ICMP Usage in Scanning[1]

Ofir Arkin

The Sys-Security Group

Founder

http://www.sys-security.com
ofir.arkin@sys-security.com

ITCon – Information Technology Consultants[2]

Senior Security Consultant

http://www.itcon-ltd.com
ofir@itcon-ltd.com

July 2000

[1] This is part of "Network Scanning Techniques", by Ofir Arkin. To be published during 2000 (http://www.sys-security.com).
[2] IT Con is a leading information security consultancy company in the E-Commerce area. For more information please contact global@itcon-ltd.com.

Table of Contents

Figures List

1.0 Introduction

The Internet Control Message Protocol is one of the *debate full* protocols in the TCP/IP protocol suite regarding its security hazards. There is no consent between the experts in charge for securing Internet networks (Firewall Administrators, Network Administrators, System Administrators, Security Officers, etc.) regarding the actions that should be taken to secure their network infrastructure in order to prevent those risks.

In this paper I have tried to outline what can be done with the ICMP protocol regarding scanning.

Scanning can be defined as: The determination of the characteristics of the target network such as identifying which systems are alive and reachable via the Internet, and what services they offer, using techniques such as ping sweeps, port scans, firewalking, trace routing, and operating system identification.

This operation eventually leads to the discovery of the network topology map of the attacked network (although we will cover methods directly aimed at network topology mapping).

The kind of information collected using scanning methods can be summarized with a few simple questions:

- "What hosts are alive?"
- "What services are running on those hosts?"
- "How those hosts are organized?"
- "What are the operating systems used on those hosts?"
- "What is the role of each host?"

The data collected allow a malicious computer attacker to identify the hosts (if any) on a target network that are running a network service, which may have a known vulnerability that may allow a remote exploit.

The sections in this paper are divided according to the various methods in scanning; Host Detection using the ICMP protocol; Host Detection using ICMP error messages generated from probed machines; Inverse Mapping Using ICMP; Using Trace Route with ICMP ECHO; and The usage of ICMP in the Operating System Finger Printing process. In the last section I have described which ICMP traffic should be filtered on the Border Router and/or Firewall in order to eliminate/reduce the risks outlined in this paper.

The paper introduces new methods for Host Detection using ICMP error messages generated from probed machines and a new method for OS Finger Printing using ICMP.

I hope that this paper would educate people to eliminate some of the security hazards the ICMP protocol carries.

2.0 Host Detection using the ICMP Protocol

The Host Detection stage gives a malicious computer attacker crucial information by identifying the computers on the targeted network that are reachable from the Internet. This process belongs to the scanning stage, which is one of the first stages in the Information Gathering process. The information collected during this stage could later lead to an attempt to break in to one (or more) of the targeted network computers. This, if the information gathered would be sufficient for the malicious computer attacker.

2.1 ICMP ECHO (Type 8) and ECHO Reply (Type 0)

We can use an *ICMP ECHO* datagram to determine whether a target IP address is active or not, by simply sending an ICMP ECHO[3] (ICMP type 8) datagram to the targeted system and waiting to see if an *ICMP ECHO Reply* (ICMP type 0) is received. If an ICMP ECHO reply is received, it means that the target is alive (few firewalls spoof ICMP ECHO replies from protected hosts); No response means the target is down or a filtering device is preventing the incoming ICMP ECHO packet from getting inside the protected network or the filtering device prevents the initiated reply from reaching the Internet.

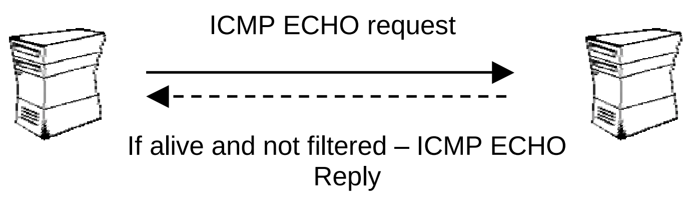


Figure 1: ICMP ECHO Mechanism

This mechanism is used by the Ping command to determine if a destination host is reachable.

In the next example two LINUX machines demonstrate the usage of Ping:

```
[root@stan /root]# ping 192.168.5.5
PING 192.168.5.5 (192.168.5.5) from 192.168.5.1 : 56(84) bytes of data.
64 bytes from 192.168.5.5: icmp_seq=0 ttl=255 time=4.4 ms
64 bytes from 192.168.5.5: icmp_seq=1 ttl=255 time=5.9 ms
64 bytes from 192.168.5.5: icmp_seq=2 ttl=255 time=5.8 ms

--- 192.168.5.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 4.4/5.3/5.9 ms
```

A Snort trace[4]:

```
01/26-13:16:25.746316 192.168.5.1 -> 192.168.5.5
ICMP TTL:64 TOS:0x0 ID:6059
```

[3] From a Technical point of view: The sending side initializes the identifier (used to identify ECHO requests aimed at different destination hosts) and sequence number (if multiple ECHO requests are sent to the same destination host), adds some data (arbitrary) to the data field and sends the ICMP ECHO to the destination host. In the ICMP header the code equals zero. The recipient should *only change* the type to ECHO Reply and return the datagram to the sender.

[4] Snort, written by Martin Roesch, can be found at http://www.clark.net/~roesch/security.html.

```
ID:5721   Seq:1  ECHO
89 D7 8E 38 27 63 0B 00 08 09 0A 0B 0C 0D 0E 0F  ...8'c..........
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567

01/26-13:16:25.746638 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0x0 ID:6072
ID:5721   Seq:1  ECHO REPLY
89 D7 8E 38 27 63 0B 00 08 09 0A 0B 0C 0D 0E 0F  ...8'c..........
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567
```

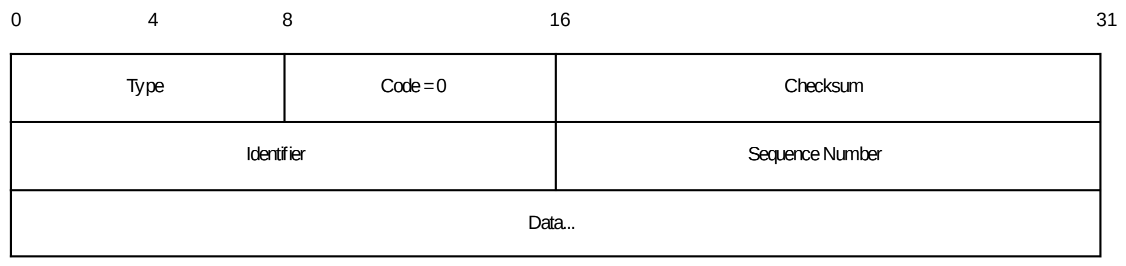


Figure 2: ICMP ECHO Request & Reply message format

Countermeasure: Block ICMP ECHO requests coming from the Internet towards your network at your border router and/or Firewall[5].

2.2 ICMP Sweep

Querying multiple hosts using ICMP ECHO is referred to as *ICMP Sweep* (or *Ping Sweep*).

For a small to midsize network Ping is an acceptable solution to this kind of host detection, but with large networks (such as Class A, or a full Class B) this kind of scan is fairly slow mainly because Ping waits for a reply (or a time out to be reached) from the probed host before proceeding to the next one.

fping[6] is a UNIX utility which sends parallel mass ECHO requests in a round robin fashion enabling it to be significantly faster than the usual Ping utility. It can also be fed with IP addresses with its accompanied tool *gping*. gping is used to generate a list of IP addresses which would be later fed into fping, directly or from a file, to perform the ICMP sweep. fping is also able to resolve hostnames of the probed machines if using the –d option.

Another UNIX tool that is able of doing an ICMP sweep in parallel, resolve the hostnames of the probed machines, save it to a file and a lot more is NMAP[7], written by Fyodor.

[5] It is better to filter unwanted traffic at your border router, reducing traffic rates for your firewall.
[5] http://ftp.tamu.edu/pub/Unix/src/
[7] http://www.insecure.org

For Windows a notable ICMP sweep tool is Pinger from Rhino9[8], able of doing what fping and NMAP do regarding this kind of scan.

Trying to resolve the names of the probed machines may discover the attackers IP number used for the probing, using the log of the authoritative DNS server.

The next example demonstrates the usage of NMAP to perform an ICMP sweep[9] against 20 IP addresses. Our test lab contains two LINUX machines running Redhat Linux v6.1, Kernel 2.2.12 (Stan & Kenny) and one Windows NT WRKS SP4 (Cartman). As it can be seen all of the machines answered the probe:

```
[root@stan /root]# nmap -sP -PI 192.168.5.1-20

Starting nmap V. 2.3BETA13 by fyodor@insecure.org (
www.insecure.org/nmap/ )
Host stan.sys-security.com (192.168.5.1) appears to be up.
Host kenny.sys-security.com (192.168.5.5) appears to be up.
Host cartman.sys-security.com (192.168.5.15) appears to be up.
Nmap run completed -- 20 IP addresses (3 hosts up) scanned in 3 seconds
```

If we wish to avoid the automatic resolving done by NMAP we should use the –n option to eliminate it.

ICMP sweeps are easily detected by IDS (Intrusion Detection Systems) whether launched in the regular way, or if used in a parallel way.

Countermeasure: Block ICMP ECHO requests coming from the Internet towards your network at your border router and/or Firewall.

2.3 Broadcast ICMP

A simpler way to map the targeted network for alive hosts is by sending an ICMP ECHO request to the broadcast or the network address of the targeted network.

The request would be broadcasted to all hosts on the targeted network. The alive hosts will send an ICMP ECHO Reply to the attacker source IP address.

The malicious computer attacker has to send only one IP datagram to produce this behavior.

This technique of host detection is applicable only to the UNIX hosts of the targeted network. Windows machines will not generate an answer (ICMP ECHO Reply) to an ICMP ECHO request aimed at the broadcast address or at the network address. They are configured not to answer those queries out-of-the box (Microsoft Windows NT 4.0 SP4 and above, Microsoft Windows 2000). This is not an abnormal behavior as RFC 1122[10] states that if we send an ICMP ECHO request to an IP Broadcast or IP Multicast addresses it *may* be silently discarded by a host.

[8] The Rhino9 group no longer exists. Their tools are available from a number of sites on the Internet to handle.

[9] The –sP –PI options enable NMAP to perform only an ICMP Sweep. The default behavior when using the –sP option is different and includes the usage of TCP ACK host detection technique as well.

[10] RFC 1122: Requirements for Internet Hosts - Communication Layers, http://www.ietf.org/rfc/rfc1122.txt.

The next example demonstrates the behavior expected from hosts when sending an ICMP ECHO request to the broadcast address of a network. The two LINUX machines on our test lab answered the query while the Microsoft Windows NT WRKS 4.0 machine silently ignored it.

```
[root@stan /root]# ping -b 192.168.5.255
WARNING: pinging broadcast address
PING 192.168.5.255 (192.168.5.255) from 192.168.5.1 : 56(84) bytes of
data.
64 bytes from 192.168.5.1: icmp_seq=0 ttl=255 time=4.1 ms
64 bytes from 192.168.5.5: icmp_seq=0 ttl=255 time=5.7 ms (DUP!)

--- 192.168.5.255 ping statistics ---
1 packets transmitted, 1 packets received, +1 duplicates, 0% packet
loss
round-trip min/avg/max = 4.1/4.9/5.7 ms
```

In the next example I have sent a Ping to the network address of the targeted network. The same behavior was produced. The LINUX machines answer the ICMP ECHO request while the Windows NT machine ignored it.

```
[root@stan /root]# ping -b 192.168.5.0
WARNING: pinging broadcast address
PING 192.168.5.0 (192.168.5.0) from 192.168.5.1 : 56(84) bytes of data.
64 bytes from 192.168.5.1: icmp_seq=0 ttl=255 time=7.5 ms
64 bytes from 192.168.5.5: icmp_seq=0 ttl=255 time=9.1 ms (DUP!)

--- 192.168.5.0 ping statistics ---
1 packets transmitted, 1 packets received, +1 duplicates, 0% packet
loss
round-trip min/avg/max = 7.5/8.3/9.1 ms
```

Note: Broadcast ICMP may result in a *Denial-Of-Service* condition if a lot of machines response to the query at once.

Countermeasure: Block the IP directed broadcast on the border router.

2.4 Non-ECHO ICMP

ICMP ECHO is not the only ICMP query message type available with the ICMP protocol.

Non-ECHO ICMP messages are being used for more advanced ICMP scanning techniques (not only probing hosts, but network devices such as a router as well).

The group of ICMP query message types includes: ECHO (Request (Type 8), Reply (Type 0)), Time Stamp (Request (Type 13), Reply (Type 14)), Information (Request (Type 15), Reply (Type 16)), and Address Mask (Request (Type 17), Reply (Type 18)).

The Information Request & Reply mechanism is now absolute as stated in RFC 1122 and RFC 1812[11].

This leaves us with two types of Non-ECHO ICMP query messages available for our usage – Time Stamp (Request and Reply) and Address Mask (Request and Reply).

2.4.1 ICMP Time Stamp Request (Type 13) and Reply (Type 14)

The *ICMP Time Stamp Request and Reply* allows a node to query another for the current time. The sender initializes the identifier (used to identify Timestamp requests aimed at different destination hosts) and sequence number (if multiple Timestamp requests are sent to the same destination host), sets the originate time stamp and sends it to the recipient.

The receiving host fills in the receive and transmit time stamps, change the type of the message to time stamp reply and returns it to the recipient. The time stamp is the number of milliseconds elapsed since midnight UT (GMT).

The originate time stamp is the time the sender last touched the message before sending it, the receive time stamp is the time the recipient first touched it on receipt, and the Transmit time stamp is the time the receiver last touched the message on sending it.

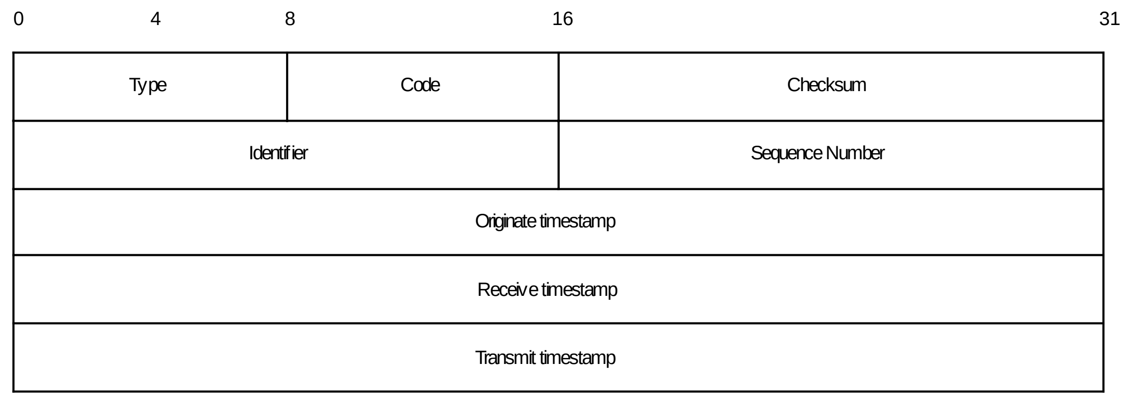


Figure 3: ICMP Time Stamp Request & Reply message format

As RFC 1122 state, a *host may* implement Timestamp and Timestamp Reply. If they are implemented a host must follow this rules:

- Minimum variability delay in handling the Timestamp request.
- The receiving host *must* answer to every Timestamp request that he receives.
- An ICMP Timestamp Request to an IP Broadcast or IP Multicast address *may* be silently discarded.
- The IP source address in an ICMP Timestamp reply *must* be the same as the specific-destination address of the corresponding Timestamp request message.
- If a source-route option is received in a Timestamp request, the return route *must* be reserved and used as a Source Route option for the Timestamp Reply option.

[11] RFC 1812: Requirements for IP Version 4 Routers, http://www.ietf.org/rfc/rfc1812.txt . As the RFC states this mechanism is now *obsolete* - A *router should not* originate or respond to these messages; A *host should not* implement these messages.

- o If a Record Route and/or Timestamp option is received in a Timestamp request, this option(s) *should* be updated to include the current host and included in the IP header of the Timestamp Reply message.

Receiving an ICMP Timestamp Reply would reveal an alive host (or a networking device) that has implemented the ICMP Timestamp messages.

In the next example I have sent an ICMP Time Stamp Request, using the icmpush[12] tool, to a Redhat 6.1 LINUX, Kernel 2.2.12 machine:

```
[root@stan /root]# icmpush -tstamp 192.168.5.5
kenny.sys-security.com -> 13:48:07
```

Snort Trace:

```
01/26-13:51:29.342647 192.168.5.1 -> 192.168.5.5
ICMP TTL:254 TOS:0x0 ID:13170
TIMESTAMP REQUEST
88 16 D8 D9 02 8B 63 3D 00 00 00 00 00 00 00 00  ......c=........

01/26-13:51:29.342885 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0x0 ID:6096
TIMESTAMP REPLY
88 16 D8 D9 02 8B 63 3D 02 88 50 18 02 88 50 18  ......c=..P...P.
2A DE 1C 00 A0 F9                                *.....
```

When I have sent an ICMP Time Stamp Request to a Windows NT WRKS 4.0 SP4 machine, I got no reply. Again, this is not an abnormal behavior from the Microsoft Windows NT machine, just an implementation choice as RFC 1122 states.

Countermeasure: Block ICMP Time Stamp Requests coming from the Internet on the border Router and/or Firewall.

2.4.2 ICMP Address Mask Request (Type 17) and Reply (Type 18)

The *ICMP Address Mask Request* (and Reply) is intended for diskless systems to obtain its subnet mask at bootstrap time. Address Mask request is also used when a node wants to know the address mask of an interface. The reply (if any) contains the mask of that interface.

A host will broadcast an address mask request, any host on the network that has been configured to send address mask replies will fill in the subnet mask, change the type of the message to address mask reply and return it to the sender.

[12] Icmpush was written by Slayer of hispahack.http://hispahack.ccc.de/ .

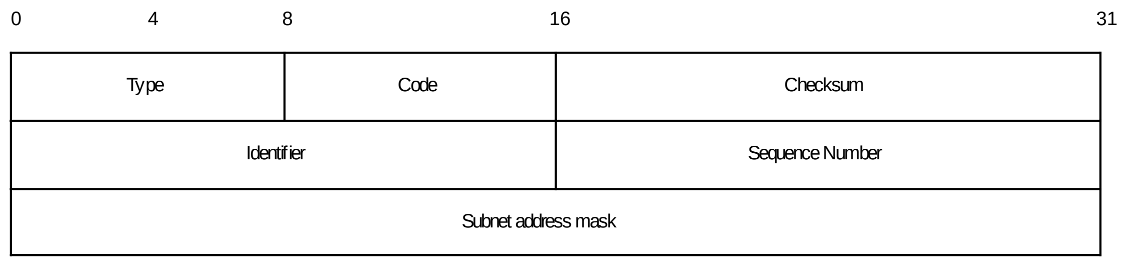


Figure 4: ICMP Address Mask Request & Reply message format

RFC 1122 states that a system that has implemented ICMP Address Mask messages *must not* send an Address Mask Reply unless it is an authoritative agent for address masks.

Usually an Address Mask request would be answered by a gateway.

Receiving an Address Mask Reply from a host would reveal an alive host that is an authoritative agent for address masks. It will also allow a malicious computer attacker to gain knowledge about your network's configuration. This information can assist the malicious computer attacker in determining your internal network structure, as well as the routing scheme.

Please note that a Router *must* implement ICMP Address Mask messages. This will help identify routers along the path to the targeted network (it can also reveal internal routers if this kind of traffic is allowed to reach them).

If the Router is following RFC 1812 closely, it should not forward on an Address Mask Request to another network.

ICMP Address Mask Request aimed at a LINUX machine would not trigger an ICMP Address Mask Reply, nor a request aimed at a Microsoft Windows box.

In the next example I have sent an ICMP Address Mask Request to the broadcast address (192.168.5.255) of a class C network 192.168.5.0, spoofing the source IP to be 192.168.5.3:

```
[root@stan /root]# icmpush -vv -mask -sp 192.168.5.3 192.168.5.255
 -> ICMP total size = 12 bytes
 -> Outgoing interface = 192.168.5.1
 -> MTU = 1500 bytes
 -> Total packet size (ICMP + IP) = 32 bytes
ICMP Address Mask Request packet sent to 192.168.5.255 (192.168.5.255)

Receiving ICMP replies ...
-----------------------------------------------------
192.168.5.3 ...
   Type = Address Mask Request (0x11)
   Code = 0x0     Checksum = 0xBF87
     Id = 0x3B7       Seq# = 0x3CB0
-----------------------------------------------------
icmpush: Program finished OK
```

```
-*> Snort! <*-
Version 1.5
By Martin Roesch (roesch@clark.net, www.clark.net/~roesch)
Kernel filter, protocol ALL, raw packet socket
Decoding Ethernet on interface eth0
02/15-13:47:37.179276 192.168.5.3 -> 192.168.5.255
ICMP TTL:254 TOS:0x0 ID:13170
ADDRESS REQUEST
B9 03 8E 49 00 00 00 00                          ...I....
```

No answer was received from the LINUX machines nor from the Windows NT machine on our test lab.

But when sending an ICMP Address Mask request aimed at a router on our network we receive a reply:

```
-*> Snort! <*-
Version 1.5
By Martin Roesch (roesch@clark.net, www.clark.net/~roesch)
Decoding Ethernet on interface eth0
01/30-09:33:00.711595 Host[13] -> Destination_ Router
ICMP TTL:254 TOS:0x0 ID:13170
ADDRESS REQUEST
90 02 04 17 00 00 00 00                          ........

01/30-09:33:00.717388 Destination_Router -> Host
ICMP TTL:63 TOS:0x0 ID:367
ADDRESS REPLY
90 02 04 17 FF FF FF F8 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00                                ......
```

Countermeasure: Block ICMP Address Mask Requests coming from the Internet on the border Router and/or Firewall.

2.5 Non-ECHO ICMP Sweeps

We can query multiple hosts using a Non-ECHO ICMP query message type. This is referred as a Non-ECHO ICMP sweep.

Who would answer our query?

Hosts that answer to the following:

- o Hosts that are in a listening state.
- o Hosts running an operating system that implemented the Non-ECHO ICMP query message type that was sent.
- o Hosts that are configured to reply to the Non-ECHO ICMP query message type (few conditions here as well, for example: RFC 1122 states that a system that implemented ICMP Address Mask messages *must not* send an Address Mask Reply unless it is an authoritative agent for address masks).

[13] The real IP Adresses of the Host IP and a local ISPs router were replaced.

Given the conditions above, the answering host(s) would almost always be a *NIX machine configured to answer for an ICMP Time Stamp Request with a Time Stamp Reply.

Countermeasure: Block ICMP Address Mask Requests & ICMP Time Stamp Requests coming from the Internet on the border Router and/or Firewall.

2.6 Non-ECHO ICMP Broadcasts

We can send a Non-ECHO ICMP query message type to the broadcast address or to the network address of the targeted network.

The request would be broadcasted to all listening hosts on the targeted network.

Who would answer our query?

- Hosts that are in a listening state
- Hosts running an operating system that implemented the Non-ECHO ICMP query message type that was sent.
- Hosts that are configured to reply to the Non-ECHO ICMP query message type (few conditions here as well, for example: a host may discard Non-ECHO ICMP query message type requests targeted at the broadcast address).

Given the conditions above, the answering hosts would almost always be *NIX machines configured to answer for an ICMP Time Stamp Request and initiate a Time Stamp Reply.

Countermeasure: Block the IP directed broadcast on the border router. Block ICMP Address Mask Requests & ICMP Time Stamp Requests coming from the Internet on the border Router and/or Firewall.

3.0 Host Detection using ICMP Error Messages generated from the probed machines

We can use various methods in order to elicit an ICMP Error Message back from a probed machine and discover its existence. Some of the methods described here are:

- Mangling IP headers
- Using non-valid field values in the IP header
    - o Using valid field values in the IP header
- Abusing Fragmentation
- The UDP Scan Host Detection method

With the first method we are using bad IP headers in the IP packet that would generate an ICMP Parameter Problem error back from the probed machine to the source IP address of the probing packet. The second method use non-valid field values in the IP header in order to force the probed machine to generate ICMP Destination Unreachable error message back to the prober. The third method discussed uses fragmentation to trigger an ICMP Fragment Reassembly Time Exceeded error message from the probed machine. The last method uses the UDP Scan method to elicit ICMP Port Unreachable error message back from a closed UDP port(s) on the probed host(s).

When using some of those methods we can determine if a filtering device is present and some can also discover the Access Control List a Filtering Device is forcing on the protected network.

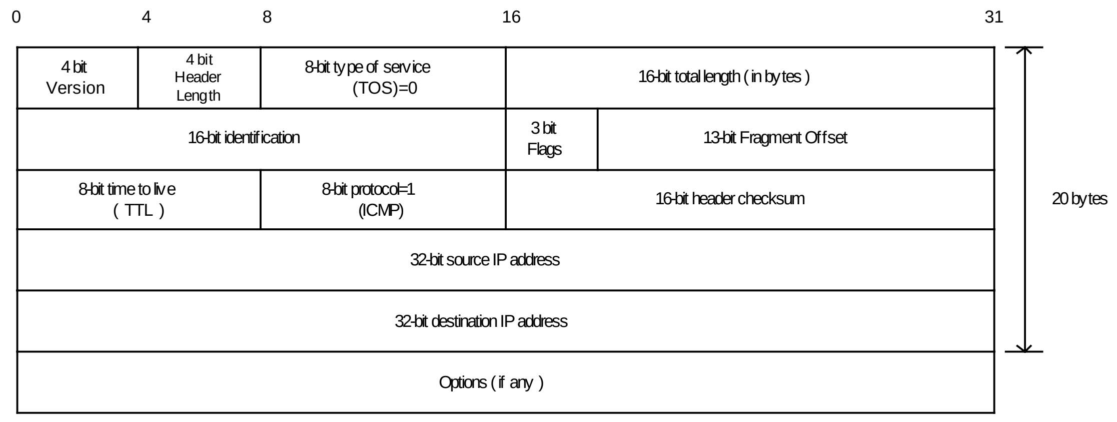


Figure 5: The IP Header

3.1 IP Packets with bad IP headers fields – generating ICMP Parameter Problem error message back from probed machines

An ICMP Parameter Problem error message is sent when a router (*must* generate this message) or a host (*should* generate this message) process a datagram and finds a problem with the IP header parameters. It is only sent if the error caused the datagram to be discarded.

The Parameter Problem message is generated usually for any error not specifically covered by another ICMP message.

We have some variants with this type of Host Detection. We send an illegal forged packet(s) with bad IP header field(s), that no specific ICMP error message is sent for this field(s). It will force a Host to send back an ICMP Parameter Problem Error message Code 0 (When code 0 is used, the pointer field will point to the exact byte in the original IP Header, which caused the problem) to the source IP address of the bad IP packet and reveal its existence. With this type of host detection it is not relevant what would be the protocol (TCP/UDP/ICMP) embedded inside the IP packet. All we care about is the Error messages generated by the probed machine (if any).

This method is very powerful in detecting host(s) on the probed network with direct access from the Internet, since a host should generate this error message. Routers must generate the ICMP Parameter Problem error message as well, but not all of them check the correctness of some fields inside the IP header like a host does (processing of some fields is done on the host only).

According to RFC 1122 a host should check for validity of the following fields when processing a packet[14]:

- Version Number – if not 4 a host must silently discard the IP packet.
- Checksum – a host should verify the IP header checksum on every received datagram and silently discard every datagram that has a bad checksum.

A router should check for the validity of the following fields when processing a packet[15]:

- Checksum – a router must verify the IP checksum of any packet it received, and must discard messages containing invalid checksums.

The conditions outlined eliminate the usage of this method to a limited number of fields only.

It is possible to send an IP packet with bad field(s) in the IP header, which will get routed without getting dropped in the way to the probed machine. It should be noted that different routers perform different checks regarding the IP header (different implementation and interpretation of RFC 1812). When a router, because of a bad IP header, drops an IP packet and sends an ICMP Parameter Problem error message, it is possible to identify the manufacture of the router, and to adjust the wrong IP header field correctly according to a field, which is not checked by the manufacture.

A router may be more forgiving than a Host regarding the IP header. This may result from the fact that a router is a vehicle for delivering the IP packet and a Host is the Destination and the place where more processing on the packet is done.

The downside for this method is the detection. Intrusion Detection Systems *should* alert you about abnormalities in the attacked network traffic, since not every day you receive IP packets with bad IP Header field(s).

We can use this type of Host Detection to sweep through the entire IP range of an organization and get back results, which will map all the alive hosts on the probed network with direct access from the Internet.

Even if a firewall or any filtering device is protecting the probed network we can still try to send those forged packets to an IP addresses with ports that are likely to be opened, for example - TCP ports 21,25,80; UDP port 53; and even try to send an ICMP message presumably coming back from a Host/Router who generated it upon receiving data from the attacked network.

[14] RFC 1122 – Requirements for Internet Host, http://www.ietf.org/rfc/rfc1122.txt.
[15] RFC 1812 – Requirements for IPv4 Routers, http://www.ietf.org/rfc/rfc1812.txt.

In my opinion Firewalls/Filtering Devices should check the validity of those fields used to elicit the ICMP Parameter Problem error message and disallow this kind of traffic.

An example is given here using the *ISIC* tool written by Mike Frantzen[16]. ISIC sends randomly generated packets to a target computer. Its primary uses are to stress test an IP stack, to find leaks in a firewall, and to test the implementation of Intrusion Detection Systems and firewalls. The user can specify how often the packets will be fragmented; have IP options, TCP options, an urgent pointer, etc.

In the next example I have sent 20 IP Packets from a LINUX machine to a Microsoft Windows NT WRKS 4 SP4 machine. The packets were not fragmented nor bad IP version numbers were sent. The only weird thing sent inside the IP headers was random IP Header length, which have produced ICMP Parameter Problem error message as I anticipated.

```
[root@stan packetshaping]# ./isic -s 192.168.5.5 -d 192.168.5.15 -p 20
-F 0 -V 0 -I 100
Compiled against Libnet 1.0
Installing Signal Handlers.
Seeding with 2015
No Maximum traffic limiter
Bad IP Version  = 0%            Odd IP Header Length    = 100%
Frag'd Pcnt     = 0%

Wrote 20 packets in 0.03s @ 637.94 pkts/s
```

tcpdump trace:

```
12:11:05.843480 eth0 > kenny.sys-security.com > cartman.sys-
security.com: ip-proto-110 226 [tos 0xe6,ECT]  (ttl 110, id 119,
optlen=24[|ip])

12:11:05.843961 eth0 P cartman.sys-security.com > kenny.sys-
security.com: icmp: parameter problem - octet 21 Offending pkt:
kenny.sys-security.com > cartman.sys-security.com: ip-proto-110 226
[tos 0xe6,ECT]  (ttl 110, id 119, optlen=24[|ip]) (ttl 128, id 37776)
```

3.1.1 ACL Detection using IP Packets with bad IP headers fields

If we probe the entire IP range of the targeted network with all combinations of protocols and ports, it would draw us the targeted network topology map, and will allow us to determine the ACL of the Filtering Device (If present, and not blocking outgoing ICMP Parameter Problem Error messages).

This, if the filtering device does not check the validity of the mangled IP header fields.

3.1.1.1 How we determine the ACL (ICMP Protocol embedded inside)?

When the embedded protocol is ICMP, we send various ICMP message types encapsulated inside IP packets with bad IP header(s). If we receive a reply from a Destination IP address we have a host that is alive and an ACL, which allows this type of message of ICMP to get to the host who generated the error message (and the Parameter Problem ICMP error message is allowed from the destination host to the Internet).

[16] http://expert.cc.purdue.edu/~frantzen/

If we are not getting any reply than one of three possibilities:

- The Filtering Device disallows packets with the kind of bad field we are using.
- The Filtering Device is filtering the Type of ICMP message we are using.
- The Filtering Device blocks ICMP Parameter Problem error messages initiated from the protected network destined to the Internet.

3.1.1.2 How we determine the ACL (TCP or UDP Protocol embedded inside)?
We can probe for every combination of protocol and port values inside an IP packet with bad IP header(s). If we would receive an answer it would indicate that the protocol and port we used are allowed to the probed host from the Internet, and the ICMP Parameter Problem error message is allowed from the destination host in the protected network out to the Internet. It would also indicate that the filtering device used on the targeted network is not validating the correctness of the fields we have used in order to elicit the ICMP Parameter Problem error message.

If the embedded protocol were either TCP or UDP, a reply would not be generated if:

- The Filtering Device disallows packets with the kind of bad field we are using.
- The Filtering Device filters the Protocol used.
- The Filtering Device is filtering the specific port we are using for the probe.
- The Filtering Device blocks ICMP Parameter Problem error messages initiated from the protected network destined to the Internet. In our case, the filtering device may be blocking the specific host we are probing for outgoing ICMP Parameter Problem datagrams.

Countermeasure: Block outgoing ICMP Parameter Problem from the protected network to the Internet on the Firewall & on the border Router.

Check with the manufacture of your filtering device which fields it validates on the IP header.

3.2 IP Packets with non-valid field values
This Host Detection method is based on different IP header fields within the crafted IP packet that would have non-valid field values, which would trigger an ICMP Destination Unreachable Error message back from the probed machines.

Note that some hosts (AIX, HP-UX, Digital UNIX) may not send ICMP Protocol Unreachable messages.

3.2.1 The Protocol Field example
3.2.1.1 Using non-Valid (not used) IP protocol values
One such field within the IP header is the protocol field. If we will put a value, which does not represent a valid protocol number, the probed machine would elicit an ICMP Destination Unreachable – Protocol Unreachable error message back to the probed machine.

By sending this kind of crafted packets to all IP addresses within the IP address range of the probed network we can map the hosts that are directly connected to the Internet (assuming that no filtering device is present, or filter the specific traffic).

3.2.1.1.1 Detecting if a Filtering Device is present

A packet sent with a protocol value, which does not represent a valid protocol number, should elicit an ICMP Destination Unreachable – Protocol Unreachable from the probed machine. Since this value is not used (and not valid) all hosts probed, unless filtered or are AIX, HP-UX, Digital UNIX machines, should send this reply. If a reply is not received we can assume that a filtering device prevents our packet from reaching our destination or from the reply to reach us back.

3.2.1.2 Using all combination of the IP protocol filed values

The difference with this variant is that we use all of the combinations available for the IP protocol field – since the IP protocol field has only 8 bits in length, there could be 256 combinations available.

NMAP 2.54 Beta 1 has integrated this variant and Fyodor have named it - IP Protocol scan. NMAP sends raw IP packets *without any further protocol header* to each specified protocol on the target machine. If an ICMP Protocol Unreachable error message is received, the protocol is not in use. Otherwise it is assumed it is opened (or a filtering device is dropping our packets).

If our goal was Host Detection only, than using the NMAP implementation would be just fine. But if we wish to use this scan type for other purposes, such as ACL detection, than we would need the protocol header as well.

We can determine if a filtering device is present quite easily using this scan method. If a large number of protocols (non valid values could be among those) seems to be “opened”/used (not receiving any reply – ICMP Protocol Unreachable) than we can assume a filtering device is blocking our probes (if using a packet with the protocol headers as well). If the filtering device is blocking the ICMP Protocol Unreachable error messages initiated from the protected network towards the Internet than all of the 256 possible protocol values would be seemed “opened”/used.

With the current implementation with NMAP the 256 possible protocol values should be “opened” when a scan is performed against a machine inside a protected network, because a packet filter firewall (or other kind of firewall) *should* block the probe since it lacks information to validate the traffic against its rule base (information in the protocol headers such as ports for example).

In the next example I have used NMAP 2.54 Beta 1 in order to scan a Microsoft Windows 2000 Professional machine:

```
[root@catman /root]# nmap -vv -sO 192.168.1.1

Starting nmap V. 2.54BETA1 by fyodor@insecure.org (
www.insecure.org/nmap/ )
Host  (192.168.1.1) appears to be up ... good.
Initiating FIN,NULL, UDP, or Xmas stealth scan against  (192.168.1.1)
The UDP or stealth FIN/NULL/XMAS scan took 4 seconds to scan 254 ports.
Interesting protocols on  (192.168.1.1):
(The 250 protocols scanned but not shown below are in state: closed)
Protocol   State       Name
1          open        icmp
2          open        igmp
6          open        tcp
17         open        udp

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

A tcpdump trace of some of the communication exchanged:

```
17:44:45.651855 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-50
0 (ttl 38, id 29363)
17:44:45.652169 eth0 < 192.168.1.1 > localhost.localdomain: icmp:
192.168.1.1 protocol 50 unreachable Offending pkt:
localhost.localdomain > 192.168.1.1: ip-proto-50 0 (ttl 38, id 29363)
(ttl 128, id 578)
17:44:45.652431 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-
133 0 (ttl 38, id 18)
17:44:45.652538 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-
253 0 (ttl 38, id 36169)
17:44:45.652626 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-92
0 (ttl 38, id 26465)
17:44:45.652727 eth0 < 192.168.1.1 > localhost.localdomain: icmp:
192.168.1.1 protocol 133 unreachable Offending pkt:
localhost.localdomain > 192.168.1.1: ip-proto-133 0 (ttl 38, id 18)
(ttl 128, id 579)
17:44:45.652760 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-
143 0 (ttl 38, id 14467)
17:44:45.652899 eth0 > localhost.localdomain > 192.168.1.1: ip-proto-30
0 (ttl 38, id 30441)
17:44:45.652932 eth0 < 192.168.1.1 > localhost.localdomain: icmp:
192.168.1.1 protocol 253 unreachable Offending pkt:
localhost.localdomain > 192.168.1.1: ip-proto-253 0 (ttl 38, id 36169)
(ttl 128, id 580)
```

3.2.2 ACL Detection using the Protocol field

First we need to determine if a filtering device is present using a non-valid (not used) protocol number probe. If a filtering device exists then no answer (ICMP Protocol Unreachable) will be received from the probed machine, assuming it is not AIX, HP-UX or Digital UNIX[17].

If a certain protocol were not allowed through the filtering device we would not receive any ICMP error message from the probed machine. Probing for all combinations of protocols and ports against an IP range of a targeted network using non-valid and valid protocol values can determine the ACL a filtering device is forcing on the protected network, along with the topology map of a targeted network (hosts reachable from the Internet).

A reply would not be generated if:

- The Filtering Device filters the Protocol we are using
- The Filtering Device is filtering the specific port we are using for the probe.
- The Filtering Device blocks ICMP Destination Unreachable - Protocol Unreachable error messages initiated from the protected network destined to the Internet. In our case, the filtering device may be blocking the specific host we are probing for outgoing ICMP Destination Unreachable - Protocol Unreachable error messages.

Note: We can use this method for ACL detection but if the protocol we are using is not used on the target machine it should be blocked on the filtering device. Than, only opened TCP/UDP ports and allowed ICMP traffic could traverse the filtering device. if that kind of traffic is allowed we can have better ACL detection solutions then we outlined here.

[17] You can determine this using OS finger printing methods.

Countermeasure: Block outgoing ICMP Protocol Unreachable error messages coming from the protected network to the Internet on your Firewall and/or Border Router. If you are using a firewall check that your firewall block protocols which are not supported (deny all stance).

3.3 Host Detection using IP fragmentation to elicit Fragment Reassembly Time Exceeded ICMP error message.

When a host receives a fragmented datagram with some of its pieces missing, and does not get the missing parts within a certain amount of time the host will discard the packet and generate an ICMP Fragment Reassembly Time Exceeded error message back to the sending host.

We can use this behavior as a Host Detection method, by sending fragmented datagrams with missing fragments to a probed host, and wait for an ICMP Fragment Reassembly Time Exceeded error message to be received from a live host(s), if any.

When we are using this method against all of the IP range of a probed network, we will discover the network topology of that targeted network.

3.3.1 ACL Detection using IP fragmentation

This method can be used not only to map the entire topology map of the targeted network, but also to determine the ACL a firewall or a filtering device is forcing on the protected network.

Simply using all combinations of TCP and UDP with different ports, with the IP addresses from the IP range of the probed network will do it. When we receive a reply it means a host we queried is alive, the port we have used is opened on that host, and the ACL allows the protocol type and the port that was used to get to the probed machine (and the ICMP Fragment Reassembly Time Exceeded error message back from the probed machine to the Internet).

If we were not getting any reply back from the probed machine it can mean:

- The Filtering Device filters the Protocol used.
- The Filtering Device is filtering the specific port we are using for the probe.
- The Filtering Device blocks ICMP Fragment Reassembly Time Exceeded error messages initiated from the protected network destined to the Internet. In our case, the filtering device may be blocking the specific host we are probing for outgoing ICMP Parameter Problem datagrams.

3.3.1.1 An Example with UDP (Filtering Device Detection)

Since UDP is a stateless protocol it may be better suited for our needs here. The first datagram would be fragmented including enough UDP information in the first fragmented datagram that would be enough to verify the packet against a Firewall’s Rule base. The second part of the datagram would not be sent. It would force any host that gets such a packet to send us back an ICMP Fragment Reassembly Time Exceeded error message when the time for reassembly exceeds.

If the port we were using were an open port, than the ICMP Fragment Reassembly Time Exceeded error message would be generated. If the port were closed then an ICMP Port Unreachable error message would be produced.

If a firewall is blocking our probed than *no reply* would be generated.

No reply would be an indication that traffic to the Host we probed is filtered.

3.3.1.2 An example with TCP

We can divide the first packet of a TCP handshake into two fragments. We would put enough TCP information in the first packet that would be enough to verify the packet against the Firewall's Rule base (this means the port numbers we are using are included in the packet). We will not send the second part of the packet, forcing any host that gets such a packet to send us back an ICMP Fragment Reassembly Time Exceeded error message when the time for reassembly exceeds. This would indicate the host is accessible by this kind of traffic, which is allowed using the port we have specified as the destination port[18].

If the port we use were open, than the ICMP error message would be generated. If the port is closed than a TCP RST packet should be sent back. If a filtering device were to block our probes than no reply would be generated. No reply would be an indication that traffic to the host we probed is filtered or the filtering device requires that the first TCP packet would not be fragmented (which is a legitimate requirement).

3.3.1.3 An Example with ICMP

We can do the same with encapsulating the ICMP protocol. When doing so the ICMP fragmented packets should sound the sirens when an Intrusion Detection system (if deployed) sees them. There is no reason to fragment an ICMP datagram.

Countermeasure: Block outgoing ICMP Fragment Reassembly Time Exceeded Error messages.

3.4 Host Detection using UDP Scans, or why we wait for the ICMP Port Unreachable

How can we determine if a host is alive using a UDP probe? – We use the UDP scan method that uses ICMP Port Unreachable error message that may be generated from probed hosts as indicator of alive hosts. With this method we are sending a UDP datagram with 0 bytes of data to a UDP port on the attacked machine. If we have sent the datagram to a closed UDP port we will receive an ICMP Port Unreachable error message. If the port is opened, we would not receive any reply.

When a filtering device is blocking UDP traffic aimed at the attacked machine, it would copycat the behavior pattern as with opened UDP ports.

If we probe a large number of UDP ports on the same host and we do not receive a reply from a large number of ports, it would look like that a large number of probed UDP ports are opened. While a filtering device is probably blocking the traffic and nearly all of the ports are closed.

How can we remedy this?
We can set a threshold number of non-answering UDP ports, when reached we will assume a filtering device is blocking our probes.

[18] In a case were a firewall is validating that the first packet is not fragmented, we can fragment another one instead. But than this scanning method would not be any different from any other scanning method using TCP flags combinations.

Fyodor has implemented a threshold with NMAP 2.3 BETA 13, so when doing a UDP scan and not receiving an answer from a certain number of ports, it would assume a filtering device is monitoring the traffic, rather than reporting those ports as opened.

3.4.1 A Better Host Detection Using UDP Scan

We will take the UDP scan method and tweak it a bit for our needs. We know that a closed UDP port will generate an ICMP Port Unreachable error message indicating the state of the port - closed UDP port. We will choose a UDP port that should be definitely closed (according to the IANA list of assigned ports ftp://ftp.isi.edu/in-notes/iana/assignments/port-numbers). For example we can use port 0 (but it would reveal our probe pretty easily).

Based on the fact that sending a UDP datagram to a closed port should elicit an ICMP Port Unreachable, we would send one datagram to the port we have chosen, than:

- If no filtering device is present we will receive an ICMP Port Unreachable error message, which will indicate that the Host is alive.
- If no answer is given – a filtering device is covering that port.

In the next example I have used the HPING2[19] tool to send one UDP datagram to host 192.168.5.5 port 50, which was closed:

```
[root@stan /root]# hping2 -2 192.168.5.5 -p 50 -c 1
default routing not present
HPING 192.168.5.5 (eth0 192.168.5.5): udp mode set, 28 headers + 0 data
bytes
ICMP Port Unreachable from 192.168.5.5  (kenny.sys-security.com)

--- 192.168.5.5 hping statistic ---
1 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms


-*> Snort! <*-
Version 1.5
By Martin Roesch (roesch@clark.net, www.clark.net/~roesch)
Kernel filter, protocol ALL, raw packet socket
Decoding Ethernet on interface eth0
03/12-12:54:47.274096 192.168.5.1:2420 -> 192.168.5.5:50
UDP TTL:64 TOS:0x0 ID:57254
Len: 8

03/12-12:54:47.274360 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0xC0 ID:0
DESTINATION UNREACHABLE: PORT UNREACHABLE
00 00 00 00 45 00 00 1C DF A6 00 00 40 11 0F D4  ....E.......@...
C0 A8 05 01 C0 A8 05 05 09 74 00 32 00 08 6A E1  .........t.2..j.
```

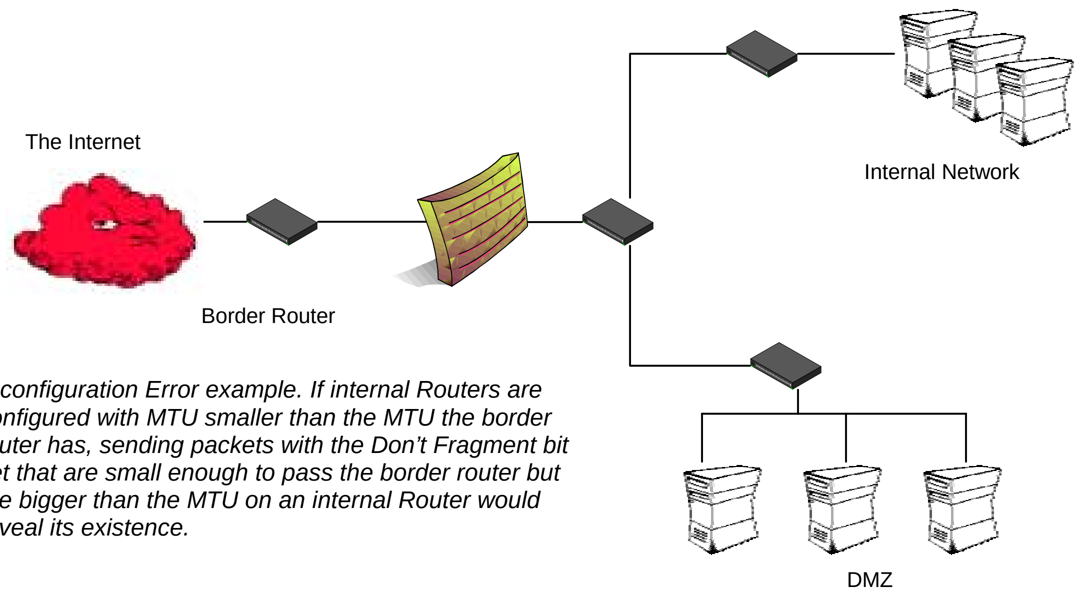
We can use the port we have chosen, or a list of UDP ports that are likely not being used, and query all the IP range of an attacked network. Getting a reply back would reveal a live host. No reply would mean a filtering device is covering those hosts UDP traffic, and probably other protocols as well.

[19] HPING2 written by antirez, http://www.kyuzz.org/antirez/hping/ .

3.5 Using Packets bigger than the PMTU of internal routers to elicit an ICMP Fragmentation Needed and Don't Fragment Bit was Set (configuration problem)

If internal routers have a PMTU that is smaller than the PMTU for a path going through the border router, those routers would elicit an ICMP "Fragmentation Needed and Don't Fragment Bit was Set" error message back to the initiating host if receiving a packet too big to process that has the Don't Fragment Bit set on the IP Header, discovering internal architecture of the router deployment of the attacked network.

This is in my opinion a configuration problem causing a security hazard.



A configuration Error example. If internal Routers are configured with MTU smaller than the MTU the border router has, sending packets with the Don't Fragment bit set that are small enough to pass the border router but are bigger than the MTU on an internal Router would reveal its existence.

Figure 6: Using Packets bigger than the PMTU of internal routers to elicit an ICMP Fragmentation Needed and Don't Fragment Bit was Set

4.0 Inverse Mapping Using ICMP (ECHO & ECHO Reply)

Inverse Mapping is a technique used to map internal networks or hosts that are protected by a filtering devices/firewall. Usually some of those systems are not reachable from the Internet. We use routers, which will give away internal architecture information of a network, even if the question they were asked does not make any sense, for this scanning type. We compile a list of IP's that list what is not there and use it to conclude were things probably are.

A router looks at the IP address and makes decisions based on that solely.

We use two ICMP message types in order to use this technique. ICMP ECHO and ICMP ECHO Reply. We send a number of ICMP ECHO / ICMP ECHO Reply datagrams to different IP's we suspect are in the IP range of the network we are probing. When a router, either an exterior or interior, gets those ICMP message types for further processing, it looks at the IP address and makes decisions of routing based on it solely. When a router gets a datagram with an IP which is not used in the IP space / network segment of the part of the probed network he serves, the router will elicit an ICMP Host Unreachable (Generated by a router if a route to the destination host on a directly connected network is not available - does not respond to ARP) or ICMP Time Exceeded (Because the amount of time the Router waits for determining the destination host is unavailable have not been reached yet, but the TTL timer have turned 0 because of the time we wait for an answer) error message(s) back to the originator of the datagram. If we do not get an answer about a certain IP we can assume this IP exist inside the probed network[20].

We are using the ICMP ECHO Reply datagrams because most of the firewalls will let them pass through.

```
[root@cartman]# ./icmpush -vv -echo Target_IP[21]
 -> Outgoing interface = 192.168.1.5
 -> ICMP total size = 12 bytes
 -> Outgoing interface = 192.168.1.5
 -> MTU = 1500 bytes
 -> Total packet size (ICMP + IP) = 32 bytes
ICMP Echo Request packet sent to Target_IP (Target_IP)

Receiving ICMP replies ...
-----------------------------------------------------
Routers_IP  ...
         Type = Time Exceeded (0xB)
   Code = 0x0     Checksum = 0xF98F
     Id = 0x0         Seq# = 0x0
-----------------------------------------------------
./icmpush: Program finished OK


CMP TTL:254 TOS:0x0 ID:13170
ID:12291   Seq:317  ECHO


02/13-09:16:31.724400 Routers_IP -> 192.168.1.5
ICMP TTL:57 TOS:0x0 ID:7410
TTL EXCEEDED
```

[20] There is also a possibility that a filtering device is blocking our probes, or the replies.

[21] The real IP's of the targeted host and the Router were replaced because of legal problems that might arise when the ISP's personal that was used would understand it was one of their Routers used for this experiment.

```
00:13:12 prober> 192.168.2.5: icmp: echo reply
00:13:13 router> prober: icmp: host unreachable
```

Theoretically speaking, using any ICMP type in order to inverse map a network using a Router is possible. The downside would be that some Routers would filter unwanted traffic of certain ICMP types.

5.0 Using traceroute to Map a Network Topology

Traceroute is a Network debugging utility, which attempts to map all the hosts on a route to a certain destination host/machine.

The program sends UDP (by default) or ICMP ECHO Request[22] datagrams in sets of three, to a certain destination host. The first three datagram's to be sent have a Time-to-Live field value in the IP Header equals to one. The program lies on the fact that a router should decrement the TTL field value just before forwarding the datagram to another router/gateway.

If a router discovers that the Time-To-Live field value in an IP header of a datagram he process equals zero (or less) he would discard the datagram and generate an ICMP Time Exceeded Code 0 – transit TTL expired error message back to the originating host.

This is when a successful round is completed and another set of three datagrams is sent, this time with a Time-to-Live field value greater by one than the last set.

The originating host would know at which router the datagram expired since it receives this information with the ICMP Time Exceeded in Transit error message (Source IP address of the ICMP error message would be the IP address of the router/gateway; inside the IP header + 64 bits of original data of the datagram field we would have additional informaiton that would bound this ICMP error message to our issued traceroute command).

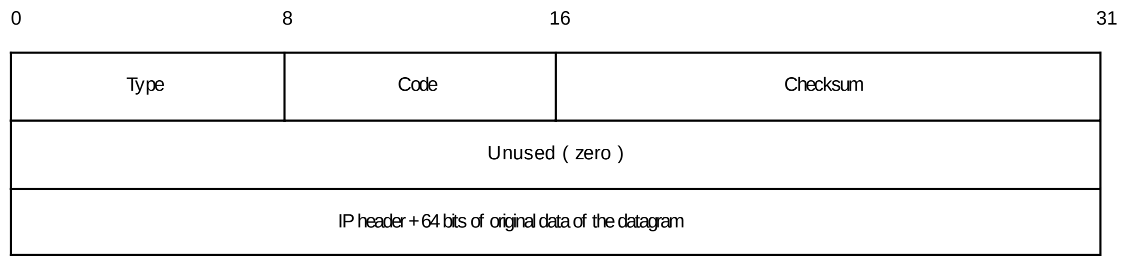


Figure 7: ICMP Time Exceeded message format

Since we increment the TTL field starting from one for each successful round (again - a round is finished when the ICMP Time Exceeded in Transit error message is received) until we receive an ICMP Port Unreachable error message (or ICMP ECHO Reply if we are using the ICMP ECHO request datagrams) from the destined machine, we map every router/gateway/host along the path to our destination.

By default, when sending UDP packets we use a destination port which is probably not used by the destination host so the UDP datagram would not be processes and an ICMP Port Unreachable error message would be generated from the destined machine. The destination port would be incremented with each probe sent.

We get ICMP responses provided there is no prohibitive filtering or any packet loss.

[22] Microsoft Windows NT and Microsoft Windows 2000 are using the tracert command, which use ICMP ECHO Request datagrams as its default.

The output we see is a line showing the Time-To-Live, the address of the gateway, and the round trip time of each probe. If we do not get a response back within 5 seconds an "*" is printed, which represents no answer.

A regular traceroute example with ICMP would be[23]:

```
zuul:~>traceroute –I 10.0.0.10
traceroute to 10.0.0.10 (10.0.0.10), 30 hops max, 40 byte
packets
1 10.0.0.1 (10.0.0.1) 0.540 ms 0.394 ms 0.397 ms
2 10.0.0.2 (10.0.0.2) 2.455 ms 2.479 ms 2.512 ms
3 10.0.0.3 (10.0.0.3) 4.812 ms 4.780 ms 4.747 ms
4 10.0.0.4 (10.0.0.4) 5.010 ms 4.903 ms 4.980 ms
5 10.0.0.5 (10.0.0.5) 5.520 ms 5.809 ms 6.061 ms
6 10.0.0.6 (10.0.0.6) 9.584 ms 21.754 ms 20.530 ms
7 10.0.0.7 (10.0.0.7) 89.889 ms 79.719 ms 85.918 ms
8 10.0.0.8 (10.0.0.8) 92.605 ms 80.361 ms 94.336 ms
9 10.0.0.9 (10.0.0.9) 94.127 ms 81.764 ms 96.476 ms
10 10.0.0.10 (10.0.0.10) 96.012 ms 98.224 ms 99.312 ms
```

Lets assume that a network is protected by a firewall, which blocks all incoming traffic except for traffic aimed at the DNS Machine's UDP port 53. If we would perform a regular traceroute aimed for the DNS machine's IP address, our UDP datagrams would be sent with a destination port, which is probably not used on the targeted machine, and probably blocked by a Firewall or another filtering device. The traces would stop at the firewall at the entrance point to the probed network.

```
zuul:~>traceroute 10.0.0.10
traceroute to 10.0.0.10 (10.0.0.10), 30 hops max, 40 byte
packets
1 10.0.0.1 (10.0.0.1) 0.540 ms 0.394 ms 0.397 ms
2 10.0.0.2 (10.0.0.2) 2.455 ms 2.479 ms 2.512 ms
3 10.0.0.3 (10.0.0.3) 4.812 ms 4.780 ms 4.747 ms
4 10.0.0.4 (10.0.0.4) 5.010 ms 4.903 ms 4.980 ms
5 10.0.0.5 (10.0.0.5) 5.520 ms 5.809 ms 6.061 ms
6 10.0.0.6 (10.0.0.6) 9.584 ms 21.754 ms 20.530 ms
7 10.0.0.7 (10.0.0.7) 89.889 ms 79.719 ms 85.918 ms
8 10.0.0.8 (10.0.0.8) 92.605 ms 80.361 ms 94.336 ms
9 * * *
10 * * *
```

We need to set the port number to 53 in order to reach the DNS server. Since the traceroute program increases the port number every time it sends a UDP datagram, we need to calculate the port number to start with, so when a datagram would be processed by the Firewall[24] and would be examined, it would have the appropriate port and other information needed to fit with the Access Control List. If we use a simple equation we can calculate the starting port:

(Target port – (number of hops * number of probes)) -1

The number of hops (gateways) from our probing machine to the firewall is taken from our earlier traceroute. We use three probes for every query with the same TTL value, each one of them uses a different destination port number.

[23] All examples taken from "A Traceroute-Like Analysis of IP Packet Responses to Determine Gateway Access Control Lists" by David Goldsmith and Michael Shiffman. No real examples were provided because of legal issues.

[24] A firewall should not elicit any reply for any traffic destined directly for him.

```
zuul:~>traceroute -p28 10.0.0.10
traceroute to 10.0.0.10 (10.0.0.10), 30 hops max, 40 byte packets
1 10.0.0.1 (10.0.0.1) 0.501 ms 0.399 ms 0.395 ms
2 10.0.0.2 (10.0.0.2) 2.433 ms 2.940 ms 2.481 ms
3 10.0.0.3 (10.0.0.3) 4.790 ms 4.830 ms 4.885 ms
4 10.0.0.4 (10.0.0.4) 5.196 ms 5.127 ms 4.733 ms
5 10.0.0.5 (10.0.0.5) 5.650 ms 5.551 ms 6.165 ms
6 10.0.0.6 (10.0.0.6) 7.820 ms 20.554 ms 19.525 ms
7 10.0.0.7 (10.0.0.7) 88.552 ms 90.006 ms 93.447 ms
8 10.0.0.8 (10.0.0.8) 92.009 ms 94.855 ms 88.122 ms
9 10.0.0.9 (10.0.0.9) 101.163 ms * *
10 * * *
```

But with the regular traceroute program we now face another difficulty. After the datagram have passed the ACL of the Firewall (and we assume the firewall lets ICMP TTL Exceeded messages out) and listed the outer leg of the Firewall itself as the next hop, the next UDP datagram sent would be with a different port number - Than again it would be blocked by the firewall.

A modification to the traceroute program has been made by Michael Shiffman[25] in order to stop the port incrementation. One side affect from sending traceroutes with a fixed port number, which is allowed on the firewalls ACL, is the final datagram, which normally would generate an ICMP Port Unreachable message now would not be generated since the UDP port would be in a listening state on the probed machine and would not provide an answer.

```
zuul:~>traceroute -S –p53 10.0.0.15
traceroute to 10.0.0.15 (10.0.0.15), 30 hops max, 40 byte
packets
1 10.0.0.1 (10.0.0.1) 0.516 ms 0.396 ms 0.390 ms
2 10.0.0.2 (10.0.0.2) 2.516 ms 2.476 ms 2.431 ms
3 10.0.0.3 (10.0.0.3) 5.060 ms 4.848 ms 4.721 ms
4 10.0.0.4 (10.0.0.4) 5.019 ms 4.694 ms 4.973 ms
5 10.0.0.5 (10.0.0.5) 6.097 ms 5.856 ms 6.002 ms
6 10.0.0.6 (10.0.0.6) 19.257 ms 9.002 ms 21.797 ms
7 10.0.0.7 (10.0.0.7) 84.753 ms * *
8 10.0.0.8 (10.0.0.8) 96.864 ms 98.006 ms 95.491 ms
9 10.0.0.9 (10.0.0.9) 94.300 ms * 96.549 ms
10 10.0.0.10 (10.0.0.10) 101.257 ms 107.164 ms 103.318 ms
11 10.0.0.11 (10.0.0.11) 102.847 ms 110.158 ms *
12 10.0.0.12 (10.0.0.12) 192.196 ms 185.265 ms *
13 10.0.0.13 (10.0.0.13) 168.151 ms 183.238 ms 183.458 ms
14 10.0.0.14 (10.0.0.14) 218.972 ms 209.388 ms 195.686 ms
15 10.0.0.15 (10.0.0.15) 236.102 ms 237.208 ms 230.185 ms
```

[25] http://www.packetfactory.net

6.0 The usage of ICMP in the OS Finger Printing Process

Finger Printing is the art of Operating System Detection.

A malicious computer attacker needs few pieces of information before lunching an attack. First, a target, a host detected using a host detection method. The next piece of information would be the services that are running on that host. This would be done with one of the Port Scanning methods. The last piece of information would be the operating system used by the host.

The information would allow the malicious computer attacker to identify if the targeted host is vulnerable to a certain exploit aimed to a certain service version running on a certain operating system.

I have outlined in this section the ICMP methods for this type of scan. One method is new – "Using wrong codes within ICMP datagrams".

6.1 Using Wrong Codes within ICMP datagrams

An interesting detail I have discovered during the lab experiments I did when I researched ICMP scanning is when a wrong code is sent along with the correct type of ICMP message, different operating systems would send different codes back.

In the next example I have sent an ICMP Timestamp Request with code 38 instead of code 0 to a LINUX machine running Redhat LINUX 6.2 Kernel 2.2.14 (it was experimented with kernel 2.2.12 as well). The LINUX machine processed the packet and sent the reply, with the code value set to 38. I was thinking that a check for the validity of the code field should be done on the targeted machine. Obviously I was wrong.

```
[root@stan /root]# icmpush -vv -tstamp -c 38 192.168.5.5
 -> Outgoing interface = 192.168.5.1
 -> ICMP total size = 20 bytes
 -> Outgoing interface = 192.168.5.1
 -> MTU = 1500 bytes
 -> Total packet size (ICMP + IP) = 40 bytes
ICMP Timestamp Request packet sent to 192.168.5.5 (192.168.5.5)

Receiving ICMP replies ...
kenny.sys-security.com -> Timestamp Reply transmited at 18:06:40
icmpush: Program finished OK
```

```
02/14-18:10:31.951977 192.168.5.1 -> 192.168.5.5
ICMP TTL:254 TOS:0x0 ID:13170
TIMESTAMP REQUEST
1D 04 9D 20 03 78 8C 8B 00 00 00 00 00 00 00 00  ... .x..........

02/14-18:10:31.952233 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0x0 ID:220
TIMESTAMP REPLY
1D 04 9D 20 03 78 8C 8B 03 75 03 00 03 75 03 00  ... .x...u...u..
8C 21 01 00 8C 21                                .!...!
```

I was looking for other ICMP query types, which the Microsoft Windows machine I had on my lab can answer, since Microsoft Windows machines do not answer ICMP Timestamp request messages. I used ICMP ECHO Request datagrams.

I have queried my LINUX box - LINUX Replied with code value set to 38 again. We can look at the tcpdump –x logs, the type and code fields are in bold type:

```
10:06:02.329509   lo < localhost.localdomain > localhost.localdomain:
icmp: echo request
                         4500 0020 3372 0000 fe01 0610 c0a8 0105
                         c0a8 0105 0826 675a 7402 0e20 0186 0cd7

10:06:02.329639   lo > localhost.localdomain > localhost.localdomain:
icmp: echo reply
                         4500 0020 096d 0000 ff01 2f15 c0a8 0105
                         c0a8 0105 0026 6f5a 7402 0e20 0186 0cd7
```

If we examine what RFC 972 requires, we see that LINUX does exactly that.

The sending side initializes the identifier (used to identify ECHO requests aimed at different destination hosts) and sequence number (if multiple ECHO requests are sent to the same destination host), adds some data (arbitrary) to the data field and sends the ICMP ECHO Request to the destination host. *In the ICMP header the code equals zero*. The recipient should *only change* the type to ECHO Reply and return the datagram to the sender.

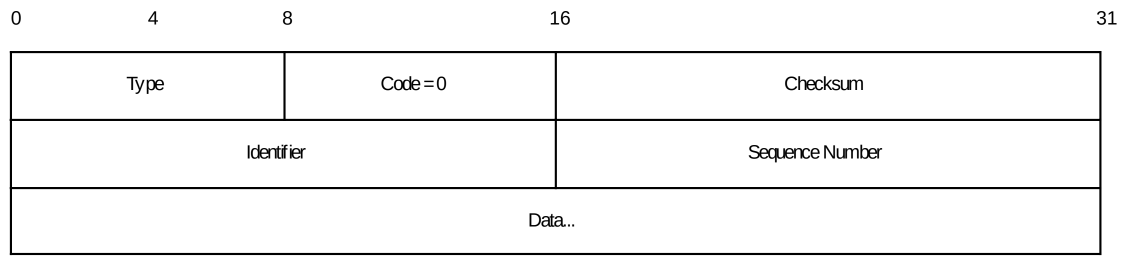


Figure 8: ICMP ECHO Request & Reply message format

This also means that we trust another machine to behave correctly.

LINUX changes the type field value to 0 and sends the reply.

I have checked the behavior of my Microsoft Windows 2000 Professional box. I have sent the same ICMP ECHO Request message to the Microsoft Windows box:

```
10:03:33.860212 eth0 > localhost.localdomain > 192.168.1.1: icmp: echo
request
                         4500 0020 3372 0000 fe01 0614 c0a8 0105
                         c0a8 0101 0826 d618 6102 f658 0183 c8e2
```

```
10:03:33.860689 eth0 < 192.168.1.1 > localhost.localdomain: icmp: echo
reply
                         4500 0020 2010 0000 8001 9776 c0a8 0101
                         c0a8 0105 0000 de3e 6102 f658 0183 c8e2
                         0000 0000 0000 0000 0000 0000 0000
```

The Microsoft Windows 2000 Professional operating system changed the code value on the ICMP ECHO Reply to 0.

I have tested this method with IBM AIX 4.1, SUN Solaris 2.6 & 2.7, OpenBSD, NetBSD, FreeBSD and they produced the same results as the LINUX box did.

Microsoft Windows NT 4.0 Server SP 6a, Microsoft Windows 98 SE produced the same behavior as the Microsoft Windows 2000 Professional.

We have a method to differentiate between a Microsoft Windows box to the rest of the world.

6.2 ICMP error Message Quenching

RFC 1812 suggests limiting the rate at which various error messages are sent.
Only few operating systems are known to follow this RFC.

An attacker can use this to send UDP packets to a random, high UDP port and count the number of ICMP Destination unreachable messages received within a given amount of time.

6.3 ICMP Message Quoting

Every ICMP error message includes the Internet Protocol (IP) Header and *at least* the first 8 data bytes of the datagram that triggered the error; more than 8 octets (bytes) *may* be sent.

Except for LINUX and Solaris almost all implementations will quote 8 bytes of the datagram that triggered the error message. Solaris sends more information than is needed and *Linux even more*.

The following example is a snort log of a LINUX machine (LINUX 6.1 Kernel 2.2.12) that have generated a Port Unreachable ICMP error message:

```
03/01-12:29:39.259510 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0xDE ID:149
DESTINATION UNREACHABLE: PROTOCOL UNREACHABLE
00 00 00 00 45 7E 04 32 00 0D 00 00 89 70 A1 7A  ....E~.2.....p.z
C0 A8 05 01 C0 A8 05 05 FE 94 6C 95 59 F2 D9 3C  ..........l.Y..<
8D AA B6 0B 2B 80 CB 8B 89 4D C9 59 19 D6 0F A0  ....+....M.Y....
D3 67 D1 0F CB ED 84 8C 91 7E 24 00 70 B9 D7 E4  .g.......~$.p...
6E AA 91 8F CF 5C ED 86 1B A2 40 1D 93 10 73 4B  n....\....@...sK
49 5B A8 D5 91 99 47 F0 15 6B EB 8B 21 2D A2 15  I[....G..k..!-..
A1 97 4C AD 6D A1 2B E5 15 07 86 77 3A 85 E9 6E  ..L.m.+....w:..n
58 87 05 73 6D FB E9 05 29 73 DD B4 C0 EA 98 1D  X..sm...)s......
6E 44 8F 47 85 A4 89 E6 CF 64 18 B5 FD 31 19 C0  nD.G.....d...1..
C0 8A 8E CB 60 B0 D5 F5 79 57 81 DD 78 0B 1B EF  ....`...yW..x...
CE 8A E5 AC 46 D4 E3 91 6C 24 80 59 CC 00 C4 AB  ....F...l$.Y....
86 CC 39 FC AD B1 AF 3F 16 B1 6D 9C 47 5D 85 F5  ..9....?..m.G]..
FC E3 CC 01 0E DC CC 48 E4 B6 0B 0E E5 08 A5 41  .......H.......A
9A D9 45 B9 7A 37 13 31 C7 96 F2 42 2E 20 95 21  ..E.z7.1...B. .!
```

```
D8 EF 74 F4 78 B3 44 14 F5 4D 45 B4 08 C0 7B 1A  ..t.x.D..ME...{.
7E B0 B5 71 2A 5A 95 61 22 0E 72 B7 1A 57 1E F2  ~..q*Z.a".r..W..
3E B9 28 33 EA 3A 23 70 34 41 CF 43 C8 B1 CE 1A  >.(3.:#p4A.C....
15 FD 42 E9 E1 4B DC 93 35 2C 10 6C 71 B5 0D 1C  ..B..K..5,.lq...
84 60 E9 68 51 30 79 AE 2E 1D 59 F0 F4 C8 AD CD  .`.hQ0y...Y.....
0E 62 1F 23 42 2F 30 70 91 DA 5C 86 4E 62 CF 93  .b.#B/0p..\.Nb..
84 B9 39 9D F2 03 B8 FA 08 E1 BA B5 86 15 1D DE  ..9.............
FD 9E 68 61 F9 71 32 CB 78 CD 6A 27 3F E7 FC 2D  ..ha.q2.x.j'?..-
54 90 90 17 76 DC 82 AD E9 07 6A A5 2F 7B F7 69  T...v.....j./{.i
89 C8 71 AA 27 DA 1A A3 CD 30 75 3C EA 36 52 EA  ..q.'....0u<.6R.
AE D9 DC 3A 0A E5 B7 BA 97 F0 91 FA D4 98 94 8F  ...:............
F9 5B CE 0A C6 5A 71 29 38 32 05 42 6D 57 8C C2  .[...Zq)82.BmW..
95 59 E3 33 0F 70 7E 61 4E D9 3E EB 75 CB D7 A1  .Y.3.p~aN.>.u...
B0 95 9C A5 F2 44 7D C6 11 E2 DC 7B CF B0 C0 BB  .....D}....{....
B8 B6 DA 95 77 76 4F A7 6B 90 4B 0F E3 36 64 EC  ....wvO.k.K..6d.
19 1A A9 91 D5 15 52 4C AE D3 42 6D DE 0E 43 2D  ......RL..Bm..C-
26 A1 ED 7E C1 8E 74 7A 2C 6A 36 5A 4B 1C DC FF  &..~..tz,j6ZK...
D2 FF 3D 61 59 C6 E4 E1 19 DD 29 77 A4 9D D2 93  ..=aY.....)w....
03 0D 1B 14 21 3B 6E 9D 66 23 05 72 D2 89 80 3D  ....!;n.f#.r...=
AE 03 A7 9F D2 89 5D D7 E9 0C B0 98 A0 04 0F AE  ......].........
9E 17 62 93 83 28 CA 81                          ..b..(..
```

This technique allows us to identify Solaris & LINUX machines even if there is no port opened.

6.4 ICMP Error Message Echoing Integrity

When sending back an ICMP error message, some stack implementations may alter the IP header.

If an attacker examines the types of alternation that have been made to the headers, he may be able to make certain assumptions about the target operating system.

Fyodor gives the following examples in his article "Remote OS detection via TCP/IP Stack Finger Printing"[26]:

> "For example, AIX and BSDI send back an IP 'total length' field that is 20 bytes too high. Some BSDI, FreeBSD, OpenBSD, ULTRIX, and VAXen change the IP ID that you sent them. While the checksum is going to change due to the changed TTL anyway, there are some machines (AIX, FreeBSD, etc.) which send back an inconsistent or 0 checksum. Same thing goes with the UDP checksum."

6.5 TOS Field in ICMP Port Unreachable Message

Nearly all stack implementations send back 0x00 as the TOS value when generating an ICMP Port Unreachable Message. All but LINUX, which sends the value of 0xc0.

In the next example we have sent one UDP packet destined to port 50 (which is closed on the destination machine) from one LINUX machine to another, both running Redhat LINUX 6.1:

[26] http://www.insecure.org/nmap/nmap-fingerprinting-article.html

```
[root@stan /root]# hping2 -2 192.168.5.5 -p 50 -c 1
default routing not present
HPING 192.168.5.5 (eth0 192.168.5.5): udp mode set, 28 headers + 0 data
bytes
ICMP Port Unreachable from 192.168.5.5  (kenny.sys-security.com)

--- 192.168.5.5 hping statistic ---
1 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

-*> Snort! <*-
Version 1.5
By Martin Roesch (roesch@clark.net, www.clark.net/~roesch)
Kernel filter, protocol ALL, raw packet socket
Decoding Ethernet on interface eth0
03/12-12:54:47.274096 192.168.5.1:2420 -> 192.168.5.5:50
UDP TTL:64 TOS:0x0 ID:57254
Len: 8

03/12-12:54:47.274360 192.168.5.5 -> 192.168.5.1
ICMP TTL:255 TOS:0xC0 ID:0
DESTINATION UNREACHABLE: PORT UNREACHABLE
00 00 00 00 45 00 00 1C DF A6 00 00 40 11 0F D4  ....E.......@...
C0 A8 05 01 C0 A8 05 05 09 74 00 32 00 08 6A E1  .........t.2..j.
```

7.0 Filtering ICMP on your Filtering Device to Prevent Scanning Using ICMP

7.1 Inbound

Incoming ICMP traffic that should be blocked in order to prevent scanning techniques that were outlined in this paper are:

- ICMP ECHO (used for Host Detection, traceroute & Inverse Mapping)
- ICMP ECHO Reply (used for Inverse Mapping)
- ICMP Time Stamp Request (used for Host Detection)
- ICMP Address Mask Request (used for Host Detection)
- All ICMP Message Types (Inverse Mapping Technique)

You should also block the IP directed broadcast on your border router.
Deny access to your Broadcast and Network addresses from the Internet.

7.2 Outbound

There are people who claim that any traffic type of ICMP should be allowed from a protected network to the Internet. This is not true. Filtering the incoming traffic does not mean we are protected from some of the security hazards I outlined in this paper.

7.2.1 ICMP ECHO Reply (Type 0)
Used to map a host using Host Detection.

7.2.2 ICMP Destination Unreachable Messages
I have demonstrated that host detection can be done with bad IP Header packets, which elicit various ICMP Parameter Problem and ICMP Destination Unreachable error messages from the probed machines and draw the attacked network topology.

7.2.3 ICMP “Fragmentation Needed and Don’t Fragment Bit was Set”
See section 3.5

7.2.4 ICMP ECHO (Type 8)
We have to have a Stateful filtering device that would perform Stateful inspection with ICMP in order to let ICMP ECHO Requests out, and receive only the corresponding ICMP ECHO Replies.

The current state with filtering devices is not that bright. Most of them do not perform Stateful inspection with the ICMP protocol. Allowing ICMP ECHO Replies inside our protected network is very dangerous and is not worth it.

Unless you use a Stateful filtering device with the ICMP protocol don’t let ICMP ECHO Replies into your protected network. This would make your requests useless so you better block them.

7.2.5 ICMP Time to Live Exceeded in Transit (Type 11 Code 0)
To eliminate traceroute and Reverse Mapping techniques we do not want to let a Time-to-Live Exceeded code 0 messages go back to the malicious computer attacker.

7.2.6 ICMP Fragmentation Reassembly Time Exceeded (Type 11 Code 1)

By blocking this ICMP type we eliminate the usage of a Host Detection technique, which sends only few fragments, form a fragmented datagram, and force the probed host to send us an ICMP Fragmentation Reassembly Time Exceeded error message back revealing his existence.

7.2.7 ICMP Parameter Problem

We have demonstrated that host detection can be made with bad IP Header packets, which would elicit various ICMP Parameter Problem and ICMP Destination Unreachable error messages from the probed machines.

7.2.8 ICMP Time Stamp Request & Reply

Time Stamp requests & replies can be used for Host Detection and Inverse Mapping.

7.2.9 ICMP Address Mask Request and Reply

Address Mask request & reply can be used for host detection and Inverse Mapping.

7.2.10 The liability Question

All System administrator / Network administrator don’t want to be held liable for an attack generated from there network by an abusive user (or a malicious computer attacker using a compromised system within the network). Therefore blocking some types of ICMP traffic from the protected network to the outside world is recommended for liability reasons:

- Destination Unreachable Codes 2-4

    - ICMP Destination Unreachable error messages 2-4 (“Port Unreachable”, “Protocol Unreachable” and “Fragmentation Needed and DF Flag was Set”) is a group of messages that are hard error conditions and when received should terminate a connection.

      This allow an attacker to send *fake* Destination Unreachable codes 2-4 to terminate valid connections between the attacked target and other hosts on the void.

      Old TCP/IP implementations terminat TCP connections when receiving those error messages. Modern TCP/IP implementations no longer terminate a TCP connection when receiving those error messages

- Source Quench messages

    - Since hosts still react to Source Quenches by slowing communication, they can be used as a Denial-of-Service measure.

- Redirect messages

    - If you can forge ICMP Redirect packets, and if your target host pays attention to them - ICMP Redirects may be employed for denial of service attacks, where a host is sent a route that loses it connectivity, or is sent an ICMP Network Unreachable packet telling it that it can no longer access a particular network.

This means that all outbound ICMP traffic should be disallowed.

7.3 Other Considerations

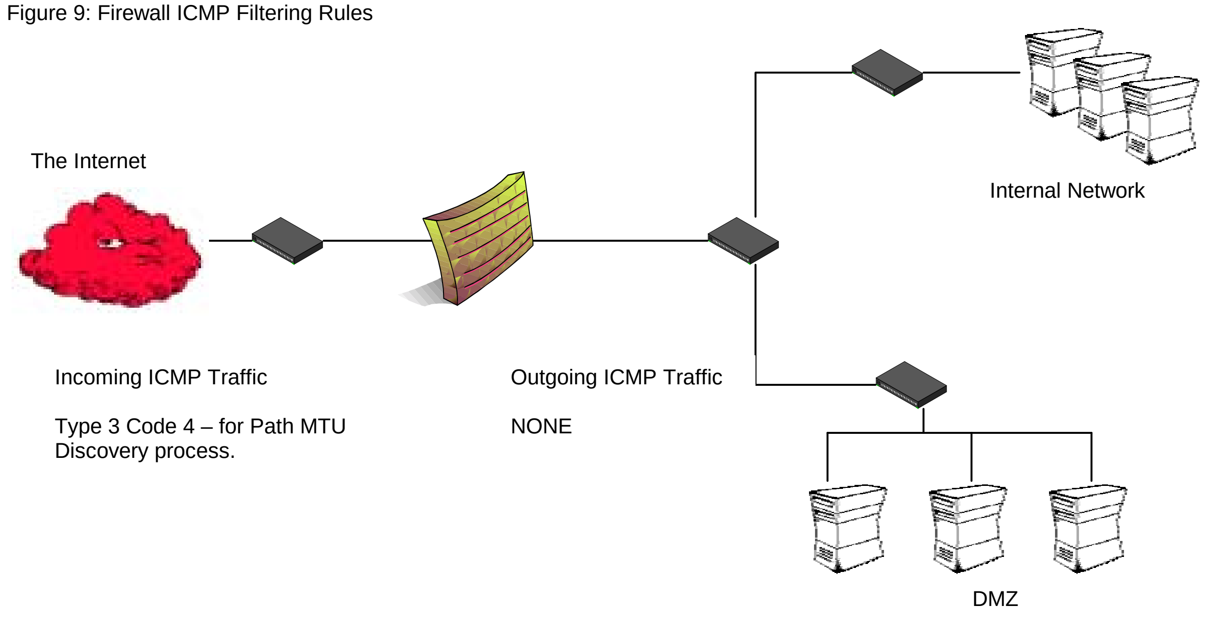
If you want to maintain strong ICMP filtering rules with your Firewall/Filtering-Device I suggest you block all incoming ICMP traffic except for Type 3 Code 4, which is used by the Path MTU Discovery process[27].

If you will block incoming ICMP "Fragmentation Needed and Don't Fragment Bit was Set" your network performance will suffer from degradation. You should understand the security risks involving in opening this kind of traffic to your protected network. The possibility of a DOS, Inverse Mapping, Host Detection, and a one-way Covert communication channel (which was not been seen in the wild yet).

Another consideration could be the usage of network troubleshooting tools such as traceroute and ping. In the case of traceroute if the filtering device you are using does not support Stateful inspection with ICMP than allowing ICMP TTL Exceeded In Transit (Type 11, code 0) error messages inside the protected network could lead to various security hazards. The same goes with ping, where ICMP ECHO reply is even more dangerous when allowed inside the protected network (Inverse Mapping, Covert Channel and more security risks).

You can limit the number of systems that really need to use the network troubleshooting tools with ACL, but bear in mind that those systems could be mapped from the Internet – and this is only the tip of the iceberg.

Figure 9: Firewall ICMP Filtering Rules



[27] See Appendix B: "Fragmentation Needed but the Don't Fragment Bit was set" and the Path MTU Discovery Process.

8.0 Conclusion

The ICMP protocol is a very powerful tool in the hands of smart malicious computer attackers. Mapping and detecting of hosts and networking devices can be done in various ways as I have outlined in this paper.

It is extremely important to understand that ICMP traffic can be used to other malicious activities other than scanning, such as:

- Denial of Service Attacks
- Distributed Denial of Service Attacks
- Covert Channel Communications

Therefore filtering Inbound and Outbound ICMP traffic is very important and may help you in preventing risks to your computing environment.

9.0 Acknowledgment

I would like to thank the following people for their help with/during this research.

Ariel Pisetsky for going over this paper correcting my English, and for his moral support.

Christopher Tresco, Systems Administrator at the Massachusetts Institute of Technology provided necessary test systems to verify my findings.

Special thanks to mr2940 for his patience while I introduced my new ideas.

James Cudney, Michael, Pat, for their support when the times where bad.

Appendix A: The ICMP Protocol[28]

Internet **C**ontrol **M**essage **P**rotocol (ICMP) is used when a *router* or a *destination host* need to inform the source host about errors in a datagram processing.

Some of ICMP's characteristics are:

- ICMP uses IP as if it were a higher-level protocol, however, ICMP is already an internal part of IP, and must be implemented by every IP module.
- ICMP is used to provide feedback about some errors in a datagram processing, not to make IP reliable. Datagrams may still be undelivered without any report of their loss. If a higher level protocol that use IP need reliability he must implement it.
- No ICMP messages are sent in response to ICMP messages to avoid infinite repetitions. The exception is a response to ICMP query messages (ICMP Types 0,8-10,13-18. See Table 1 ICMP Query Messages).
- For fragmented IP datagrams ICMP messages are only sent about errors on fragment zero (first fragment).
- ICMP error messages are never sent in response to a datagram that is *destined* to a *broadcast* or a *multicast* address.
- ICMP error messages are never sent in response to a datagram sent as a link layer broadcast.
- ICMP error messages are never sent in response to a datagram whose source address does not represents a unique host – the source IP address cannot be *zero*, a *loopback* address, a *broadcast* address or a *multicast* address.
- When an ICMP message of ***unknown type*** is received, it must be silently *discarded*.
- Routers will almost always generate ICMP messages but when it comes to a destination host(s), the number of ICMP messages generated is implementation dependent.

ICMP Query Messages	***ICMP error Messages***
ECHO	*Destination Unreachable*
Router Advertisement	*Source Quench*
Router Solicitation	*Redirect*
Time Stamp	*Time Exceeded*
Information	*Parameter Problem*
Address Mask	

Table 1: ICMP message types

[28] ICMP is described in RFC 972 (http://www.ietf.org/rfc/rfc0972.txt) with updates in RFC 950 (http://www.ietf.org/rfc/rfc0950.txt).

A.1 ICMP Messages

ICMP messages are sent in IP datagrams. The protocol number will be always one (ICMP), and the Type-of-Service will be zero. The IP data field will contain the actual ICMP message:

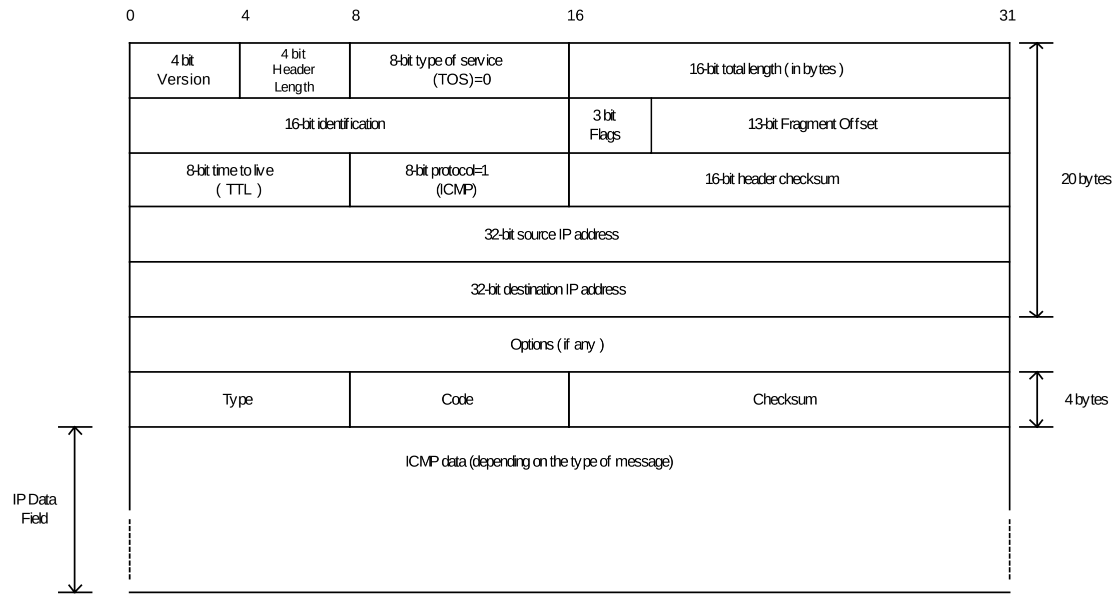


Figure 10: ICMP Message Format

ICMP error message length

Every ICMP error message includes the Internet (IP) Header and *at least* the first 8 data octets (bytes) of the datagram that triggered the error; more than 8 octets (bytes) *may* be sent; this header and data must be unchanged from the received datagram.

The **TYPE** field specifies the type of the message, while the error code for the datagram reported on by this ICMP message is contained in the **CODE** field. The code interpretation is dependent upon the message type.

Type	Name	Code
0	ECHO Reply	0 No Code
1	Unassigned	
2	Unassigned	
3	Destination Unreachable[29]	
		0 Net Unreachable
		1 Host Unreachable
		2 Protocol Unreachable
		3 Port Unreachable
		4 Fragmentation Needed and Don't Fragment was Set
		5 Source Route Failed
		6 Destination Network Unknown
		7 Destination Host Unknown
		8 Source Host Isolated
		9 Communication with Destination Network is Administratively Prohibited
		10 Communication with Destination Host is Administratively Prohibited
		11 Destination Network Unreachable for Type of Service.
		12 Destination Host Unreachable for Type of Service.
		13 Communication Administratively Prohibited.
		14 Host Precedence Violation
		15 Precedence cutoff in effect
4	Source Quench	0 No Code
5	Redirect	
		0 Redirect Datagram for the Network (or subnet)
		1 Redirect Datagram for the Host
		2 Redirect Datagram for the Type of Service and Network
		3 Redirect Datagram for the Type of Service and Host
6	Alternate Host Address	0 Alternate Address for Host
7	Unassigned	
8	Echo	0 No Code
9	Router Advertisement	0 No Code
10	Router Selection	0 No Code
11	Time Exceeded	
		0 Time to Live exceeded in Transit
		1 Fragment Reassembly Time Exceeded
12	Parameter Problem	
		0 Pointer indicates the error
		1 Missing a Required Option
		2 Bad Length
13	Timestamp	0 No Code
14	Timestamp Reply	0 No Code

[29] RFC 972 defines codes 1-5. RFC 1122 defines codes 6-12. RFC 1812 defines codes 13-15.

Type	Name	Code	
15	Information Request	0	No Code
16	Information Reply	0	No Code
17	Address Mask Request	0	No Code
18	Address Mask Reply	0	No Code
19	Reserved (for Security)	0	No Code
	20-29 reserved (for Robustness Experiment)		
30	Traceroute		
31	Datagram Conversion Error		
32	Mobile Host Redirect		
33	IPv6 Where-Are-You		
34	IPv6 I-Am-Here		
35	Mobile Registration Request		
36	Mobile Registration Reply		
39	SKIP		
40	Photuris		
		0	Reserved
		1	unknown security parameters index
		2	valid security parameters, but authentication failed
		3	valid security parameters, but decryption failed

Table 2: ICMP Types & Codes

Checksum – contains the 16bit one's complement of the one's complement sum of the ICMP message starting with the ICMP Type field. For computing this checksum, the checksum field is assumed to be zero.

Data – Will contain a part of the original IP message for which this ICMP message was generated. The length of the DATA field equals the IP datagram length less the IP header length. Every ICMP error message includes the Internet (IP) Header and *at least* the first 8 data octets (bytes) of the datagram that triggered the error; more than 8 octets (bytes) *may* be sent; this header and data must be unchanged from the received datagram.

Appendix B: ICMP "Fragmentation Needed but the Don't Fragment Bit was set" and the Path MTU Discovery Process [30]

When one host needs to send data to another host, the data is transmitted in a series of IP datagrams. We wish the datagrams be the largest size possible that does not require fragmentation[31] along the path from the source host to the destination host.

Fragmentation by the IP layer raises few problems:

- If one fragment from a packet is dropped, we need to retransmit the whole packet.
- Load on the routers, which needs to do the fragmentation.
- Some simpler firewalls would block all fragments because they do not contain the header information for a higher layer protocol needed for filtering.

The **M**aximum **T**ransfer **U**nit (***MTU***) is a link layer restriction on the maximum number of bytes of data in a single transmission. The smallest MTU of any link on the current path between two hosts is called the ***Path MTU***.

B.1 The PATH MTU Discovery Process

We use the Don't Fragment Bit Flag in the IP header to dynamically discover the Path MTU of a given route. The source host assumes that the PMTU of a path is the known MTU of its first hop. He will send all datagrams with that size, and set the Don't Fragment Bit. If along the path to the destination host, there is a router that needs to fragment the datagram in order to pass it to the next hop, an ICMP error message (Type 3 Code 4 "Fragmentation Needed and DF set") will be generated, since the Don't Fragment bit was set. When the sending host receives the ICMP error message he should reduce his assumed PMTU for the path.

The process can end when the estimated PMTU is low enough for the datagrams not to be fragmented. The source host itself can stop the process if he is willing to have the datagrams fragmented in some circumstances.

Usually the DF bit would be set in all datagrams, so if a route changes to the destination host, and the PMTU is lowered, than we would discover it.

The PMTU of a path might be increased over time, again because of a change in the routing topology. To detect it, a host should periodically increase its assumed PMTU for that link.

The link MTU field in the ICMP "Fragmentation Needed and DF set" error message, carries the MTU of the constricting hop, enabling the source host to know the exact value he needs to set the PMTU for that path to allow the voyage of the datagrams beyond that point (router) without fragmentation.

B.2 Host specification

A host must reduce his estimated PMTU for the relevant path when he receives the ICMP "Fragmentation Needed and the DF bit was set" error message. RFC 1191 does not outline a specific behavior that is expected from the sending host, because different applications may have different requirements, and different implementation architectures may favor different strategies.

[30] RFC 1191, http://www.ietf.org/rfc/rfc1191.txt, J. Mogul, S. Deering.

[31] When we send a packet that it is too large to be sent across a link as a single unit, a router needs to slice/split the packet into smaller parts, which contain enough information for the receiver to reassemble them. This is called fragmentation.

The only required behavior is that a host *must* attempt to avoid sending more messages with the same PMTU value in the near future. A host can either cease setting the Don't Fragment bit in the IP header (and allow fragmentation by the routers in the way) or reduce the datagram size. The better strategy would be to lower the message size because fragmentation will cause more traffic and consume more Internet resources.

A host using the PMTU Discovery process *must* detect decreases in Path MTU as fast as possible. A host *may* detect increases in Path MTU, by sending datagrams larger than the current estimated PMTU, which will usually be rejected by some router on the path to a destination since the PMTU usually will not increase. Since this would generate traffic back to the host, the check for the increases must be done at infrequent intervals. The RFC specify that an attempt for detecting an increasment *must not* be done less than 10 minutes after a datagram Too Big has been received for the given destination, or less than 2 minute after a previously successful attempt to increase.

The sending host must know how to handle an ICMP "Fragmentation Needed and the DF bit was set" error message that was sent by a device who does not know how to handle the PMTU protocol and does not include the next-hop MTU in the error message. Several strategies are available:

- The PMTU should be set to the minimum between the currently assumed PMTU and 576[32]. The DF bit should not be set in future datagrams for that path.
- Searching for the accurate value for the PMTU for a path. We keep sending datagrams with the DF bit set with lowered PMTU until we do not receive errors.

A host must not reduce the estimation of a Path MTU value below 68 bytes.

A host MUST not increase its estimate of the Path MTU in response to the contents of a Datagram Too Big message.

B.3 Router Specification

When a router cannot forward a datagram because it exceeded the MTU of the next-hop network and the Don't Fragment bit was set, he is required to generate an ICMP Destination Unreachable message to the source of the datagram., with the appropriate code indicating "Fragmentation needed and the Don't Fragment Bit was set". In the error message the router *must* include the MTU of the next-hop in a 16bit field inside the error message.

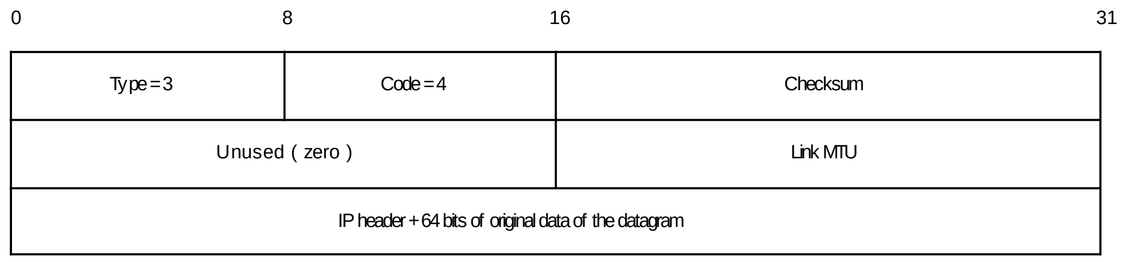


Figure 11: ICMP Fragmentation Required with Link MTU

[32] The usage of the lesser between 576 and the first-hop MTU as the PMTU for a destination, which is not connected to the same network was the old implementation. The results were the use of smaller datagrams than necessary, waste of Internet resources, and not being optimal.

The value of the next-hop MTU field should be set to the size in bytes of the largest datagram that could be forwarded, along the path of the original datagram, without being fragmented by this router. The size includes IP header plus IP data and no lower level headers should be included.

Because every router should be able to forward a datagram of 68 bytes without fragmenting it, the link MTU field should not contain a value less than 68.

B.4 The TCP MSS (Maximum Segment Size) Option and PATH MTU Discovery Process

The RFC specify that a host that is doing Path MTU Discovery *must not* send datagrams larger than 576 bytes unless the receiving host grants him permission.

When we are establishing a TCP connection both sides announce the maximum amount of data in one packet that should be sent by the remote system – The maximum segment size, MSS (if one of the ends does not specify an MSS, it defaults to 536 – there is no permission from the other end to send more than this amount). The packet generated would be, normally, 40 bytes larger than the MSS; 20 bytes for the IP header and 20 bytes for the TCP header. Most systems announce an MSS that is determined from the MTU on the interface that the traffic to the remote system passes out from the system through.

Each side upon receiving the MSS of the other side should not send any segments larger than the MSS received, regardless of the PMTU. After receiving the MSS value the Path MTU Discovery process will start to take affect. We will send our IP packets with the DF bit set allowing us to recognize points in the path to our destination that cannot process packets larger as the MSS of the destination host plus 40 bytes. When such an ICMP error message arrives, we should lower the PMTU to a path (according to the link MTU field, or if not used, to use the rules regarding the old implementation) and retransmit. The value of the link MTU cannot be higher than the MSS of the destination host. When retransmission occurs resulting from ICMP type 3 code 4 error message, the congestion windows should not change, but slow start should be initiated. The process continues until we adjust the correct PMTU of a path (not receiving ICMP error messages from the intermediate routers) which will allow us to fragment at the TCP layer which is much more efficient than at the IP layer.