Albert-Ludwigs-Universität Freiburg
Institut für Informatik
AG CoNe: Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Proseminar Rechnernetze (WS 2007/2008)

Internet Control Message Protocol

Michael Ziegler

Matrikelnummer: 2372998

08.01.2008

Contents

1 Introduction

1.1 What is ICMP?

The IP protocol provides an unreliable, best-effort means of transporting datagrams. A datagram passes multiple routers on its way to the destination host, of which each one is either able to deliver the packet directly, or passes it on to another router. This goes on until the packet finally reaches its destination, or an error causes the packet to be dropped.

If such an error occurs, it is necessary to inform the source host that their data was lost, and why. That is the purpose of the Internet Control Message Protocol (ICMP).

1.2 Functionality and limitations

Whenever a router encounters an error condition of any kind, it sends an ICMP message back to the sender of the packet. This ICMP message is an entirely independent packet which is transported over the network just as any other packet would be, there is no special priority or extra guarantee that the message will ultimately arrive, as it may meet an error condition itself.

Since IP headers only name the addresses of the two end points that are communicating, namely the source and the destination, routers have no choice but to inform the sender of the certain error condition, even if they cannot provide a solution. Still, informing the sender about occurring problems is preferred over silently ignoring them. In cases where the sender is not able to fix the reported issue himself, communication between the network administrators of all affected networks is relied upon, e.g. a home user whose Internet connection is not working anymore is expected to call the ISP's hotline.

However, since an ICMP message is just an ordinary IP packet, it may encounter an error condition itself, e.g. if there is no route back to the sender, that would trigger a new ICMP message. To avoid flooding the network with ICMP messages about ICMP messages about ICMP messages, every router checks whether the erroneous packet was an ICMP message, and if so, silently ignores the error, hoping that eventually, someone will notice about the problem. Furthermore, most network failures are only temporary, so simply resending the data may be sufficient.

1.3 ICMP in the protocol stack

It is important to keep in mind, that ICMP, even though it uses IP packets for transportation of its messages, is not considered to be a higher level protocol, but instead a required part of IP. In order to be able to easily tansport ICMP messages over an IP network, ICMP messages are encapsulated into IP packets though.

Still, since ICMP messages are in fact just IP packets, they may be encapsulated further by lower level protocols, like Ethernet:

		ICMP HEADER (8 bytes)	DATA
	IP HEADER (20 bytes)	ICMP MESSAGE	
FRAME HEADER	IP PACKET		

Table 1.1: Encapsulation of ICMP messages on the Ethernet

1.4 Message format

In general, all ICMP messages follow the same format, with slight variations:

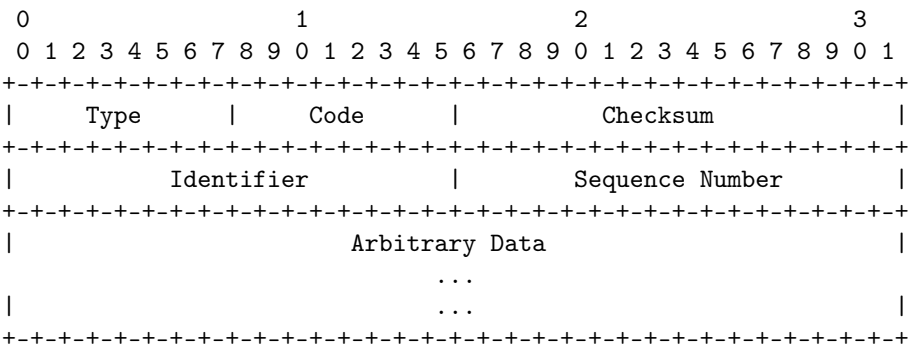

Figure 1.1: Typical ICMP packet structure as defined in RFC792[2]

Some of these fields may be omitted depending on the ICMP message type that is being sent. Whenever a certain field is not used for the respective message type, it is filled with zeroes. The header format is always the same, but ICMP allows for arbitrary data to be appended to the header and transferred in the message. Most messages include the first 64 bytes of the original IP packet, so the sender is able to identify the original IP packet that caused the problem.

Identification of the source packet allows to recognise fake ICMP messages which may be misused for DoS[1] attacks, e.g. by sending Source Quench messages to make the victim stop sending data altogether. All modern IP implementations support message verification.

1.5 Example network structure

In this paper, I will demonstrate several mechanisms using the ping command, which sends ICMP Echo Requests through the network. All of the following examples were run on a Linux machine connected to the Internet via an Ethernet router and a DSL router, using a host located on the Internet as target:

$$10.5.0.197 \overset{Ethernet}{\longleftrightarrow} 10.5.0.1 \overset{Ethernet}{\longleftrightarrow} 192.168.0.1 \overset{DSL(PPPoE)}{\longleftrightarrow} \text{Internet} \overset{Ethernet}{\longleftrightarrow} 85.25.9.79$$

1.6 Listings

The listings in this paper contain commands for the Bourne again Shell (bash) on Linux, but are not intended for copy-and-paste use. Lines starting with a # are comments, lines starting with a $ are commands, and all other lines show the output of the last command.

[1] Denial of Service

2 ICMP Message types

2.1 Message types

The type field in the ICMP header classifies the message as to what it means. The most important message types are:

Type field	Meaning	Relevance
3	Destination unreachable	Crucial
4	Source Quench	Minor
5	Redirect	Deprecated
8, 0	Echo Request/Reply	Crucial
9	Router Advertisement	Deprecated
10	Router Solicitation	Deprecated
11	Time to live exceeded	Crucial
12	Parameter Problem	Minor
13, 14	Timestamp Request/Reply	Deprecated
15, 16	Information Request/Reply	Deprecated
17, 18	Address Mask Request/Reply	Deprecated
30	Traceroute	Minor

Table 2.1: ICMP message types

Many ICMP message types are no longer used since the features they were once intended to provide are nowadays implemented in their own protocols, for instance 15/16/17/18 were superseded by the more flexible DHCP[1] and 13/14 are implemented by NTP[2].

2.2 Type 3: Destination unreachable

When watching the ICMP traffic on a network for some time, one notices that a remarkable amount of all ICMP messages sent are in fact of Type 3. This type of message uses the Code header field to tell the sender what exactly went wrong. The most important codes are listed in Table 2.2.

2.2.1 Code 0: Network unreachable

This error means that the target host is located in a network which is unreachable, so not only this certain host will be affected, but instead all hosts on the target network cannot be reached.

There are mainly two possible causes for this situation:

[1] Dynamic Host Configuration Protocol, RFC 2131
[2] Network Time Protocol, RFC 958

Code field	Meaning
0	Network unreachable
1	Host unreachable
2	Protocol unreachable
3	Port unreachable
4	Fragmentation needed but DF set

Table 2.2: ICMP Type 3: Error codes

- The router sending this message does not have a matching route in its routing table, so it does not know how to reach the target network at all.
- The next hop that is listed in the routing table is unreachable.

2.2.2 Code 1: Host unreachable

Whenever a router receives a packet and its destination IP resides in a LAN directly connected to the router, it sends an ARP request into the LAN to determine the target host's MAC address. If that request fails, the IP address does not exist, so the router sends an ICMP "Host unreachable" message back to the sender.

```
1  # 192.168.0.20 does not exist, 192.168.0.150 is the adress of the router at
2  # 10.5.0.1 in the target network.

4  $ ping 192.168.0.20
5  PING 192.168.0.20 (192.168.0.20) 56(84) bytes of data.
6  From 192.168.0.150 icmp_seq=2 Destination Host Unreachable
7  From 192.168.0.150 icmp_seq=3 Destination Host Unreachable
8  From 192.168.0.150 icmp_seq=4 Destination Host Unreachable

10 --- 192.168.0.20 ping statistics ---
11 4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3000ms
```

Listing 2.1: ICMP Host unreachable

Firewall caveats

Some personal firewalls promise to their users to make their computer invisible to the rest of the network, simply by not responding to ICMP Echo requests. However, this is technically impossible, because ARP requests are still being answered, so the router located before the target host does not send an ICMP Host Unreachable message.

2.2.3 Code 2: Protocol unreachable

This message is hardly ever seen on the Internet, since the Internet is a homogeneous TCP/IP network. It means that the target host does not support the transport protocol (TCP/UDP) that the source host is trying to communicate with. Since all modern operating systems actually support IP, TCP and UDP, this error message is highly unlikely to occur on the Internet.

2.2.4 Code 3: Port unreachable

Whenever a client tries to send data to a UDP port that is closed on the target host, the recipient answers with this error message. This message is not used when the connection is made via TCP, because TCP is connection oriented and has its own means for denying communication. Since

UDP lacks this feature, the sender is informed via ICMP that datagrams sent to this certain port will not be accepted.

Firewall caveats

When configuring a packet filter to deny communication with a service in certain conditions, there are two possible configurations:

- Deny: Packets are simply dropped and ignored.
- Reject: Before the packet is dropped, the sender is informed that connections to this port will not be accepted.

Whenever an administrator chooses to block packets, using the Deny option is strongly discouraged because the sender thinks their packet got lost and keeps re-sending it, unnecessarily consuming bandwith and time on both their and our end. Had we used the Reject option, the sender would know that communication is not allowed and therefore stop trying.

2.2.5 Code 4: Fragmentation needed but DF set

Network interfaces that exchange data between each other have a buffer that stores data which is to be sent or has been received via the physical link before it can actually be processed. This buffer's size is somewhat limited, so the maximum size that e.g. an Ethernet frame can grow to must not exceed the available storage. How much that is depends on the standard the hardware is compatible to, e.g. Ethernet has a maximum frame size of 1526 bytes, where 1500 bytes can be used for payload data. This is called the Maximum Transfer Unit (MTU).

If an IP packet is to be transmitted that exceeds the MTU, it will be fragmented, the fragments will be encapsulated in different frames that are smaller or equal in size than the MTU, transmitted over the network segment and reassembled on the other side. This process wastes both time and bandwith, and it would not be necessary, had the original IP packet not been too large in the first place.

In order to keep our packets small enough to be transferred to the target host without fragmentation, we would first need to know the smallest MTU of all the media the datagrams will have to pass through on their way to the target host, the so called Path MTU. To detect the Path MTU, we can perform Path MTU Discovery.

IP provides a way to prevent the datagram from being fragmented on its way by setting the "Don't fragment" flag in the IP header. Whenever a router encounters a packet with this bit set that is too large in size to be transmitted further, it sends an ICMP message back to the sender indicating that the packet would have to be fragmented, and naming the MTU size of the following network segment in the sequence number field. The source host then reduces its MTU value, so that the packets can be transmitted without having to fragment them. That way, network throughput is maximised by allowing the packets to be as big as they can possibly be without causing problems.

Firewall caveats

Firewalls that drop these ICMP messages instead of forwarding them to the source host mostly render the source host unable to establish a usable connection to any host behind them, simply due to the fact that the source tries PMTUD and sets the DF bit, but then is not notified when packets get lost, so it does not reduce the packet size.

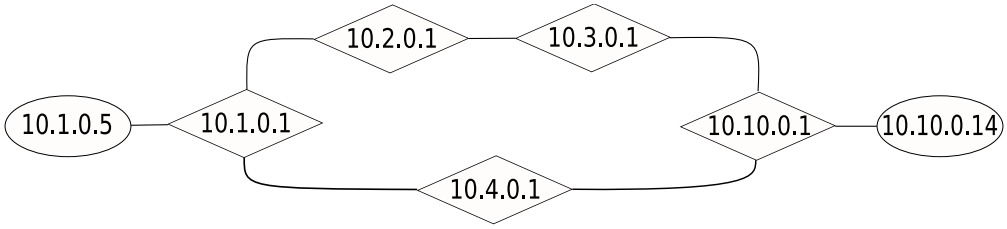

Figure 2.1: Type 5: Example network structure

2.3 Type 4: Source Quench

Source Quench messages are used to request that the sender lower their sending rate, since the router is congested and cannot forward the data at the rate it arrives, but instead drops all packets that cannot be forwarded. Each dropped packet causes one Source Quench message to be sent.

Some routers try to avoid congestion completely by monitoring network traffic and sending Source Quenches before their sending queues overflow to prevent packet loss.

Many IP implementations ignore Source Quenches completely and rely on higher level protocols, like TCP, to avoid congestion, since ICMP does not take fairness into account.

2.4 Type 5: Redirect

Imagine a network built the way shown in figure 2.1.

Assuming the host with IP 10.1.0.5 wants to send data to 10.10.0.14. There are two possible routes the packets could take, measured by hopcount the route via 10.4.0.1 is shorter than the route via 10.2.0.1 and 10.3.0.1. Maybe though, 10.1.0.1 routes all the packets via 10.2.0.1, regardless of the shorter route via 10.4.0.1.

If 10.2.0.1 is aware of the situation, it can use an ICMP redirect message to inform the sender of this inefficient route. The problem with this mechanism is, that in most scenarios, the sender is not the router that uses the inefficient route, therefore they are not in the position to do anything about it. Hence, this message type is hardly ever used in favour of dedicated routing protocols like RIP, OSPF or BGP.

2.5 Type 11: Time To Live exceeded

In a network, each router has a list of other routers that it uses to forward a packet to its destination. For each router, it is not possible to trace the route the packet was forwarded along before, so there is in fact no way of preventing packets from being routed in circles if two or more routers form a circular route. The routers simply have no way of detecting this.

To prevent packets that are being routed in circles from indefinitely ghosting around the network, IP packets are sent with a field that indicates how many routers this certain packet is allowed to pass before it either reaches its destination or is being discarded. This is called the "Time To Live" (TTL) or maximum hopcount. Each router that forwards the packet on its way decreases this value by 1.

When a router gets a packet with TTL 1 and is not its addressee, it drops the packet and sends an ICMP "TTL exceeded" message back to the sender, who either increases the TTL or complains to the network administrators about being unable to reach the destination host.

Since ICMP is not able to determine the route taken, it is only able to tell the sender that something went wrong, even though the sender might not be in the position to fix the problem. Again, communication and cooperation between all network participants is relied upon.

2.6 Type 8/0: Echo Request/Reply

As ICMP is a protocol designed to help devices and network admins with debugging their network system, it provides a simple but very powerful tool for doing so: the Echo system.

2.6.1 Functionality

Basically, the process consists of a host or router sending an ICMP Echo Request to the target host. When the target host receives the request, it sends back an ICMP Echo Reply containing the exact same ident number, sequence number and data. The identifier field is a constant number used by the ping process on the sending host to tell which messages are theirs, which becomes relevant as soon as multiple ping processes are run on the same host simultaneously. The sequence number is a running number given to each package, to be able to determine which request has been answered with a certain message.

2.6.2 What an answer tells us

If we're lucky enough to receive an Echo Reply message, we know that at least the following conditions have been met:

1. IP and routing on the source host are working.
2. There is a path from the source host to the target host.
3. IP and routing on the target host are working.
4. There is a path for the way back.

2.6.3 Use cases

An Echo Request can be used, besides testing a host for reachability, to simulate other techniques that are used on the network.

Path MTU Discovery

As stated before, the Echo Request can contain arbitrary data in its Data field which is simply copied into the Data field of the answer packet, once it reached the target host.

Since this data can be as long as needed, Echo requests can be used to perform PMTUD, as shown in Listing 2.2.

```
# Do PMTUD with 1500 bytes payload data, try to reach 85.25.9.79, try once.

$ ping -c1 -s 1500 -M do 85.25.9.79
PING 85.25.9.79 (85.25.9.79) 1500(1528) bytes of data.
From 10.5.0.197 icmp_seq=1 Frag needed and DF set (mtu = 1500)

# The packet was 1528 bytes in size, that is too big for ethernet, reduce to 1500.
# 1500 bytes of Payload + 28 bytes for ICMP and IP header = 1528 bytes of frame data.

# Do PMTUD with 1472 bytes payload data, try to reach 85.25.9.79, try once.

$ ping -c1 -s $((1500-28)) -M do 85.25.9.79
PING 85.25.9.79 (85.25.9.79) 1472(1500) bytes of data.
From 192.168.0.1 icmp_seq=1 Frag needed and DF set (mtu = 1454)

# The packet was 1500 bytes in size, that is fine for Ethernet (10.5.0.1 forwarded it),
# but too big for the next segment (DSL), so reduce to 1454.

$ ping -c1 -s $((1454-28)) -M do 85.25.9.79
PING 85.25.9.79 (85.25.9.79) 1426(1454) bytes of data.
1434 bytes from 85.25.9.79: icmp_seq=1 ttl=55 time=89.6 ms

# The target host answered, so the Path MTU is 1454.

$ ping -c1 -s $((1500-28)) -M do 85.25.9.79
PING 85.25.9.79 (85.25.9.79) 1472(1500) bytes of data.
From 10.5.0.197 icmp_seq=1 Frag needed and DF set (mtu = 1454)

# Local MTU value has been decreased to match Path MTU for that host

$ ping -c1 -s $((1500-28)) -M do 192.168.0.1
PING 192.168.0.1 (192.168.0.1) 1472(1500) bytes of data.
1480 bytes from 192.168.0.1: icmp_seq=1 ttl=63 time=2.03 ms

# PMTU for Hosts on the local network is still 1500, even though
# PMTU value for hosts on the internet is set to 1454.
```

Listing 2.2: Path MTU Discovery

Route tracing

Sometimes it may become necessary to know the route packets to a certain host are taking, e.g. to verify they are taking the correct one, or to check if one of the routers has become unreachable.

As seen in Section 2.5, routers discard packets with a TTL value of 0 and send an ICMP message reporting the problem back to the sender. This is used to trace the route to a host by starting with a TTL of one, then increasing the TTL step by step until the target machine answers, like demonstrated in listing 2.3.

```
2  # Set a TTL counter variable
3  $ ttl=1

5  # Print TTL and send an Echo Request to the target host using the current TTL
6  $ echo TTL: $ttl && ping -t $((ttl++)) -c1 85.25.9.79
7  TTL: 1
8  PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
9  From 10.5.0.1 icmp_seq=1 Time to live exceeded

11 # First gateway: 10.5.0.1

13 $ echo TTL: $ttl && ping -t $((ttl++)) -c1 85.25.9.79
14 TTL: 2
15 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
16 From 192.168.0.1 icmp_seq=1 Time to live exceeded

18 # Second gateway: 192.168.0.1

20 # the command is always the same, omitted from here.

22 TTL: 3
23 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.

25 # third gateway does not send an ICMP TTL exceeded message (though it should),
26 # so there is no telling which address it has.

28 TTL: 4
29 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
30 From 217.0.77.130 icmp_seq=1 Time to live exceeded

32 TTL: 5
33 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
34 From 217.239.40.189 icmp_seq=1 Time to live exceeded

36 TTL: 6
37 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
38 From 193.159.225.146 icmp_seq=1 Time to live exceeded

40 TTL: 7
41 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
42 # No message received

44 TTL: 8
45 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
46 From 62.75.135.29 icmp_seq=1 Time to live exceeded

48 TTL: 9
49 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
50 From 85.25.95.30 icmp_seq=1 Time to live exceeded

52 TTL: 10
53 PING 85.25.9.79 (85.25.9.79) 56(84) bytes of data.
54 64 bytes from 85.25.9.79: icmp_seq=1 ttl=55 time=50.7 ms

56 # Found Routers:
57 #  1. 10.5.0.1
58 #  2. 192.168.0.1
59 #  3. ???
60 #  4. 217.0.77.130
61 #  5. 217.239.40.189
62 #  6. 193.159.225.146
```

```
63  #  7. ???
64  #  8. 62.75.135.29
65  #  9. 85.25.95.30
66  # 10. 85.25.9.79
```

Listing 2.3: Route tracing

2.7 Type 9/10: Router Advertisement/Solicitation

This is used in conjunction with type 5 to form a basic routing protocol. Since all routing protocols support similar mechanisms, this type is deprecated and not used any more.

Router Advertisement Routers send this message periodically, naming any networks they are connected to.

Router Solicitation When a host is newly connected to the network, it may send this message to ask its neighbour routers to send an advertisement instead of waiting for the next periodical one.

2.8 Type 12: Parameter Problem

This message indicates that the IP header has been corrupted, mostly due to a transmission error, in case of which resending the packet solves the problem.

2.9 Type 13/14: Timestamp Request/Reply

This message type is in fact the predecessor of the NTP protocol. It allows for two machines to synchronise their system timers by requesting a timestamp, but it does not provide a method of transmitting time zone information. For this reason, NTP is used instead.

2.10 Type 15/16: Information Request/Reply

These messages were intended to allow for a host to find the address of the network it resides on, by simply sending a broadcast request message with an all zero recipient address. The reply then contains the address of the network the host resides on, but does not name an unused IP address for the requester to use. Therefore it has been replaced by DHCP.

2.11 Type 17/18: Address Mask Request/Reply

This message type is the same as Type 15/16, concerning the subnet mask.

2.12 Type 30: Trace Route

Whenever a router forwards or discards a packet that has the Traceroute bit set in its IP header, it sends an ICMP Traceroute message to the sender that carries information about the TTL value of the original packet, and the link speed and MTU of the current router's connection.

Bibliography

[1] Douglas E. Comer. *Internetworking with TCP/IP*, volume 1: Principles, Protocols and Architecture. Prentice-Hall, Englewood Cliffs, New Jersey, 4th edition, January 2000.

[2] J. Postel. Rfc 792: Internet control message protocol. `http://www.ietf.org/rfc/rfc792.txt`, 1981.

[3] S. Deering. Rfc 1256: Icmp router discovery messages. `http://www.ietf.org/rfc/rfc1256.txt`, 1991.