality = 5
Attack is destined to DNS servers, and then criticality is (5).

Lethality = 2
It is a scan process that does not have direct harm to target hosts. Lethality would be (2).

System countermeasures = 4
Bind version (9) do not answer "version.bind" query by default, so any bind.verion query would not get useful answer. Not all DNS servers use Bind version (9), and even version (9) Bind's answer does not blind the attack totally, because any answer can be analyzed to get some information from it. Any way *Snort* Logs do not show any reply, so I would rank it as (4).

Network countermeasures = 3
Network security devices can not help much here unless we have a content filtering device, or Intrusion Prevention System, because *version.bind* is just similar to any other DNS UDP resolving query.

Snort Logs do not show any response, or ICMP messages to the attack, which can be taken as indication that the network has been protected, but -in the same time- we can not ignore the probability the response was not captured, because –simply- it just seems as any legitimate traffic. I would rank it as (3).

So the conclusion is:
Severity = (5+2)-(4+3) = 0

Defensive recommendation

Snort Logs do not show any response to version.bind UDP query, which indicates that attack did not succeed. To protect your network, and defeat such attack, you need to do the following:

- Configure your Bind to not answer such query, or update your Bind to the latest Bind version, where this configuration option is there by default. I would recommend updating your Bind to the latest version, to resolve all existing vulnerabilities.
- Install Content-Filtering, or Intrusion Prevention System Device, where such traffic would be blocked, before reaching its destination.

Multiple choice test question

Which one of the following is probably called a *DNS named version* attempt:

a)

```
06/04-03:43:13.504488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x3C
255.255.255.255:31337 -> 46.5.13.160:515 TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
63 6B 6F                                         cko
```

b)

```
06/04-08:35:19.444488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x48
210.195.43.32:4202 -> 46.5.37.23:53 UDP TTL:46 TOS:0x0 ID:28790 IpLen:20 DgmLen:58
Len: 30
12 34 00 80 00 01 00 00 00 00 00 00 07 76 65 72  .4...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....
```

c)

```
06/04-04:03:31.544488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x46
217.35.53.194:2632 -> 46.5.180.133:21 TCP TTL:108 TOS:0x0 ID:33457 IpLen:20 DgmL
en:56 DF
***AP*** Seq: 0xBCCB4A0  Ack: 0x7EE89664  Win: 0xC51  TcpLen: 20
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..
```

Answer: b

Detect(3) - BAD-TRAFFIC tcp port 0 traffic

```
[**] [1:524:6] BAD-TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
11/11-15:28:13.616507 211.47.255.21:35981 -> 207.166.155.132:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6A39ECBB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:6] BAD-TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
11/11-01:00:39.616507 211.47.255.22:60086 -> 207.166.237.132:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9BD2B58B  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

Snort Log: “2002.10.11”

```
11/10-18:00:18.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60086 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9BD2B58B  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:00:21.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60086 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9BD2B58B  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

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

11/10-18:00:27.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60086 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9BD2B58B  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:00:39.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60086 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9BD2B58B  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:00:50.626507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60684 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9E50FF67  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:00:53.626507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60684 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9E50FF67  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:00:59.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60684 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9E50FF67  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:11.626507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:60684 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x9E50FF67  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:22.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33063 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA001E649  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:25.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33063 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA001E649  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:31.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33063 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA001E649  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:43.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33063 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA001E649  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

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

11/10-18:01:54.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33663 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA1B9E452  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:01:57.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33663 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA1B9E452  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:02:03.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33663 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA1B9E452  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/10-18:02:15.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.22:33663 -> 207.166.237.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0xA1B9E452  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:28:13.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:35981 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6A39ECBB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:28:19.606507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:35981 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6A39ECBB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:28:31.706507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:35981 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6A39ECBB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:28:42.616507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36227 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6BF461C0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:28:51.666507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36227 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6BF461C0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:03.636507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36227 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6BF461C0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
```

```
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:14.796507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36489 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6E08AB51  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:17.756507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36489 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6E08AB51  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:23.756507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36489 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6E08AB51  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:36.686507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36489 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6E08AB51  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:49.656507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36769 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6F4869F0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:29:55.736507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36769 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6F4869F0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

11/11-08:30:07.736507 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x42
211.47.255.21:36769 -> 207.166.155.132:0 TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x6F4869F0  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

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

▪ Source of Trace

I picked file "2002.10.11" from raw TCP DUMP logs that are available on incidents.org, at the following URL path [http://www.incidents.org/logs/Raw]. I did some analysis based on the log file, to come up with approximate network topology. Log packets have only two MAC addresses [`"0:0:C:4:B2:33"` and `"0:3:E3:D9:26:C0"`], with different IP addresses/networks, which indicate that sniffer has been put between two gateways, and all other networks are behind those two gateways. First gateway –I will call it Boarder router- seems to connect your network to Internet, according to variant networks that come through it. Second one –I will call it Access router- seems to connect your network to a couple of networks (all share the same prefix “207.166.x.”) that have been monitored by the sniffer, and have not been sanitized, which means you care about them, but they are not yours. Such networks –I will call them Customers network- are not on the same security level with those that are connected to the Boarder router. Current log file shows following customers network {207.166.87.x, 207.166.155.x, 207.166.237.x …etc}.

In (Figure-4) I used prior input from “*Detect(2) – DNS named version attempt*”, and combined it with current to draw –approximately- a picture of the network topology.

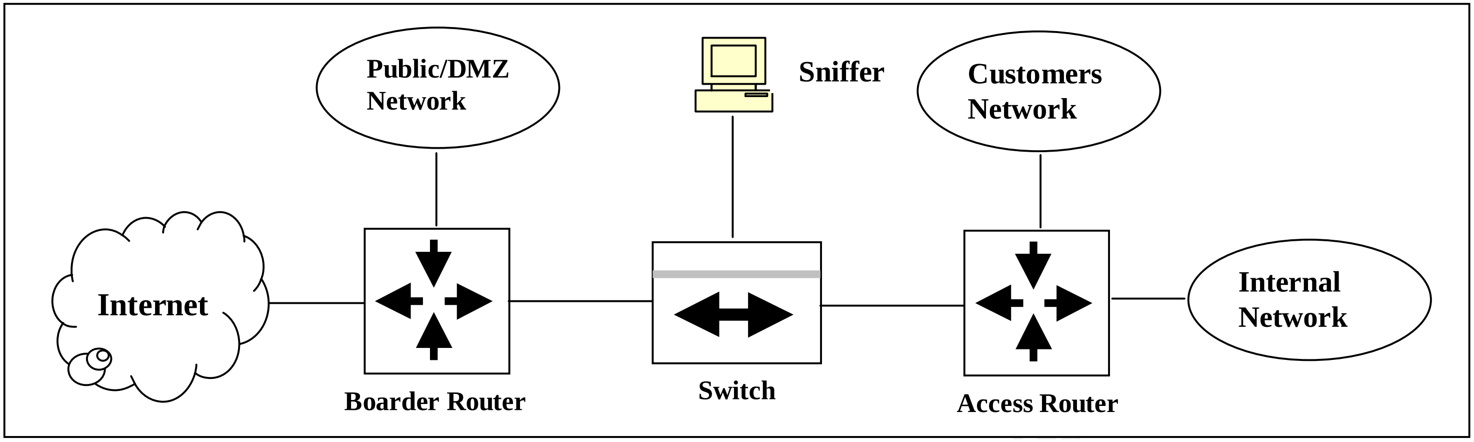


Figure-4: Network Topology

- **Detect was generated by**

Logs are generated using Snort version (1.9.0) with default rule set, on a Linux box. First I run snort with these options (snort -N -c snort.conf -r 2002.10.11) to get the alert file. I noticed –then- many "`BAD-TRAFFIC`" alert messages, from different –most probably- unrelated sources. I picked “BAD-TRAFFIC TCP port 0 traffic” alert message -as detect- for further analysis. I collected related logs by the following command:

```
snort -dev -r 2002.10.11 "tcp and dst port 0”
```

Logs show us only two sources from the same network {211.47.255.21 and 211.47.255.22}, and two targets {207.166.237.132 and 207.166.155.132}. The two prior sources did not send any other traffic. I investigated “2002.10.11” log file, to pick any related traffic, or to use any piece of log to draw a history background, so:

- I did not find any response to all SYN packets.
- I did not find any ICMP messages.
- No prior suspicious activities have been found against these two target networks, except for “SCAN Squid Proxy attempt”, but I strongly think that there is no relation between the two detects.
- No activities have been detected from these two networks {207.166.237.x and 207.166.155.x}.

Used commands are briefed in the following:

```
# Return non. All packets are for port 0.
snort -dev -r 2002.10.11 "src host 211.47.255.22 or src host 211.47.255.21 and not
dst port 0"

# SCAN Squid Proxy attempt
snort -dev -r 2002.10.11 "dst net 207.166.155 or dst net 207.166.237 and not dst
port 0"

# Return non Replies
snort -dev -r 2002.10.11 "tcp and src port 0"

# No ICMP messages
snort -dev -r 2002.10.11 "icmp"

# No traffic from these networks have been detected.
snort -dev -r 2002.10.11 "src net 207.166.237 or src net 207.166.155"
```

Log Format

Log format is in a snort standard format as a Sequence of packets. Each packet consists of:

- Date & Time.

- Data Link layer headers: Source and Destination address, protocol, and packet length.
- For next Two mixed Layers: you need some knowledge of IP Header, ICMP, and Transport Layer Headers (TCP, UDP). We will explain some fields :
    - IP Header fields :
        - TTL: Time to Live.
        - TOS: Type of Service.
        - ID: packet identification.
        - IpLen: IP header Length.
        - DgmLen: IP packet Total Length.
        - DF : Don't Fragment bit is set to one.
    - TCP Header fields :
        - ******S* : SYN bit is set.
        - Seq: Sequence Number.
        - Ack: Acknowledge number.
        - Win: Windows size.
        - TcpLen: TCP header length.
        - TCP options
- Application Layer data in hex and ASCII.

Snort Rule

From " bad-traffic.rules" file, I used the following rule:

```
alert tcp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD-TRAFFIC tcp port 0 traffic";
classtype:misc-activity; sid:524; rev:6;)
```

▪ Probability the source address was spoofed

It is not spoofed IP, but a real one, because the attack is a reconnaissance process, to fingerprint system OS, where the target reply is important to gather the information, otherwise it will be useless to send a spoofed packet to gather that information, if we know the feed back would not get back.

[whois -h whois.apnic.net] query shows the IP owner:

```
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
tech-c:       HM127-AP
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
mnt-lower:    MNT-KRNIC-AP
changed:      hostmaster@apnic.net 19991118
changed:      hostmaster@apnic.net 20010606
status:       ALLOCATED PORTABLE
source:       APNIC
```

- **Description of attack**

It is an OS Fingerprinting scan process that gathers information about targets in two steps, according to current logs:

- Send a SYN packet on –unexpected & reserved- TCP port 0, and wait for the target response, to detect system OS [1].
- Use the SYN-Flood Resistance method by sending consecutive SYN packets to TCP port 0 that could fill up TCP/IP stack, or exhaust system resources, to assess system behaviour, capabilities, TCP\IP stack size, which help to detect system OS [2].

Many scanning tools have its modules to detect system OS, as a one option/step during the scanning process. Such tools are: NMAP [http://www.insecure.org/], NESSUS [http://www.nessus.org/], ISS [http://www.iss.net/] …etc.

There is a CVE record for SYN-Flood attack under "CVE-1999-0116"[3], but there is no one for OS fingerprinting. We have another CVE record “CVE-1999-0675” [6], that describes a similar concept on UDP traffic that may deny check point firewall-1 from service.

- **Attack mechanism**

Each System platform has its own TCP/IP stack implementation, which means various system behaviour/response exists from one system to another according to input packet. You may use this fact to detect any system OS, by sending –for example- multiple TCP packets, with different options each time, and observe responses.

Port 0 is a reserved port, according to RFC-1700 "ASSIGNED NUMBERS", and no packets are expected to use it, which means that there is no clear way on how to handle it, but each OS platform has its own way, and can be predicted for that. This method is called "port 0 OS fingerprinting" [1].

Another way to predict OS is by TCP/IP stack size. You may use the "SYN Flood Resistance" method where you send many TCP SYN packets, with no intention to complete the three hands shaking, and establish the connection. Your intention is to assess the size of the stack, and system behaviour.

Attack under analysis combines the two methods into one, and launched the attack from two sources {211.47.255.21 and 211.47.255.22} from the same network, against only two particular targets {207.166.155.132 and 207.166.237.132}. Both attacks were Interested on only two targets that share the same last IP portion (host number=132). This interest was not –mostly- an accident, but may be as a result of a prior scan. The attack load was not heavy, which indicates that the two targets are most probably a host (Server or Desktop PC/Station). The two attacks followed a similar style, which means they are probably managed by one team or different individuals but with the same tools.

“211.47.255.22” launched the attack first. It hit the first target “207.166.237.132” with 16 packets in 2 minutes. The time period was varying, but it seems to follow some pattern as following: {3, 6, 12, 11, 3, 6, etc start over again}. We can classify time periods in 4 groups of 4 time period elements. Each time period group includes the following: {3, 6, 12, and 11}. Packets that are within the same time period share the same source port. Later on, after a gap of many hours, the second attack source “212.47.255.21” hit the second IP “207.166.155.132” 13 times. The time period between the packets was varying, and pattern has been figured out, but it did not exceed 13 seconds.

Each attack source tried to hide the scan process by slowing it down a little bit -as possible-, but the time was not wide open, and must be within a limits, so the TCP/IP stack had to be filled before the first un complete session timed out, and removed from it. Both attack sources finished the scan within almost 2 minutes.

- **Correlations**

As much you know about your victim, the better and right tool you pick to attack it. Remote OS detection is usually a part of any scanning process, which is a part of the reconnaissance process of any attack. It is almost a fact, that there is no attack without a scan, and usually there is no scan, without OS detection.

One way to detect system OS is via TCP/IP stacks fingerprinting. "Port 0 OS fingerprinting" and "SYN-Flood Resistance" are methods of TCP/IP stack fingerprinting.

Many correlated references are found, and will be classified below:

- DoS vulnerabilities in TCP\IP stacks (SYN-Flood): Denial of service is possible whenever the attacker is able to consume the victim resources (i.e. TCP/IP stacks), and SYN-Flood is such attack. Many references announced this knowledge as following:
    - Common Vulnerability and Exposures announced it under the following candidate number “CVE-1999-0116", and URL path is [http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0116].
    - CERT® Coordination Center announced it as “CERT Advisory CA-2000-21 Denial-of-Service Vulnerabilities in TCP/IP Stacks”, and URL path is [http://www.cert.org/advisories/CA-2000-21.html]
    - BindView's RAZOR team discovered set of network DoS vulnerabilities and the name Naptha is being used to describe them as a group. URL path is [http://razor.bindview.com/publish/advisories/adv_NAPTHA.html].
    - *Securityfocus.com* has an article helps in Hardening TCP/IP stack to defeat SYN denial of service. The article topic is “Hardening the TCP/IP stack to SYN attacks”, and URL path is [http://www.securityfocus.com/infocus/1729].
- OS fingerprinting references:
    - *Insecure.org* web site published an article, with topic “Remote OS detection via TCP/IP Stack Fingerprinting”. This article discussed ways to get information about a host, by querying its TCP\IP stack. URL path is [http://www.insecure.org/nmap/nmap-fingerprinting-article.txt].
    - *Networkpenetration.com* published a paper written by “Ste Jones”, under a topic “Port 0 OS Fingerprinting”, that describes how to use port 0 to fingerprint a system OS. URL path is [**http://www.networkpenetration.com/port0.html**]
    - *Internet Storm Center*, Where port (0) is among the top attacked ports, and includes varies kinds of attacks, not necessary to be OS fingerprinting. URL path is [http://isc.incidents.org/port_details.html?port=0]
    - *Snort.org* has two rules that detect any traffic (TCP/UDP) to port0, and classify it as possible reconnaissance. I will write here only the UDP rule.

```
alert udp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD-TRAFFIC udp port 0 traffic";
reference:cve,CVE-1999-0675; reference:nessus,10074; classtype:misc-activity;
sid:525; rev:5;)
```

▪ Evidence of active targeting

The attack has been carried on against two particular hosts: 207.166.237.132 and 207.166.155.132.

▪ Severity

Severity is calculated using the following formula:
Severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality = 3

Attack is most probably destined to a host, that can be server, or a desktop PC, so criticality is (3).

Lethality = 4

It is a reconnaissance process, but it can deny the host service, so Lethality would be (4).

System countermeasures = 2

There are some patches, and procedures to harden TCP/IP stack, and System OS, but they can not guarantee total protection. We are not sure if such measures were implemented on target hosts, but logs did not show any response that contradicts this claim, so I would rank it as (2).

Network countermeasures = 3

According to the following facts I would rank it as (3):

- Port 0 OS fingerprinting" can be easily blocked by the firewall.
- DoS via TCP/IP stack can be controlled, and denied via the firewall, but not guaranteed for 100%.
- The attack did not succeed –according to available log-. There were no responses.

So the conclusion is:
Severity = (3+4)-(2+3) = 2

▪ Defensive recommendation

Snort Logs do not show any response, so I am going to assume that the attack did not succeed, even though full information was not available (such as firewall ICMP reply).

To defeat such attack you need –in general word- to lie, or to hide your identity. To be more specific, we need to divide the attack to its two elements:

- Port 0 OS fingerprinting:
  - It can be defeated by blocking any traffic use this port at the firewall.
  - The rest of solutions –below- are applicable here too.
- SYN-Flood Resistance:
  - Hardening the TCP/IP stack by applying the right vendor patch.
  - Follow the vendor procedure to tune your operating system, to be more resilient.

▪ Multiple choice test question

Which is true about TCP port 0?

a) It can be used to detect system OS by TCP/IP fingerprinting.
b) It is a reserved port for a special use, and should not be used by any legitimate traffic.
c) This port number is reassigned by the OS.
d) No program should be listening on port 0 and no program should connect from port 0 thus it should be blocked on your firewall.
e) All above answers.

Answer: e

▪ Questions & Answers from incidents.org Posting

On Wed, 29 Oct 2003, Timm, Kevin wrote asked questions and my answers are followed.

Q **You are missing two important pieces with the timing. These pieces will tell you something about the target.**

All displayed packets were coming from the attack sources, so if you figured any thing according to those packets, it would be -then- related to the source not the target.

If you mint by the two important pieces with the timing, that they can help to figure the target OS/Platform, then I think I really miss this particular information. Otherwise, if you mint that this information can help to know the attack goals, and so would help on detecting the target type, or job, then this is true, and I wrote something about that, when I said:
[
The attack load was not heavy, which indicates that the two targets are most probably a host (Server or Desktop PC/Station).
]
I described the time period between each packet, and I draw a pattern line, but I did not get much information from it.

All information has been written under detect topics.

Q **When it comes to fingerprinting I think that is it important to note that NMAP uses several different tests and that by port 0 alone very few determinations can be made. If this was an**

NMAP scan are there Snort signatures that would detect the other traffic? Likewise with other tools.

I have to say that my analysis relied on the current log "2002.10.11".

I did not say the scan was an NMAP scan.

This scan or OS fingerprinting was may be a part of a large scan process, divided on many days. So it is true that using port 0 could not be enough, but who said that the attacker used only this method.

Snort has a rule set in a file named as "scan.rules", Current detect was obtained from "bad-traffic.rules" file.

I have to tell that I did not find many resources that talk or describe this kind of fingerprinting. Refer to the reference list at the end of the detect.

Q **What about IP ID? Does that tell you anything?**

IP ID = 0.
We agree it is suspicious. We know that using IP ID = 0 is legal, but the IP ID should differ each time (Incremental/Random), and not be equal.

If you forge any packet, you would construct it as you like.

This is what it tells me.!! Nothing else.?!!

References (Part-2, Detect-3)

[1] **Jones Ste. "Port 0 OS Fingerprinting". July 28, 2003. URL: http://www.networkpenetration.com/port0.html**

[2] Fyodor. "Remote OS detection via TCP/IP Stack Fingerprinting". October 18, 1998. URL: http://www.insecure.org/nmap/nmap-fingerprinting-article.txt

[3] Common Vulnerability and Exposures. "CVE-1999-0116". Sep 25, 1999. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0116

[4] CERT® Coordination Center. "CERT Advisory CA-2000-21 Denial-of-Service Vulnerabilities in TCP/IP Stacks". December 4, 2000. URL: http://www.cert.org/advisories/CA-2000-21.html

[5] **RAZOR** team security researchers. "The Naptha DoS vulnerabilities". November 30, 2000. URL: http://razor.bindview.com/publish/advisories/adv_NAPTHA.html

[6] Common Vulnerability and Exposures. "CVE-1999-0675". Oct 13, 2000. URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0675

[7] Burdach, Mariusz. "Hardening the TCP/IP stack to SYN attacks". September 10, 2003. URL: http://www.securityfocus.com/infocus/1729

Part 3 – Analyze This

Executive Summary

This is a scenario based security audit for a UMBC University. I have been given three (3) log types (Alerts, Scan and OOS), and five (5) consecutive days of log files for each type, that covers the period of "19 Oct 2003" to "23

Oct 2003". Log files contained (1,400,738) Total Alert events, (11,699,747) Scan events and (21,319) OOS data events.

Analysis shows us scanning/hacking activities toward UMBC University. It shows us a lot of scanning and some hacking activities generated from UMBC University as well. It shows some internal compromised hosts. It shows us that a lot of detected traffic/scanning was most probably for file sharing app related stuff.

My general guide line recommendations are: Prevent Inbound connections to your internal network. Control access to your public services. Prevent outbound connection to well known exploited ports. Limit access and use of file sharing application. Keep your systems patched and do not trust internal hosts as well as external hosts.

Analyzed Files

For each file type: Alerts, Scans, OOS (Out Of Spec), I selected five (5) consecutive days log files from "19 Oct, 2003" to "23 Oct, 2003". Logs were picked up from the following path [http://www.incidents.org/logs/]. The following table list these log files.

Alerts	Scans	OOS
alert.031019	scans.031019	OOS_Report_2003_10_19
alert.031020	scans.031020	OOS_Report_2003_10_20
alert.031021	scans.031021	OOS_Report_2003_10_21
alert.031022	scans.031022	OOS_Report_2003_10_22
alert.031023	scans.031023	OOS_Report_2003_10_23

The Analysis

For the period of "19 Oct 2003" to "23 Oct 2003" there are total of (13,121,789) events distributed as following:

- (1,400,738) Alerts events, and (285,102) Alerts if we exclude SCAN events {spp_portscan, NMAP TCP ping! and Null scan!}.
- (11,699,747) Scan events.
- (21,319) OOS data records.

Seven sections are followed to cover all required analysis aspects, but first we have to agree on some definitions:

- "MY.NET" in Alerts log files and Out-Of-Spec (OOS) log files is equal to "130.85" that appears in Scans log files.
- According to the high load of the scan events (1,115,636 Alerts are scan events), then I tried –generally- to isolate port scan events in Alerts log files and joined them -in analysis- with Scans log files.
- Alerts & Scans log files contain sometimes non formative lines. It is simply error lines probably due to bug of log manipulating script.

▪ List of Alerts

Alerts log files contained (50) alert types/classes. In Following I list all detects prioritized by number of occurrences, then I included two charts that visualizes the distribution of alerts. The first chart *Chart-1* views the Scan Percentage versus the remainder of alerts. The second chart *Chart-2* views the distribution of the rest of alerts (Excluding Scan alerts (i.e.: spp_portscan, NMAP TCP ping! And Null scan!).

Events	**No. of Detects**
spp_portscan	1114429
SMB Name Wildcard	199350
SMB C access	28548
MY.NET.30.4 activity	15606
EXPLOIT x86 NOOP	11563
Connect to 515 from inside	7126
MY.NET.30.3 activity	5727

TCP SRC and DST outside network	4518
External RPC call	3266
High port 65535 TCP - possible Red Worm – traffic	3172
Possible Trojan server activity	2009
ICMP SRC and DST outside network	1825
NMAP TCP ping!	752
SUNRPC high port access!	494
Null scan!	455
High port 65535 UDP - possible Red Worm – traffic	438
[UMBC NIDS IRC Alert] IRC user /kill detected, possible Trojan.	341
[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC	182
FTP passwd attempt	105
[UMBC NIDS] External MiMail alert	103
Back Orifice	84
TFTP - Internal UDP connection to external TFTP server	83
Incomplete Packet Fragments Discarded	74
Tiny Fragments - Possible Hostile Activity	62
[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC	55
EXPLOIT x86 stealth noop	53
NETBIOS NT NULL session	50
DDOS shaft client to handler	38
[UMBC NIDS IRC Alert] Possible drone command detected.	37
EXPLOIT x86 setuid 0	27
EXPLOIT x86 setgid 0	26
EXPLOIT NTPDX buffer overflow	25
DDOS mstream client to handler	14
FTP DoS ftpd globbing	14
TFTP - Internal TCP connection to external tftp server	13
[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.	12
TFTP - External UDP connection to internal tftp server	11
Attempted Sun RPC high port access	10
RFB - Possible WinVNC - 010708-1	10
External FTP to HelpDesk MY.NET	6
HelpDesk MY.NET.70.49 to External FTP	5
NIMDA - Attempt to execute cmd from campus host	4
[UMBC NIDS IRC Alert] K\:line'd user detected, possible Trojan.	4
[UMBC NIDS] Internal MSBlast Infection Request	3
TFTP - External TCP connection to internal tftp server	2
connect to 515 from outside	2
Probable NMAP fingerprint attempt	2
[UMBC NIDS IRC Alert] Possible trojaned box detected attempting to IRC	1
Bugbear@MM virus in SMTP	1
IRC evil - running XDCC	1

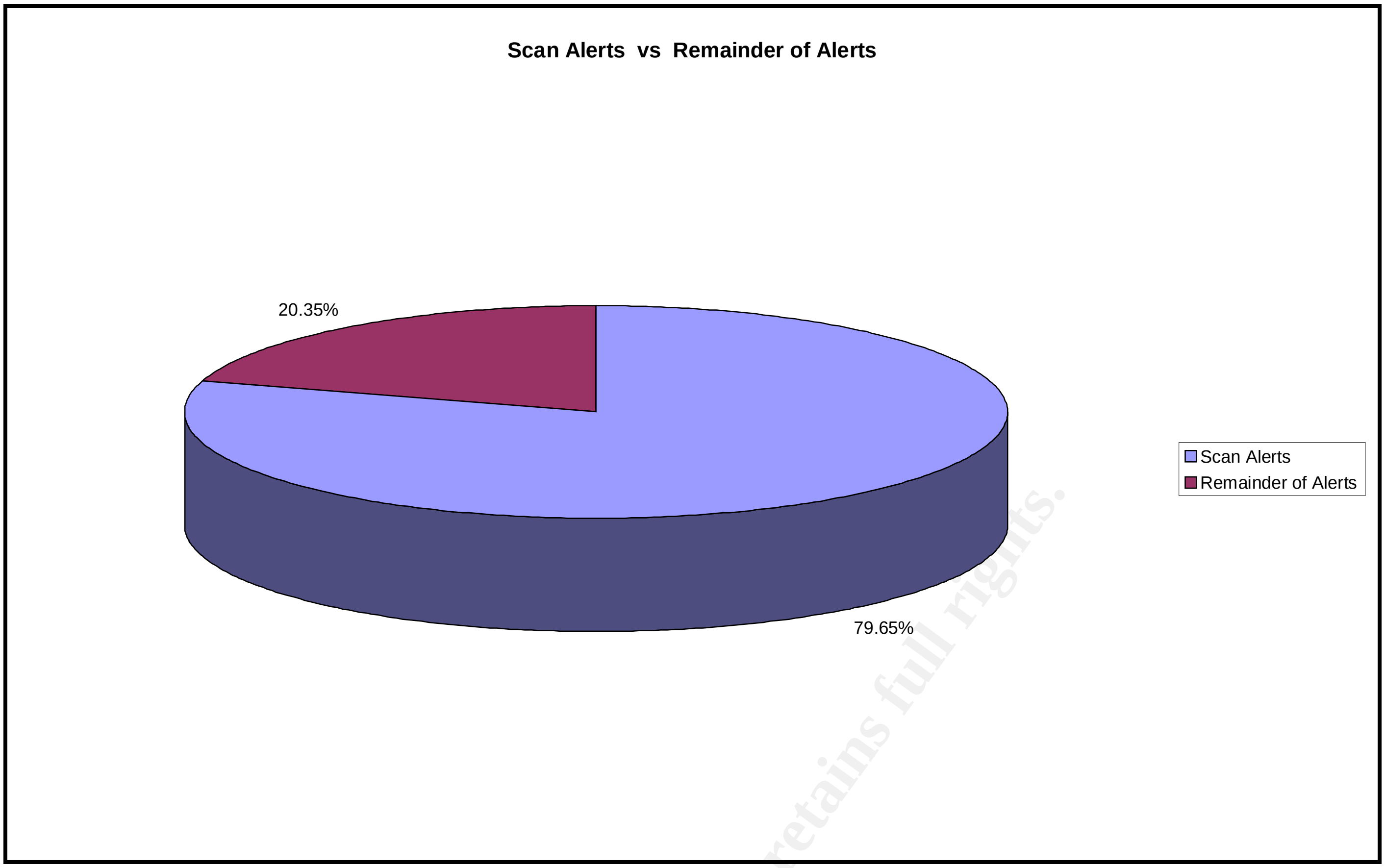


Chart 1

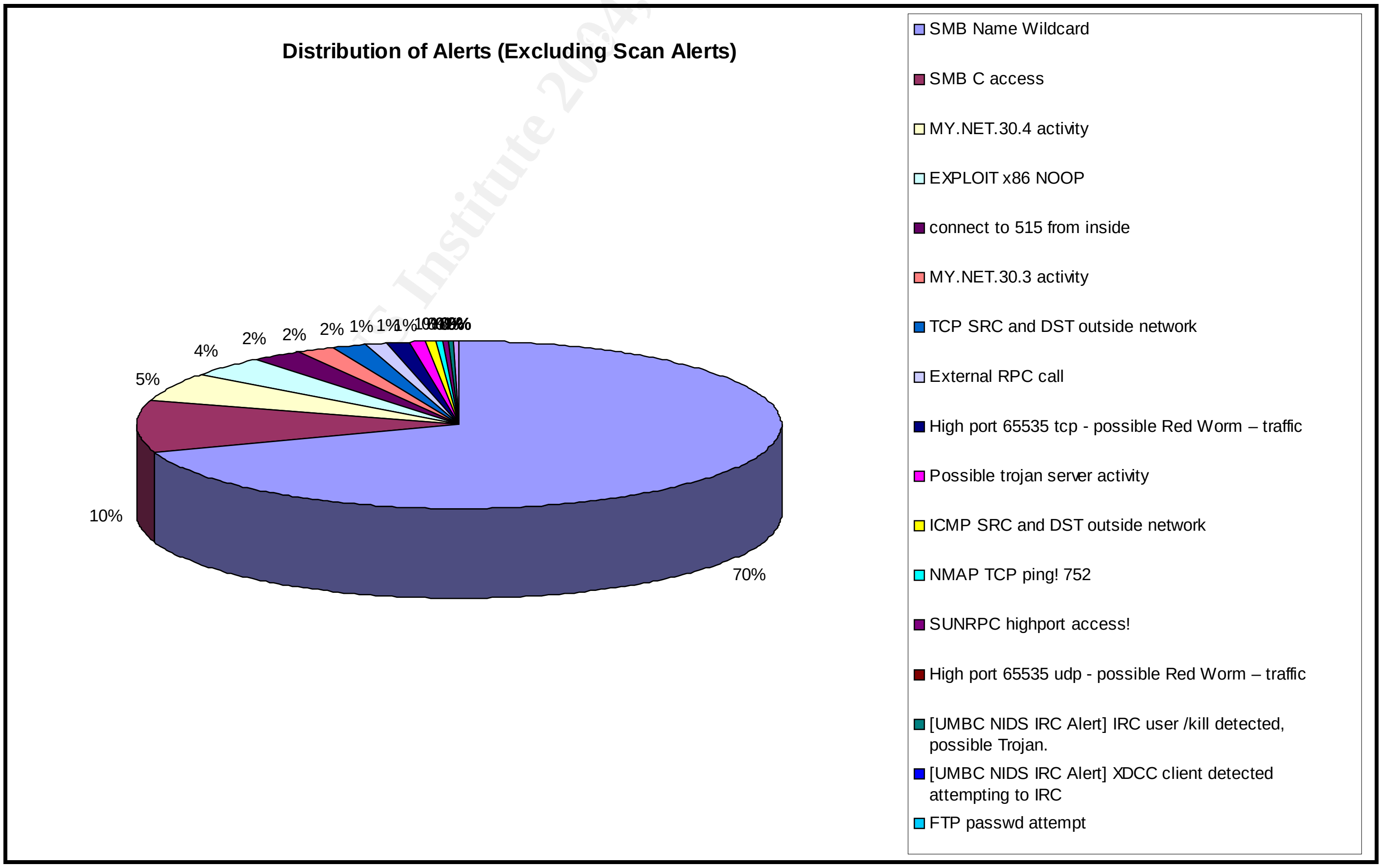


Chart 2

Alerts Brief Description

In this section, I will give a brief description of the top 10 detects, followed by a number of the rest detects selected randomly. Brief description list will not include “spp_portscan” and “Null scan!” alerts. Each description should –not must- include 5 parts or more: alert message indication, hosts involved on this alert, percentage of attack to other alerts, relation with other alerts, Defense Recommendation.

- **spp_portscan & Null scan!**

  Due to the high rate of scanning process, which covered (79.6%) of total detected alerts, I would rather combine it with *Scans* log files analysis later.

- **SMB Name Wildcard**

  There were (199350) events logged for this type, that covers (14.23%) of total alerts. Microsoft Server Message Block (SMB) is a protocol for sharing data and resources between computers (Windows & UNIX), and can be exploited to gain access to those resources. Windows hosts exchanges information via UDP port 137 –and other- as a part of the file sharing protocol. If this service is permitted from outside, then attacker can gather important information, that known to the targeted host, such as local network nodes, and users’ accounts. Alert messages show a huge traffic –most probably a scanning process- generated from at least (885) unique internal sources belong to the university network (MY.NET) toward at least (132,272) unique external IP addresses for what seems to be NetBIOS SMB service on port 137.

  Even though there is no solid evidence, but –at least- there is a possibility that all/many of internal sources have been compromised first, and then used later to launch this logged traffic. The other reasonable possibility is that these internal hosts have been used by the students intentionally to hack other hosts.

  Three tables are listed as following: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
MY.NET.80.51	115633
MY.NET.150.133	72074
MY.NET.29.2	3100
MY.NET.84.224	1291
MY.NET.150.198	474
MY.NET.42.9	193
MY.NET.17.34	143
MY.NET.84.154	141
MY.NET.111.65	133
MY.NET.150.44	118

TOP 10 Destinations	No. Events
198.62.205.6	1265
151.197.115.143	1251
193.114.70.169	1208
199.181.134.74	878
169.254.45.176	710
162.42.228.33	489
12.242.192.6	479
68.115.148.88	352
24.210.149.96	327
65.82.118.28	315

TOP 10 Traffic	No. Events
MY.NET.84.224:137 → 199.181.134.74:137	878
MY.NET.84.224:137 → 146.82.109.230:137	266
MY.NET.150.133:137 → 162.42.228.33:137	190
MY.NET.150.133:137 → 12.242.192.6:137	179
MY.NET.42.9:137 → 129.7.110.70:137	175
MY.NET.150.133:137 → 65.82.118.28:137	117
MY.NET.17.34:137 → 198.62.205.6:137	108
MY.NET.150.133:137 → 200.188.225.114:137	88
MY.NET.150.133:1457 → 68.115.148.88:137	79
MY.NET.84.224:137 → 207.44.204.46:137	77

As a recommendation to stop such traffic (internal → External), or (External → Internal), the following services have to be blocked on both direction, on routers & firewalls: [ports 137/tcp and UDP, 138/TCP and UDP, 139/tcp and UDP, and 445/TCP]. It is recommended too to patch your systems. Thirdly, you need to inspect the list of the top 10 internal sources; disconnect them from the network until you verify the situation, and recover them.

- **SMB C access**
  There were (28548) events logged for this type, that covers (2.04%) of total alerts. This event is generated when an attempt is happened to access a share via NetBIOS SMB service on port 139, such as trying to access a remote file located on a drive C in a host. If this service is permitted then attacker simply can gain access to C: file system. Unauthorized remote Administrative access means that system is totally compromised. (624) unique external IP addresses sent the traffic to (961) unique internal IP addresses belong to the university network (MY.NET).

  Three tables are listed as following: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
80.50.168.42	663
138.89.11.51	295
61.147.18.195	236
61.223.139.116	224
203.197.20.41	217
202.5.88.228	212
81.195.245.4	211
202.213.143.7	208
81.196.41.135	193
62.134.85.27	190

TOP 10 Destinations	No. Events
MY.NET.84.228	5088
MY.NET.191.52	1146
MY.NET.152.166	149
MY.NET.111.225	123
MY.NET.110.220	117
MY.NET.110.204	116
MY.NET.110.212	109
MY.NET.110.205	109
MY.NET.110.203	108
MY.NET.72.243	107

TOP 10 Traffic	No. Events
138.89.11.51:4348 -> MY.NET.84.228:139	132
81.196.41.135:3153 -> MY.NET.84.228:139	121
202.145.95.209:2180 -> MY.NET.84.228:139	121
219.76.177.102:57803 -> MY.NET.84.228:139	119
61.223.139.116:3741 -> MY.NET.84.228:139	114
61.149.22.158:1030 -> MY.NET.191.52:139	114
213.139.208.2:1781 -> MY.NET.191.52:139	114
66.126.31.60:3724 -> MY.NET.84.228:139	110
200.138.171.221:1916 -> MY.NET.84.228:139	108
81.195.245.4:2814 -> MY.NET.84.228:139	107

  To stop such traffic, I recommend the same solution as previous alert “SMB Name Wildcard”.

- **MY.NET.30.4 activity**
  There were (15606) events logged for this alert type, that covers (1.11%) of total alerts. Alerts show (474) unique external IP addresses send traffic to “MY.NET.30.4”. Two tables are listed as following: List of top 10 sources, top 10 destination ports.

TOP 10 Sources	No. Events
68.55.85.180	2934
68.54.91.147	2743
172.142.110.232	1124
151.196.19.202	997
68.33.10.149	474
68.55.62.79	441
68.55.205.180	440
68.84.131.246	396
151.196.34.226	365
151.196.42.116	351

TOP 10 Destination Ports	No. Events
MY.NET.30.4:51443	10378
MY.NET.30.4:80	3901
MY.NET.30.4:524	1210
MY.NET.30.4:135	30
MY.NET.30.4:445	17
MY.NET.30.4:554	8
MY.NET.30.4:139	6
MY.NET.30.4:4000	5
MY.NET.30.4:21	5
MY.NET.30.4:9090	3

There are more ports targeted such as: 8888, 8080, 8000, 8081, 110, 23 … etc, so it is not hard to see that “MY.NET.30.4” host faced a scan or attack processes. There is – though- an expected normal traffic among these scanning activities, unless this host is just a honey pot, and for that reason its traffic had been logged.

Ignoring honey pot theory, and inspecting these activities and targeted ports –Which are selective-, we can tell that there is something on this host that call others. Ports such as: 135, 445, 139 … etc tells us that the system could be Windows. Ports such as: 51443, 524 … etc tells us that the network could be Novell Network. Ports such as: 80, 21, 8080 … etc tells that the host provides services [Web server, Proxy, Ftp …etc].

Logged traffic does not show any traffic going from the targeted host, so we can not say for sure if this system is compromised or not, but it is a good idea to check it.

DNS query tell us that this IP address “130.85.30.4” belongs to UMBC University, and host name is: lan2.umbc.edu. There is another host in the list which is “lan1.umbc.edu”, and its address is “130.85.30.3”.

- **EXPLOIT x86 NOOP**

There were (11563) events logged for this type that covers (0.83%) of total alerts. This event is generated when an unusual large number of consecutive NOOP instructions (NO Operation) -used in padding code- have been detected. This method is usually used to exploit a buffer overflow by filling an address space with a large number of NOOPs, followed by an arbitrary code, to be –then- executed. (1418) unique external IP addresses sent the traffic to (952) unique internal IP addresses belong to the university network (MY.NET).

Three tables are listed as following: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
209.6.97.168	764
24.87.153.94	418
216.232.208.22	412
195.110.140.66	314
63.229.211.22	280
4.34.198.112	260
217.82.34.195	253
4.3.6.237	246
207.9.129.85	243
130.67.101.88	242

TOP 10 Destinations	No. Events
MY.NET.15.198	375
MY.NET.27.103	366
MY.NET.5.95	235
MY.NET.80.16	200
MY.NET.81.18	190
MY.NET.29.2	160
MY.NET.66.53	149
MY.NET.189.62	85
MY.NET.5.15	73
MY.NET.69.175	72

TOP 10 Traffic	No. Events
209.6.97.168:1975 → MY.NET.29.2:445	159
209.6.97.168:3747 → MY.NET.15.198:445	149
209.6.97.168:3668 → MY.NET.66.53:445	149
209.6.97.168:2886 → MY.NET.27.103:445	149

24.87.153.94:3544 → MY.NET.80.16:445	123
24.87.153.94:4617 → MY.NET.81.18:445	101
217.208.53.187:2390 → MY.NET.15.198:445	100
209.6.97.168:4311 → MY.NET.81.18:445	82
217.208.53.187:2284 → MY.NET.27.103:445	77
209.6.97.168:2225 → MY.NET.80.16:445	70

False positives can be seen often, because there are many applications (i.e. http, ftp) can generate similar pattern specially when transferring large files. False positive is possible, but I think this is not the case here, since the targeted port was –mostly- “445”, which is one of the ports that are used to exploit a buffer overflow in a Windows Distributed Component Object Model (DCOM) Remote Procedure Call (RPC).

- **Connect to 515 from inside**

  There were (7126) events logged for this type, that covers (0.51%) of total alerts. Port (515) is used by the printer service. Some worms like “Code Red” scan for such service to exploit it and others, so this alert is generated when ever there is an attempt to connect to outside on print service, and that helps to discover any probable infected insiders. Only one internal source “MY.NET.162.41” was detected (7126) times trying to connect to external destination “128.183.110.242” on port “515”. Host “MY.NET.162.41” is most probably compromised, so it has to be disconnected from the network, and checked urgently. DNS query tell us that this IP address “130.85.162.41”, has the following name: physics422-laptop.umbc.edu, which appeared to be a laptop.

  To resolve this matter, I recommend the following:
  - Block any traffic from inside to port 515.
  - Install Anti-Virus program in all systems (UNIX, Windows), patched them.
  - Install network Anti-Virus to protect the network.
  - Install IDS.

- **MY.NET.30.3 activity**

  There were (5727) events logged for this alert, that covers (0.41%) of total alerts. Alerts show (100) external IP addresses sent traffic to “MY.NET.30.3”. Two tables are listed as following: List of top 10 sources, top 10 destination ports.

TOP 10 Sources	No. Events
68.57.90.146	1224
68.55.27.157	735
68.55.233.51	639
68.55.62.79	605
141.157.6.106	572
68.55.105.5	462
68.55.53.222	209
68.55.250.229	200
68.48.217.68	107
165.247.97.243	101

TOP 10 Destination Ports	No. Events
MY.NET.30.3:524	5607
MY.NET.30.3:135	28
MY.NET.30.3:80	17
MY.NET.30.3:445	12
MY.NET.30.3:554	8
MY.NET.30.3:4000	8
MY.NET.30.3:21	6
MY.NET.30.3:139	3
MY.NET.30.3:9090	2
MY.NET.30.3:5128	2

The same thing that we said about “MY.NET.30.4” we say here, so there is no need to repeat it again. The targeted ports are mostly similar, and most attack sources attacked “MY.NET.30.4” too.

Logged traffic does not show any traffic going from the targeted host, so we can not say for sure if this system is compromised or not, but it is a good idea to check it.

DNS query tell us that this IP address “130.85.30.3” belongs to UMBC University, and host name is: “lan1.umbc.edu”.

- **TCP SRC and DST outside network**

  There were (4518) events logged for this alert, that covers (0.32%) of total alerts. This alert is triggered when both source and destination IP addresses are external, which is not normal. Traffic would not pass within a network unless one or both addresses (source & destination) are internal, from within that network.

  We have three scenarios:

  - The source is internal and destination is external → (Outgoing Traffic).
  - The source is external and destination is internal → (Incoming Traffic).
  - The source is external and destination is external → how, Two answers?
    - Internal Hosts may spoof an external address and use it as a source. Why? This should not happen on legitimate traffic. Any system does that could compromised.
    - There is a mistake on the routing table? Causes unrelated traffic to end up on the internal network.

  For the Alert log, we found (27) unique external IP addresses sent traffic to (111) unique external IP addresses and the following three tables give more details: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
169.254.244.56	4279
68.55.0.64	78
10.0.1.12	42
192.168.1.101	28
192.168.0.5	23
66.93.118.119	11
192.168.2.85	10
192.168.0.101	8
68.55.50.36	4
65.118.41.150	4

TOP 10 Destinations	No. Events
218.16.124.131	2854
211.91.144.72	1420
68.55.61.253	42
63.211.66.115	14
66.93.118.125	11
17.250.248.64	10
63.251.80.206	8
204.221.192.173	7
64.12.24.62	5
152.163.14.25	4

TOP 10 Traffic	No. Events
10.0.1.12:49291 → 68.55.61.253:143	23
10.0.1.12:49289 → 68.55.61.253:143	19
169.254.244.56:2442 → 218.16.124.131:21	13
169.254.244.56:2769 → 218.16.124.131:21	12
169.254.244.56:2494 → 218.16.124.131:21	12
169.254.244.56:2478 → 218.16.124.131:21	12
169.254.244.56:2447 → 211.91.144.72:996	12
169.254.244.56:2440 → 218.16.124.131:21	12
169.254.244.56:2428 → 211.91.144.72:996	12
169.254.244.56:2423 → 218.16.124.131:21	12

  As a defense recommendation, it is necessary to apply Anti spoofing Access-List (ACL) on router interfaces in both direction.

- **External RPC call**

  There were (3266) events logged for this alert type that covers (0.23%) of total alerts. Remote Procedure Call (RPC) is a protocol used to request a service (such as Network File System "NFS") located in another system. It is similar to programming subroutine call but via Client-Server network criteria. RPC service listen on port 111, but then wide range of ports can be assigned for a particular running program. There are many security wholes related to RPC service that can be used to gain root access to a system by exploiting a buffer overflow, and executing an arbitrary code on it. This alert message is generated whenever external attempt exists to connect to internal network on port "111". The false positive alarms would be many if you provide for external networks a public service that uses RPC protocol.

According To Alert files (4) unique external IP addresses attempted to connect to (1831) unique internal IP addresses on port “111”, and the following two tables give more details: List of top sources and top 10 destinations.

TOP 10 Sources	No. Events
193.114.70.169	2837
81.15.45.1	420
166.102.99.229	7
64.209.74.229	2

TOP 10 Destinations	No. Events
MY.NET.24.65	18
MY.NET.6.15	8
MY.NET.28.9	6
MY.NET.75.140	5
MY.NET.60.172	5
MY.NET.55.118	5
MY.NET.28.8	5
MY.NET.28.12	5
MY.NET.28.11	5
MY.NET.28.10	5

As a defense recommendation, it is recommended -unless you provide a public service that uses such protocol- to block any traffic from external networks to internal network on port “111”.

- **High port 65535 TCP - possible Red Worm – traffic**

  There were (3172) events logged for this alert type, that covers (0.23%) of total alerts. Code Red worm propagate itself by attempting to exploit a Buffer Overflow vulnerability in Microsoft web server Internet Information Services (IIS) Indexing Service DLL running on Windows NT, Windows 2000, and beta versions of Windows XP. Code Red worm looks for IIS by scanning for port “80”. Alert files showed that this alert is generated whenever there is a packet with high port “65535” either as a source or destination. Targeted ports are varying: 80, 25, 21 …etc, which indicate that there are false positive alarms, because the Code Red worm scans for just port “80” -According to its description-. There is another reason to have false positives, because relying on the ports itself to classify the traffic is not perfect, especially when it is very possible to have a legitimate traffic that could match attack signature. Referring to Internet Storm Center [http://isc.incidents.org], there are incidents –under port “65535”- detected and classified as Trojans, so after all the traffic that has been logged under this alert type ”High port 65535 TCP - possible Red Worm – traffic” ended to be bad traffic, but it is not necessary a Code Red worm activities.

  Alert files show (102) unique sources (External & Internal), and (116) unique destinations (External & Internal) match the alert pattern, and the following three tables give some details: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
MY.NET.80.105	1113
200.96.13.157	1023
MY.NET.153.141	310
66.66.71.92	284
MY.NET.24.20	24
MY.NET.53.44	23
MY.NET.24.44	23
MY.NET.100.230	18
MY.NET.24.33	17
202.156.254.68	15

TOP 10 Destinations	No. Events
200.96.13.157	1112
MY.NET.80.105	1022
66.66.71.92	320
MY.NET.153.141	268
202.156.254.68	23
198.86.10.116	18
127.0.0.2	18
MY.NET.24.44	17
24.35.71.146	17
MY.NET.53.44	15

TOP 10 Traffic	No. Events
MY.NET.80.105:3951 -> 200.96.13.157:65535	1112
200.96.13.157:65535 -> MY.NET.80.105:3951	1022
MY.NET.153.141:2071 -> 66.66.71.92:65535	309

66.66.71.92:65535 -> MY.NET.153.141:2071	268
MY.NET.53.44:1265 -> 202.156.254.68:65535	23
MY.NET.24.44:80 -> 198.86.10.116:65535	18
MY.NET.100.230:65535 -> 127.0.0.2:25	18
MY.NET.24.33:443 -> 24.35.71.146:65535	17
202.156.254.68:65535 -> MY.NET.53.44:1265	15
198.86.10.116:65535 -> MY.NET.24.44:80	13

From above tables the following addresses need to be audited, and if necessary be patched or reinstalled: MY.NET.80.105, MY.NET.153.141, MY.NET.53.44, MY.NET.24.44 and MY.NET.100.230. As a general defense recommendation, all your systems must be frequently patched.

- **Possible Trojan server activity**

  There were (2009) events logged for this alert type that covers (0.14%) of total alerts. This alert is generated when ever port “27374” –used by Sub Seven Trojan horse- existed on the traffic, which indicates backdoor activities, especially when there is a response from a system listening on that port.

  (87) unique sources (External & Internal) and (327) unique destinations (External & Internal) were detected using or sending traffic to port “27374”. The three following tables give some details: List of top 10 sources, top 10 destinations and top 10 traffic/session.

TOP 10 Sources	No. Events
200.163.61.175	553
MY.NET.163.249	409
66.169.146.100	303
212.95.105.31	71
67.64.149.135	63
67.121.127.74	63
24.35.69.248	61
68.50.99.13	60
24.211.143.10	54
24.199.192.33	44

TOP 10 Destinations	No. Events
MY.NET.163.249	560
200.163.61.175	402
MY.NET.12.6	29
MY.NET.24.34	21
64.41.183.130	18
66.169.146.100	15
MY.NET.24.74	14
MY.NET.5.20	11
200.30.141.234	10
12.167.138.125	9

TOP 10 Traffic	No. Events	Comments
200.163.61.175:27374 -> MY.NET.163.249:6667	553	Ports “27374” & “6667” → BAD
MY.NET.163.249:6667 -> 200.163.61.175:27374	402	=
64.41.183.130:27374 -> MY.NET.12.6:25	20	If “MY.NET.12.6” is a mail server, then this could be a legitimate traffic.
MY.NET.12.6:25 -> 64.41.183.130:27374	18	=
MY.NET.29.3:80 -> 200.30.141.234:27374	10	If “MY.NET.29.3” is a web server, then this could be a legitimate traffic.
MY.NET.24.74:443 -> 12.167.138.125:27374	9	Host name is “webmail.umbc.edu”, so this is could be a legitimate traffic.
128.121.26.2:27374 -> MY.NET.12.6:25	9	If “MY.NET.12.6” is a mail server, then this could be a legitimate traffic.
151.196.48.182:27374 -> MY.NET.24.33:443	8	If “MY.NET.29.3” is a web server, then this could be a legitimate traffic.
MY.NET.163.249:6667 -> 200.203.68.71:27374	7	Ports “27374” & “6667” → BAD
207.214.63.10:27374 -> MY.NET.24.74:443	7	Host name is “webmail.umbc.edu”, so this is could be a legitimate traffic.

For Protection, I recommend the following:

- The following internal addresses should be inspected immediately to verify its situation: MY.NET.163.249, MY.NET.12.6, MY.NET.29.3, MY.NET.24.74 and MY.NET.24.33.

  - Audit all systems: Patch them, unnecessary system listening ports should stop, any unnecessary running process/daemon should stop.

- **FTP passwd attempt**
  There were (105) events logged for this alert type. This alert is generated when ever an attempt is made to download a copy of the "passwd" file from an FTP server. (35) unique external IP addresses have been detected attempting to download "passwd" file from (6) unique internal addresses. The following two tables list the top sources and destinations.

TOP 10 Sources	No. Events		TOP 10 Destinations	No. Events
212.202.168.140	63		MY.NET.24.47	42
199.243.85.90	3		MY.NET.24.13	18
12.47.47.2	3		MY.NET.24.9	17
63.119.136.67	2		MY.NET.24.27	12
192.223.163.5	2		MY.NET.24.19	9
156.153.255.236	2		MY.NET.24.18	7
128.38.137.51	2			
68.8.130.38	1			
68.48.166.121	1			
68.42.146.143	1			

  I recommend the following:
  - Check files/directories permissions.
  - Check FTP server configuration.
  - Update the ftp software version in case to resolve any bug that can be exploited.

- **NIMDA - Attempt to execute cmd from campus host**
  There were (4) events logged under this alert type. One method of this worm to propagate is to scan for IIS web servers, and tries to infect them. Later on, any web surfers are going to be infected once they browse those infected web servers. This alert is generated whenever internal hosts are trying to infect other IIS systems. These internal sources are themselves infected, and they are trying to infect others. These are the (4) logged events:

(4) Logged events
MY.NET.70.176:1135 -> 64.4.25.188:80
MY.NET.75.103:1086 -> 206.24.190.77:80
MY.NET.97.150:1087 -> 64.4.25.188:80
MY.NET.97.228:2312 -> 66.240.1.50:80

  For protection, I recommend and require the following:
  - Disconnect all these internal sources from the network immediately, and need to be reinstalled and patched.
  - All systems that these internal sources are able to connect to, needs to be inspected too.
  - Do the following on each system:
    - Install host Anti Virus software on each system.
    - Configure your browsing program securely.
  - Install Anti Virus software on the mail server.
  - Use a content filter web proxy that will detect any worm/web viruses, and block it immediately.

- **Connect to 515 from outside**
  Only (2) events were detected for this alert. This is port "515" that is used –as we said- by the print service. Current alert is just the opposite of the previous alert "Connect to 515 from inside". Only, one external source "64.209.74.229" tried to connect to two internal destinations: "MY.NET.60.14"and "MY.NET.24.44". DNS query tell us destination host names as following:

- 130.85.60.14 → www.gl.umbc.edu. (Web server).
- 130.85.24.44 → userpages.umbc.edu (Web server).

To resolve this matter, I recommend the following:

- Block any traffic from outside to port 515.
- Patch all systems.
- Install network Anti-Virus to protect the network.
- Install IDS.

Summary of Scans

We have two information sources that help to gather and analyse scanning activities, and so I would like to brief each one, and then draw a conclusion.

Scans log files

Scans log files contain the scan data from the port scan pre-processor. (11,699,747) events were logged in these files, and only TCP and UDP were detected. A brief of them are following.

- TCP events

  (8591446) were TCP events. (4638) unique sources (internal & external addresses) scanned (4877806) unique destinations (internal & external). (41150) different ports have been targeted, and (140) different TCP scan techniques have been used. The following three tables list the top 10 sources, top 10 destinations, top 10 destination ports, and top TCP scan techniques.

TOP 10 Sources	No. Events
130.85.70.154	1294171
130.85.163.107	966592
130.85.84.194	886857
130.85.163.249	668901
130.85.42.1	273689
130.85.70.129	213577
130.85.80.149	175933
130.85.111.72	171526
130.85.73.94	141393
130.85.72.163	106250

TOP 10 Destinations	No. Events
130.85.15.27	30276
130.85.24.34	18819
130.85.111.52	2500
130.85.97.104	2181
130.85.12.6	1440
130.85.24.44	1147
130.85.69.181	756
130.85.60.14	723
209.242.23.200	552
130.85.100.165	539

Top 10 Destination Ports	No. Events	Comments
135	5808279	end-point map per (epmap) Microsoft DCE Locator service, **Microsoft** Exchange server
80	2048358	http
445	211495	Server Message Block
4000	72892	Sky dance DDoS Trojan
554	63766	Real Time Stream Control Protocol (RTSP)
21	44991	FTP
139	44314	NETBIOS Session Service
1433	17673	Microsoft-SQL-Server
3389	16246	MS Terminal Services
443	15668	https
6346	15549	BearShare file sharing application, gnutella file sharing application

Top TCP scan techniques	No. Events
SYN ******S*	8577404
SYN 12****S*	6639
FIN *******F	2707

INVALIDACK ***A*R*F	2386
UNKNOWN 1**A*R**	1276
NULL ********	343
UNKNOWN 1****R**	70
UNKNOWN *2*A**S*	65
UNKNOWN *2*A****	62
INVALIDACK ***AP*S*	57
NOACK *****RS*	44

- UDP events

  (3108301) were UDP events. (630) unique sources (internal & external addresses) scanned (368425) unique destinations (internal & external addresses). The following three tables list the top 10 sources, top 10 destinations and top 10 destination ports.

TOP 10 Sources	No. Events
130.85.1.3	2164273
130.85.1.5	211077
130.85.70.176	76452
130.85.84.232	61035
130.85.112.159	59506
130.85.153.94	49522
130.85.84.143	48774
130.85.80.51	40636
130.85.42.5	38150
130.85.98.28	36247

TOP 10 Destinations	No. Events
192.26.92.30	57085
192.55.83.30	43945
203.20.52.5	32455
130.94.6.10	32276
204.152.186.189	26947
131.118.254.33	26036
216.109.116.17	25707
131.118.254.34	24599
131.118.254.35	23570
205.231.29.244	19972

Top 10 Destination Ports	No. Events	Comments
53	2367138	DNS Resolving port
22321	132539	Wnn6 (Taiwanese input), probably a file sharing application port
6257	109499	WinMX File Sharing
137	93325	NETBIOS Name Service
7674	50813	Probably a file sharing application port
4673	32677	Probably a file sharing application port
41170	24070	Blubster File Sharing
0	10946	System Reserved port
123	7764	Net Controller Trojan / Network Time Protocol
4672	4867	remote file access server
6112	4613	CDE Sub process Control Service (dtspcd), Free Standard Game Server (FSGS)

Alerts log files

Three alert types have been detected and classified by me as scan activities. It belongs to scanning process, and so they brought here. The three alerts are: “spp_portscan”, “NMAP TCP ping!” and “Null scan!”

- spp_portscan

  (1114429) events were detected and (4844) unique sources (internal & external addresses) are existed. This alert is generated by “ssp_portscan” pre-processor to record any start, Finish, Status of scan activities. The following table list the top sources.

Top 10 Sources	No. Events
MY.NET.1.3	200840
MY.NET.163.107	117572
MY.NET.84.194	111670
MY.NET.70.154	76081
MY.NET.163.249	69350
MY.NET.1.5	23943
MY.NET.111.72	20920
MY.NET.73.94	16876
MY.NET.42.1	15485
MY.NET.70.129	13822
MY.NET.70.176	12804

- NMAP TCP ping!

(752) events were detected. The alert is generated when a TCP packet with Flags: A, and Ack 0 is detected. (150) unique external IP addresses scanned (59) unique internal IP addresses. The following two tables list the top sources & destinations.

TOP 10 Sources	No. Events
64.152.70.68	192
63.211.17.228	176
216.5.176.162	47
205.244.232.133	38
208.155.15.101	10
65.246.192.101	9
12.146.100.101	7
81.255.54.252	6
206.102.126.101	6
195.77.24.2	6

TOP 10 Destinations	No. Events
MY.NET.1.3	367
MY.NET.1.5	71
MY.NET.12.4	64
MY.NET.100.165	38
MY.NET.24.44	26
MY.NET.24.34	20
MY.NET.12.6	20
MY.NET.6.7	14
MY.NET.24.74	14
MY.NET.1.4	13

- Null scan!

(455) events were detected. This alert is generated when a TCP packet with none of its control bits (URG, ACK, PSH, RST, SYN, and FIN) is detected. Additionally, both the sequence number and acknowledgement number were set to 0 (flags:0; seq:0; ack:0). (60) unique external IP addresses scanned (56) unique internal IP addresses. The following two tables list the top sources & destinations.

TOP 10 Sources	No. Events
80.134.197.231	150
63.251.52.75	107
67.119.234.194	43
67.119.232.52	42
200.78.0.1	14
210.196.81.10	10
206.14.191.84	9
217.112.233.50	8
213.225.61.162	7
218.17.70.52	6

TOP 10 Destinations	No. Events
MY.NET.69.181	150
MY.NET.12.4	85
MY.NET.12.6	26
MY.NET.97.35	22
MY.NET.97.73	18
MY.NET.53.31	18
MY.NET.53.209	13
MY.NET.153.211	12
MY.NET.84.143	11
MY.NET.153.197	8

Scans Conclusion

- Top scanners are internal IP addresses.
- Top two UDP scanners are the top two "NMAP TCP ping!" destination addresses?! Host "130.85.1.3" → "umbc3.umbc.edu", and host "130.85.1.5" → "umbc5.umbc.edu" are required inspection.

- Top destination scanned ports are following: Windows RPC DCOM ports, SMB services ports, HTTP, DNS, Microsoft-SQL-Server, Trojans ports, File Sharing ports,
- This huge amount of outgoing scan activities, and incoming activities as well raise questions about the integrity of some internal hosts (Specially the top internal scanners), and protection methods that are in place.

Out Of Specification events Overview

OOS log files include packets that violate protocol standards. Files events are –partially- covered on other various log files (i.e. Scans, Alerts). (21,319) events were logged, where (652) unique source addresses sent Out of Specification packets to (162) unique destination addresses. (186) different ports have been targeted, and (51) TCP options have been used. The following three tables list the top 10 sources, top 10 destinations, top 10 destination ports, and top TCP used options.

TOP 10 Sources	No. Events	TOP 10 Destinations	No. Events
217.174.98.145	1142	MY.NET.111.52	7867
195.111.1.93	1130	MY.NET.12.6	4115
212.16.0.33	1038	MY.NET.100.165	1672
158.196.149.61	973	MY.NET.69.181	1504
194.67.62.194	792	MY.NET.24.44	1407
82.82.64.209	685	MY.NET.75.240	839
213.23.46.99	682	MY.NET.84.143	734
195.208.238.143	472	MY.NET.24.34	471
195.14.47.202	454	MY.NET.100.230	327
200.77.250.50	437	MY.NET.6.7	282

Top 10 Destination Ports	No. Events	Comments
25	13447	SMTP
80	4194	HTTP
8887	1489	HTTP, Messenger
4662	1255	eDonkey2000 file sharing app.
113	406	Identd, Kazimas (Trojan)
110	246	Pop3, ProMail Trojan
1214	90	KAZAA file sharing app
6881	56	Bit Torrent P2P
6883	41	Delta Source DarkStar Trojan
443	26	HTTPS
3264	26	ccmail

Top TCP used options	No. Events
12****S*	21255
********	296
****P***	122
12***R**	37
12*A*R**	15
**U*P*SF	7
***A**SF	7
***AP*SF	4
*2UAPRSF	3
*2U***SF	2
*2***RSF	2

▪ Top Talkers

I thought it is good to find the top talkers for each file type, and then I will pick the first 4 Alerts top talkers, the first 3 Scans top talkers and the first 3 OOS top talkers to talk about them. My criteria -at each log type- are to find the top most frequent sources that have the maximum number of detected events.

Alerts

The following are the list of top 10 Alerts talkers, but after excluding scan alerts [spp_portscan, NMAP TCP ping! and Null scan!].

Top 10 Talkers	No. Events	Alerts
MY.NET.80.51	115624	SMB Name Wildcard
MY.NET.150.133	72067	SMB Name Wildcard
MY.NET.162.41	7130	connect to 515 from inside (7126), SMB Name Wildcard (4)
169.254.244.56	4279	TCP SRC and DST outside network
MY.NET.29.2	3100	SMB Name Wildcard
68.55.85.180	2934	MY.NET.30.4 activity
193.114.70.169	2890	External RPC call (2837), NETBIOS NT NULL session (50), MY.NET.30.4 activity (3)
68.54.91.147	2743	MY.NET.30.4 activity
MY.NET.84.224	1291	SMB Name Wildcard
68.57.90.146	1251	MY.NET.30.3 activity (1224), MY.NET.30.4 activity (27)
172.142.110.232	1124	MY.NET.30.4 activity

Scans

Top 10 Talkers	No. Events
130.85.1.3	2164273
130.85.70.154	1294171
130.85.163.107	966592
130.85.84.194	886857
130.85.163.249	668901
130.85.42.1	273689
130.85.70.129	213577
130.85.80.149	175933
130.85.111.72	171526
130.85.73.94	141393
130.85.72.163	106250

OOS (Out of Spec)

The same Top 10 sources list as it appears above in "Out Of Specification events Overview" section. There is no need to just replicate it again.

Top Talkers Conclusion list

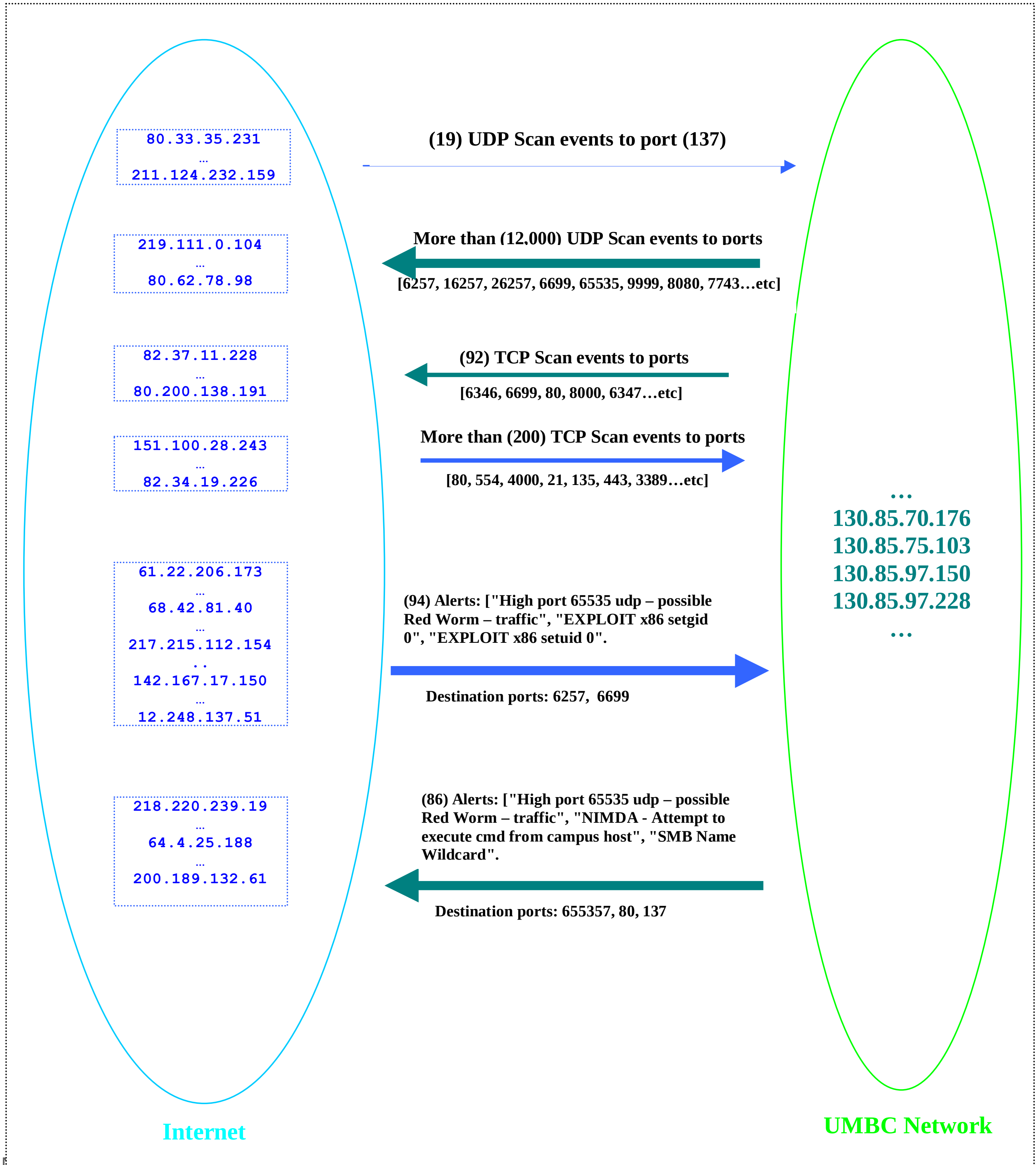
To be fair enough, I gave a chance of different log types to be involved on my Top-Talkers conclusion list that would reflect a real picture of attacks under analysis. I picked the first 4 Alerts top talkers, the first 3 of others. I sort them according to No. of events.

Top 10 Talkers	No. Events	Log Type
130.85.1.3	2,164,273	Scans
130.85.70.154	1,294,171	Scans
130.85.163.107	966,592	Scans
130.85.80.51	115,624	Alerts
130.85.150.133	72,067	Alerts
130.85.162.41	7,130	Alerts

169.254.244.56	4,279	Alerts
217.174.98.145	1,142	OOS
195.111.1.93	1,130	OOS
212.16.0.33	1,038	OOS

- **Internal Infected hosts -With Nimda- related Traffic Analysis (Link Graph is included)**

We have at least four infected internal hosts with "Nimda" worm. We will gather all traffic sent to or from these four hosts: [130.85.70.176, 130.85.75.103, 130.85.97.150 and 130.85.97.228]. I summarized that traffic in a graph, and then I analyzed it below.


80.33.35.231
…
211.124.232.159
219.111.0.104
…
80.62.78.98
82.37.11.228
…
80.200.138.191
151.100.28.243
…
82.34.19.226
61.22.206.173
…
68.42.81.40
…
217.215.112.154
..
142.167.17.150
…
12.248.137.51
218.220.239.19
…
64.4.25.188
…
200.189.132.61
(19) UDP Scan events to port (137)
More than (12,000) UDP Scan events to ports
[6257, 16257, 26257, 6699, 65535, 9999, 8080, 7743…etc]
(92) TCP Scan events to ports
[6346, 6699, 80, 8000, 6347…etc]
More than (200) TCP Scan events to ports
[80, 554, 4000, 21, 135, 443, 3389…etc]
(94) Alerts: ["High port 65535 udp – possible Red Worm – traffic", "EXPLOIT x86 setgid 0", "EXPLOIT x86 setuid 0".
Destination ports: 6257, 6699
(86) Alerts: ["High port 65535 udp – possible Red Worm – traffic", "NIMDA - Attempt to execute cmd from campus host", "SMB Name Wildcard".
Destination ports: 655357, 80, 137
…
130.85.70.176
130.85.75.103
130.85.97.150
130.85.97.228
…
Internet
UMBC Network

- The four Internal UMBC hosts [130.85.70.176, 130.85.75.103, 130.85.97.150 and 130.85.97.228] have been active, and most probably compromised. The following give some details.
- Four internal hosts scanned heavily external destination addresses on different ports (TCP/UDP): 65535 (Varies Trojans), "6257" (WinMX File Sharing), "6699" (File sharing app), "9999" (distinct, or possible Trojan ports), "6347" (gnutella-rtr file sharing app.) …etc.
- Four Internal UMBC hosts have been scanned by external source addresses on different ports: a windows UDP port "137" (NETBIOS service name), and windows RPC DCOM TCP port "135", "3389" (MS Terminal Services), "554" (Real Time Stream Control Protocol), "4000" (Sky dance DDoS Trojan) …etc.
- Four Internal hosts are infected with Nimda worm.
- All or part of the four internal hosts reply back a connection on "6257", which could be just normal file sharing protocol, and it could a Trojan horse, running on that port.
- There are attempts to connect to all or part of the four internal hosts on TCP port "6699". This could be another file sharing app, or a Trojan horse connection attempt/scan.

Registration information of five selected external source addresses

I picked the five external source addresses from the previous "Top Talkers" section. I selected the top 6 external addresses according to the number of events. If -at any time- the current picked external address is part of same network as previous selected external address, I skip it to the next external one, and so on.

- **Host 1 →** "68.55.85.180" (2934) events were logged

  It belongs to "`Comcast Cable Communications, Inc.`". Registered information as following:

```
Comcast Cable Communications, Inc. JUMPSTART-1 (NET-68-32-0-0-1)
                                  68.32.0.0 - 68.63.255.255
Comcast Cable Communications, Inc. BALTIMORE-A-6 (NET-68-55-0-0-1)
                                  68.55.0.0 - 68.55.255.255
```

- **Host 2 →** "193.114.70.169" (2890) events were logged

  It belongs to "`FIRST-PROCUREMENT-ASSOCIATES-LIMITED, GB`". Registered Information as following:

```
inetnum:      193.114.70.160 - 193.114.70.191
netname:      FIRST-PROCUREMENT-ASSOCIATES-LIMITED
descr:        FIRST PROCUREMENT ASSOCIATES LIMITED
country:      GB
admin-c:      JB7221-RIPE
tech-c:       AB480-RIPE
status:       ASSIGNED PA
notify:       ripe-notify@uk.psi.com
mnt-by:       PSINET-UK-SYSADMIN
changed:      sysadmin@uk.psi.com 19990903
source:       RIPE

route:        193.114.0.0/15
descr:        EUNETGB-114-AGG
origin:       AS1290
mnt-by:       PSINET-MNT
changed:      network-ripe@uk.psi.com  20021015
source:       RIPE
```

```
  person:       John Barke
  address:      FIRST PROCUREMENT ASSOCIATES LIMITED
  address:      1St Andrews House
  address:      Vernon Gate
  address:      Derby
  address:      DE1 1UJ
  phone:        +44 1332 604 313
  nic-hdl:      JB7221-RIPE
  notify:       ripe-notify@uk.psi.com
  mnt-by:       PSINET-UK-SYSADMIN
  changed:      sysadmin@uk.psi.com 19990903
  source:       RIPE

  person:       Anthony Bennett
  address:      FIRST PROCUREMENT ASSOCIATES LIMITED
  address:      1St Andrews House
  address:      Vernon Gate
  address:      Derby
  address:      DE1 1UJ
  phone:        +44 1332 604 313
  nic-hdl:      AB480-RIPE
  notify:       ripe-notify@uk.psi.com
  mnt-by:       PSINET-UK-SYSADMIN
  changed:      sysadmin@uk.psi.com 19990903
  source:       RIPE
```

- **Host 3 →** "217.174.98.145"   (1142) events were logged
  It belongs to "Sunet 2000 Ltd. 120 8 Prishvina Moscow, Russia". Registered Information as following:

```
inetnum:      217.174.96.0 - 217.174.98.255
netname:      SUNET2000
descr:        Sunet 2000 Ltd
descr:        120 8 Prishvina Moscow
descr:        Russia
country:      RU
admin-c:      AT4804-RIPE
tech-c:       AT4804-RIPE
rev-srv:      ns.sunet.ru
status:       ASSIGNED PA
mnt-by:       SUNET2000-MNT
changed:      hostmaster@ripe.net 20010411
changed:      andy@sunet.ru 20010618
source:       RIPE

route:        217.174.96.0/21
descr:        SUNET2000
origin:       AS20655
holes:        217.174.103.0/24
mnt-by:       SUNET2000-MNT
changed:      dg@sunet.ru 20020904
changed:      andy@sunet.ru 20030429
changed:      andy@sunet.ru 20030820
changed:      andy@sunet.ru 20031028
source:       RIPE
```

```
person:       Andy E Trushin
```

```
address:       112 41/8 Andropova Stupino Russia
phone:         +7 095 796 9797
phone:         +7 902 693 4286
fax-no:        +7 095 772 7616
e-mail:        andy@sunet.ru
e-mail:        andy@ahome.ru
nic-hdl:       AT4804-RIPE
mnt-by:        SUNET2000-MNT
changed:       crocodil@express.ru 20000714
changed:       tangaldi@express.ru 20010806
changed:       andy@sunet.ru 20030303
source:        RIPE
```

- **Host 4** → "195.111.1.93"   (1130) events were logged

  It belongs to "Computer and Automation Institute of Hungarian Academy of Sciences, HU". Registered Information as following:

```
  inetnum:       195.111.0.0 - 195.111.3.255
  netname:       SZTAKI
  descr:         Computer and Automation Institute
  descr:         of Hungarian Academy of Sciences
  country:       HU
  admin-c:       BM238
  tech-c:        SN375-RIPE
  status:        ASSIGNED PA
  remarks:       hrcode=0805353b2
  notify:        hostmaster@iif.hu
  mnt-by:        NIIF-MNT
  changed:       hostmaster@iif.hu 19980520
  changed:       hostmaster@iif.hu 19991210
  changed:       hostmaster@iif.hu 20010105
  source:        RIPE

  route:         195.111.0.0/16
  descr:         HBONE/HUNGARNET Block 03
  origin:        AS1955
  mnt-by:        AS1955-MNT
  changed:       net-admin@sztaki.hu 19970313
  source:        RIPE

  role:          SZTAKI Netmaster
  remarks:       This object has been updated in an
                 automated process to fix references by
                 names on 20030116. For more information,
                 please see http://www.ripe.net/db/refs-by-name-
cleanup.html
  address:       Hungarian Academy of Sciences
  address:       Computer and Automation Institute
  address:       Victor Hugo u. 18-22.
  address:       H-1132 Budapest
  address:       Hungary
  phone:         +36 1 1497986
  fax-no:        +36 1 1297866
  e-mail:        net-admin@sztaki.hu
  admin-c:       BM238
  tech-c:        Np442-RIPE        # was 'Not public'
  nic-hdl:      SN375-RIPE
  remarks:      netmaster on duty
  mnt-by:       SZTAKI-MNT
  changed:      hostmaster@sztaki.hu 19980202
  changed:      ripe-dbm@ripe.net 20030116
  source:       RIPE

  person:       Balazs Martos
  address:      Computer and Automation Institute
  address:      Hungarian Academy of Sciences
  address:      Victor Hugo u. 18-22.
  address:      H-1132 BUDAPEST
  address:      Hungary
  phone:        +36 1 1497532
  fax-no:       +36 1 1297866
  e-mail:       martos@sztaki.hu
  nic-hdl:      BM238
  notify:       martos@sztaki.hu
  mnt-by:       SZTAKI-MNT
  changed:      horvath@sztaki.hu 19940920
  changed:      hostmaster@sztaki.hu 19980121
  source:       RIPE
```

- **Host 5 →** "212.16.0.33"   (1038) events were logged

It belongs to "Information, Marketing and Telecommunications Center of Moscow State University, RU". Registered Information as following:

```
  inetnum:      212.16.0.0 - 212.16.1.255
  netname:      MSUNETCOM
  descr:        Information, Marketing and Telecommunications Center
of
  descr:        Moscow State University
  country:      RU
  admin-c:      AG1526-RIPE
  tech-c:       MSU1-RIPE
  status:       ASSIGNED PA
  notify:       noc@msu.net
  mnt-by:       MSUNET-MNT
  changed:      milla@msu.ru 20030415
  source:       RIPE

  route:        212.16.0.0/19
  descr:        Moscow State University Network
  descr:        Moscow, Russia
  origin:       AS8592
  notify:       noc@msu.net
  mnt-by:       MSUNET-MNT
  changed:      kent@msu.ru 19980224
  source:       RIPE

  role:         NOC MSUNET TEAM
  address:      Main building, Room 1012
  address:      Moscow State University
  address:      Lenin's Hills
  address:      119899 Moscow
  address:      Russia
```

```
phone:          +007 095 939-2829
fax-no:         +007 095 930-8700
e-mail:         noc@msu.net
admin-c:        AG1526-RIPE
tech-c:         KMS1-RIPE
tech-c:         LE169-RIPE
nic-hdl:        MSU1-RIPE
mnt-by:         MSUNET-MNT
changed:        kent@msu.ru 20020827
changed:        milla@msu.ru 20030403
source:         ripe

person:         Alexander Gladilin
address:        2nd building, Room P6
address:        Moscow State University
address:        Lenin's Hills
address:        119992 Moscow
address:        Russia
phone:          +007 095 939-1601
e-mail:         gladilin@direct.ru
nic-hdl:        AG1526-RIPE
notify:         gladilin@direct.ru
mnt-by:         MSUNET-MNT
changed:        milla@msu.ru 20020826
changed:        milla@msu.ru 20030403
source:         RIPE
```

- **Host 6 →** "158.196.149.61" (973) event were logged
  It belongs to "VSB - Technical University, Ostrava, CZ". Registered Information as following:

```
inetnum:        158.196.0.0 - 158.196.255.255
netname:        TUONET
descr:          VSB - Technical University
descr:          Ostrava
country:        CZ
admin-c:        PT7
tech-c:         MP4055-RIPE
tech-c:         JV6-RIPE
tech-c:         ID330-RIPE
tech-c:         JG1602-RIPE
status:         ASSIGNED PI
mnt-by:         TENCZ-MNT
changed:        ors@Czechia.EU.net 19960804
changed:        er-transfer@ripe.net 20031016
changed:        tkpv@cesnet.cz 20031024
source:         RIPE

route:          158.196.0.0/16
descr:          TUONET
origin:         AS2852
mnt-by:         AS2852-MNT
remarks:        Please report abuse -> abuse@cesnet.cz
remarks:        Network problems -> noc@cesnet.cz
changed:        tkpv@cesnet.cz 20010716
source:         RIPE
```

```
person:       Premysl Tichy
address:      Technical University of Ostrava
address:      Department of Computer Science
address:      17. listopadu 15
address:      Ostrava-Poruba
address:      708 33
address:      The Czech Republic
phone:        +420 596993250
phone:        +420 596993452
phone:        +420 596919352
fax-no:       +420 596919352
e-mail:       Premysl.Tichy@vsb.cz
nic-hdl:      PT7
notify:       notify@ces.net
changed:      ors@Czechia.EU.net 19970224
changed:      tkpv@cesnet.cz 20031024
source:       RIPE

person:       Josef Verich
address:      Technical University of Ostrava
address:      17. listopadu 15
address:      Ostrava - Poruba
address:      708 33
address:      The Czech Republic
phone:        +420 596991257
phone:        +420 596919352
fax-no:       +420 596919352
e-mail:       Josef.Verich@vsb.cz
nic-hdl:      JV6-RIPE
notify:       notify@ces.net
changed:      ors@Czechia.EU.net 19970224
changed:      tkpv@cesnet.cz 20031024
source:       RIPE

person:       Ivan Dolezal
address:      Technical University of Ostrava
address:      17. listopadu 15
address:      Ostrava-Poruba
address:      708 33
address:      The Czech Republic
phone:        +420 596 993479
fax-no:       +420 596 919352
e-mail:       Ivan.Dolezal@vsb.cz
nic-hdl:      ID330-RIPE
notify:       notify@ces.net
changed:      tkpv@cesnet.cz 20031024
source:       RIPE

person:       Jiri Grygarek
address:      Technical University of Ostrava
address:      17. listopadu 15
address:      Ostrava-Poruba
address:      708 33
address:      The Czech Republic
phone:        +420 596 993240
fax-no:       +420 596 919352
e-mail:       Jiri.Grygarek@vsb.cz
```

```
nic-hdl:        JG1602-RIPE
notify:         notify@ces.net
changed:        tkpv@cesnet.cz 20031024
source:         RIPE

person:         Martin Pustka
address:        Technical University of Ostrava
address:        17. listopadu 15
address:        Ostrava-Poruba
address:        708 33
address:        The Czech Republic
phone:          +420 596 993174
fax-no:         +420 596 919352
e-mail:         Martin.Pustka@vsb.cz
nic-hdl:        MP4055-RIPE
notify:         notify@ces.net
changed:        tkpv@cesnet.cz 20031024
source:         RIPE
```

Defensive recommendations

I gathered here in this section, defensive recommendations that would resolve current probably existing wholes, and protect the UMBC network from future bad behave or at least eliminate it. Recommendation steps are following:

- Apply security recommendations that have been explained in “Techniques to Protect the Network” section of part-1 on this practical assignment. If it is not possible to apply them all, then apply at least “Anti Spoofing Techniques” section.
- Because a UMBC is a university network, then it would be possible to follow tighter router ACL configuration strategy. On that way, I would recommend:
  - Do not accept any inbound connection (i.e. initiate connection from outside) to all non public services and hosts.
  - Customise your inbound connection to your public services very well.
  - Filter any connection (Inbound & Outbound) on any port that should not be accepted, and can be used badly, such as:
    - Reserved ports → 0
    - Well known Trojan ports → 4000, 27374, 123, 113, 6883…etc
    - Well known exploited ports → 111, 515, 554, 1433, 3389, 6112 …etc
    - NETBIOS services, SMB, and Windows RPC DCOM ports → 135 TCP/UDP, 137 TCP/UDP, 138 TCP/UDP, 139 TCP/UDP, 445 TCP/UDP, and 593 TCP.
    - File sharing application ports → 1214, 4672, 6346, 22321, 6257, 4662, 7674, 4673, 41170 …etc
  - Any exceptional case has to be treated on a very specific manner.
- Install IDS systems to monitor the network, and probably response specific cases such as DoS attacks.
- Install Antivirus software as a mail server based, or host based tools.
- Patch the systems (UNIX & WINDOWS) and keep them patched.
- Update –periodically- your services software to the latest version.
- Follow security configuration recommendation guides when configuring your systems (UNIX & WINDOWS).
- Audit your network periodically.
- There are most probably internal compromised hosts have been detected, or at least need to be inspected for suspicious activities, such as: 130.85.1.3, 130.85.1.5, 130.85.30.3, 130.85.30.4, 130.85.162.41, .130.85.80.105, 130.85.153.141, 130.85.53.44, 130.85.163.249, 130.85.12.6, 130.85.29.3, 130.85.24.74, 130.85.24.33, 130.85.24.47, 130.85.24.13, 130.85.24.9, 130.85.24.27,

130.85.24.19, 130.85.24.18, 130.85.70.176, 130.85.75.103, 130.85.97.150, 130.85.97.228, 130.85.60.14, 130.85.24.44 …etc.

Description of Analysis Process

I like to list my analysis process in steps, and I think it would be more sense, and easier to understand:

1. First I Read, and tried to understand the assignment, and its requirements. As a part of that I referred to GCIA Study Guide version 3.3 [http://www.giac.org/gcia_study_guide_v33.pdf], and to the following students' practical:
   - Paul M Young, GCIA #0603, http://www.giac.org/practical/GCIA/Paul_Young_GCIA.pdf
   - Les M Gordon, GCIA # 0586, , http://www.giac.org/practical/GCIA/Les_Gordon_GCIA.doc
   - Al Maslowski-Yerges, GCIA # 0608, http://www.giac.org/practical/GCIA/Al_Maslowski-Yerges_GCIA.pdf
2. To know what information pieces I need to gather, and to organise my work, I wrote down the required sections that would cover all analysis aspects.
3. I picked up for each file type enough log files (5 consecutive days) from the following path http://www.incidents.org/logs/]. For the ease of the analysis, I joined/concatenated each 5 log files into one single log file, and ended with three inclusive files -one for each type- : Alerts, Scans, and OOS. I scanned these log files content.
4. I extracted scan related alerts from Alerts log type files and joined them in analysis with Scans log type files.
5. To analyse the logs, I looked for tools that would help me to give a summary report, and any required detailed information. I collected these information, analysed them, and then I wrote this report. Used tools are Following:
   - SnortSnarf, Homepage: [http://www.silicondefense.com/software/snortsnarf/]
   - UNIX tools such as: AWK, GREP, SORT, UNIQ, SED, and CUT. I had a useful help by referring to the following students' practical:
     - Calabrese, Chris. http://www.giac.org/practical/Chris Calabrese_GCIA.zip
     - Baker, Chris. http://www.giac.org/practical/Chris_Baker_GCIA.zip

Some of UNIX commands that have been used to gather information are as following:

```
=================
Alerts (Non Scan)
=================

# Top Talkers
grep "\[\*\*\]" ../Alerts/alert.cat | grep -vf scan-alerts | cut -d \] -f 3 | cut -d \- -f 1 | sed s/:.*//
| sort | uniq -c | sort -gro alert.sources.uniq.sorted


=============
Scan Analysis
=============

# "ssp_portscan" alert Information
grep "spp_portscan" alert.cat | awk '{ if($7 ~ /from/) print $8; else print $7; next}' | cut -d : -f 1 |
sort | uniq -c | sort -gro spp_portscan-source.uniq.sorted

# "nmap-ping" alert Information.
grep "NMAP TCP ping\!" alert.cat | cut -d \] -f 3 | cut -d \- -f 1 | sed s/:.*// | sort | uniq -c | sort -
gro nmap-ping-source.uniq.sorted

grep "NMAP TCP ping\!" alert.cat | cut -d \] -f 3 | cut -d \> -f 2 | sed s/:.*// | sort | uniq -c | sort -
gro nmap-ping-dest.uniq.sorted

# "null scan" alert Information
grep "Null scan\!" alert.cat | cut -d \] -f 3 | cut -d \- -f 1 | sed s/:.*// | sort | uniq -c | sort -gro
null-scan-source.uniq.sorted

grep "Null scan\!" alert.cat | cut -d \] -f 3 | cut -d \> -f 2 | sed s/:.*// | sort | uniq -c | sort -gro
null-scan-dest.uniq.sorted



UDP Information (Scans log files)
---------------------------------
grep UDP scans.cat | cut -d \  -f 4 | sed s/:.*// | sort | uniq -c | sort -gro udp-source.uniq.sorted

grep UDP scans.cat | cut -d \> -f 2 | sed s/:.*// | sort | uniq -c | sort -gro udp-dest.uniq.sorted

grep UDP scans.cat | cut -d \> -f 2 | cut -d \: -f 2 | awk {'print $1'} | sort | uniq -c | sort -gro udp-
dest-port.uniq.sorted


TCP Information (Scans log files)
---------------------------------
grep -v UDP scans.cat | cut -d \  -f 4 | sed s/:.*// | sort | uniq -c | sort -gro tcp-source.uniq.sorted

grep -v UDP scans.cat | cut -d \> -f 2 | sed s/:.*// | sort | uniq -c | sort -gro tcp-dest.uniq.sorted

grep -v UDP scans.cat | cut -d \> -f 2 | cut -d \: -f 2 | awk {'print $1'} | sort | uniq -c | sort -gro
tcp-dest-port.uniq.sorted


============
OOS Analysis
============

grep "\->" OOS_Report.cat | cut -d \  -f 2 | sed s/:.*// | sort | uniq -c | sort -gro OOS-
source.uniq.sorted

grep "\->" OOS_Report.cat | cut -d \>  -f 2 | sed s/:.*// | sort | uniq -c | sort -gro OOS-dest.uniq.sorted

grep "\->" OOS_Report.cat | cut -d \>  -f 2 | cut -d \: -f 2 | sort | uniq -c | sort -gro OOS-dest-
port.uniq.sorted

grep "Seq" OOS_Report.cat | awk {'print $1'} | sort | uniq -c | sort -gro OOS-TCP-Options.uniq.sorted
```

References (Part 3)

[1] Young, Paul. GCIA #0603. URL: http://www.giac.org/practical/GCIA/Paul_Young_GCIA.pdf

[2] Gordon, Les. GCIA # 0586. URL: http://www.giac.org/practical/GCIA/Les_Gordon_GCIA.doc

[3] Al Maslowski-Yerges. GCIA # 0608. URL: http://www.giac.org/practical/GCIA/Al_Maslowski-Yerges_GCIA.pdf

[4] SnortSnarf, Homepage: http://www.silicondefense.com/software/snortsnarf/

[5] Calabrese, Chris. http://www.giac.org/practical/Chris Calabrese_GCIA.zip

[6] Baker, Chris. http://www.giac.org/practical/Chris_Baker_GCIA.zip

[7] CERT® Coordination Center. “CERT® Advisory CA-2003-08 Increased Activity Targeting Windows Shares”. March 11, 2003. URL: http://www.cert.org/advisories/CA-2003-08.html

[8] CERT® Coordination Center. “CERT® Incident Note IN-2000-02 (Exploitation of Unprotected Windows Networking Shares)”. April 7, 2000. URL: http://www.cert.org/incident_notes/IN-2000-02.html

[9] Alexander, Bryce. “Intrusion Detection FAQ Port 137 Scan” May 10, 2000. URL: http://www.sans.org/resources/idfaq/port_137.php

[10] White hats, Inc. IDS177 "NETBIOS-NAME-QUERY". Nov 27, 2003. URL: http://www.whitehats.com/info/IDS339

[11] White hats, Inc. IDS339 "NETBIOS-SMB-C$ACCESS". Nov 27, 2003. URL: http://www.whitehats.com/info/IDS339

[12] Caswell, Brian & Roesch, Marty . Snort Signature Database “NETBIOS SMB C$ access”. Thu Nov 27, 2003. URL: http://www.snort.org/snort-db/sid.html?sid=533

[13] Caswell, Brian & Roesch, Marty . Snort Signature Database “SHELLCODE x86 NOOP”. Nov 28, 2003. URL: http://www.snort.org/snort-db/sid.html?sid=648

[14] Caswell, Brian & Roesch, Marty . Snort Signature Database “SHELLCODE x86 NOOP”. Nov 28, 2003. URL: http://www.snort.org/snort-db/sid.html?sid=1394

[15]] White hats, Inc. IDS181 "SHELLCODE-X86-NOPS". Nov 27, 2003. URL: http://www.whitehats.com/info/IDS181

[16] CERT® Coordination Center. “CERT® Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL”. January 17, 2002. URL: http://www.cert.org/advisories/CA-2001-19.html

[17] Tocheva K, Erdelyi G, Podrezov A, Rautiainen S. and Hypponen M. “F-Secure Virus Descriptions : Nimda”. September 18-19th, 2001. URL: http://www.f-secure.com/v-descs/nimda.shtml

[18] CERT® Coordination Center. URL: http://www.cert.org/advisories

[19] White hats, Inc. URL: http://whitehats.com/ids/index.html

[20] Snort. URL: http://www.snort.org