Practice Lab 4: The ICMP Protocol

1. Preliminary Work

Pre-reading: Kurose, Section 5.6: "Internet Control Message Protocol (ICMP)."

This lab session can be conducted on a **Windows or Linux operating system.**

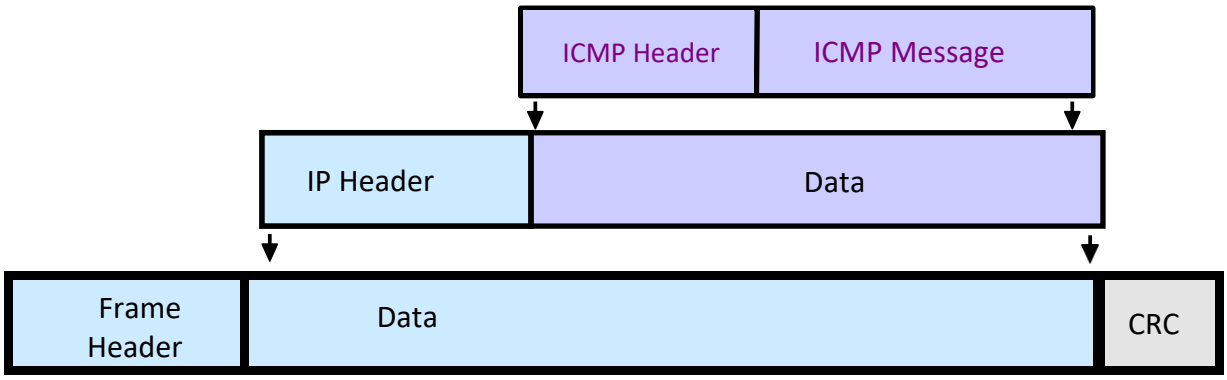
2. Introduction to ICMP

In this lab, we will study the **ICMPv4 (Internet Control Message Protocol)** and some commands derived from it.

The Internet lacks hardware mechanisms for **connectivity verification**. Additionally, the **IP protocol** does not provide **failure detection tools**. Therefore, ICMP was designed to allow **hosts and routers** to send **control messages** to other hosts and routers. It is defined in **RFC 792**.

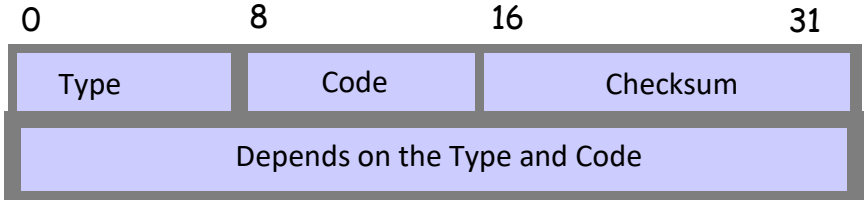
ICMP helps determine why a **datagram was not delivered** (e.g., no route available, destination unreachable, TTL expired, etc.). It only **notifies the source** of the issue without correcting it.

Since **ICMP messages** may originate **outside** the IP network where the **original datagram** was generated, they need to be carried within the **data field of an IP datagram**. However, ICMP is considered a **network layer protocol**, not a higher-layer protocol above IP.



Each ICMP message has its own format, but all start with the following fields:

- **Type (8 bits):** Identifies the **message type**.
- **Code (8 bits):** Provides **additional information** about the message type.
- **Checksum (16 bits):** Uses the **same algorithm as IP**.



The **message type** determines the **meaning and format**. ICMPv4 messages are categorized as shown in **Table 1**:

Category	Type	Message	Code	Description
Error Messages	**3**	Destination Unreachable	0	Network Unreachable
			1	Host Unreachable
			2	Protocol Unreachable
			3	Port Unreachable
			4	Fragmentation needed (DF set)
	5	Redirect	-	Routing Source Failed
	11	Time Exceeded	0	TTL expired in transit
			1	Fragment reassembly timeout
	12	Parameter Problem	-	Malformed header detected
Information Messages	**8**	Echo Request	0	Request for a response
	0	Echo Reply	-	Response to an Echo Request
	9	Router Advertisement	-	Router broadcast
	10	Router Solicitation	-	Request for router information
	13	Timestamp Request	-	Request for timestamp
	14	Timestamp Reply	-	Response with timestamp

Table 1: ICMPv4 Message Types

Error messages include the IP header and the first eight bytes of data from the original datagram. This contains the source and destination IP addresses, along with transport layer port numbers of the problematic datagram.

To prevent network issues, particularly broadcast storms, ICMP never generates error messages in response to:

- Another **ICMP error message**.
- A **datagram sent to a broadcast address**.
- Any **fragment that is not the first one**.
- A datagram whose **source address does not define a unique network connection** (e.g., 0.0.0.0, loopback, or broadcast addresses).

In this lab, we will focus on the ICMP messages detailed in the following sections.

ICMP Echo Messages

A **reply to an Echo Request** returns **the same data received** in the request. These messages are used to build the **ping tool**, which helps administrators and users **diagnose network problems**.

ICMP Echo Messages allow:

- Verifying whether a **destination is active** and if a **route exists**.
- Measuring the **round-trip time (RTT)**.
- Estimating **route reliability**.
- Used by **both hosts and routers**.

ICMP Time Exceeded Messages

These messages can be sent by **routers** or **hosts**:

- **Routers**: When a **datagram's TTL reaches zero** and it is discarded.
- **Hosts**: When a **fragmented datagram's reassembly timer expires** before all fragments arrive.

This type of message provides support for the **traceroute command**, which we will study later.

ICMP Destination Unreachable Messages

Sent by a **router or host** when it **cannot forward or deliver** an **IP datagram**. These messages are sent **to the original sender**.

ICMP Redirect Messages

Routers send **ICMP Redirect Messages** when they detect that a **better route exists** for forwarding a datagram. This helps optimize routing efficiency.

ICMP Timestamp Messages

ICMP Timestamp Request and Reply messages allow a system to determine the **latency** of an IP path by requesting the **current timestamp** from a destination.

Exercise 1

Computer **B** has received the **IP datagrams** shown in the table, which were sent by computer **A**. During the reception of these datagrams, the only **open ports on B** were **TCP ports 22 and 30,000**.

N. º	Id.	MF	OFFSET	Total Length	Protocol	Type (ICMP) / Port UDP or TCP
1	1340	1	185	1500	ICMP	8 (type 8, not port 8)
2	1341	0	0	877	UDP	8.000
3	1342	1	0	1500	TCP	22
4	1340	0	370	78	ICMP	8
5	1342	0	185	1340	TCP	22

- What data will the transport layer receive? Justify your answer.
- Will ICMP messages be generated? Justify your answer. If so, indicate which datagram(s) will generate them.

1) Packets 3 and 5 will arrive to the transport layer, but the IP layer will discard packet 2.
2) Packet 2 generates "Destination unreachable, port unreachable". Packets 1 and 4 are echo requests, so an ICMP echo response is generated for each of them.

3. IP header analysis

Exercise 2

Start the **Wireshark protocol analyser** and capture the packets generated when loading the website **www.upv.es** in a browser. Use a **capture filter** to eliminate other traffic.

Note: Application layer protocols can be filtered by specifying the **server port**: **port 80** for HTTP or **port 443** for HTTPS.

Analyse the **first four packets** and answer the following questions regarding the **IP header**:

	Id	*TTL*	*IP Source*	*IP Destination*
Packet 1	0x57d8	128	158.42.181.10	158.42.4.23
Packet 2	0x0000	60	158.42.4.23	158.42.181.10
Packet 3	0x57da	128	158.42.181.10	158.42.4.23
Packet 4	0x57dd	128	158.42.181.10	158.42.4.23

- **Regarding the TTL (Time To Live) field in the captured packets:**
  - Does it always have the same value? No, packets received have a lower TTL than those sent
  - In general, do all packets sent by a computer have the same initial TTL? Yes
  - What would be the initial TTL value in **packet 2** (the first packet sent by the server)?
- Observe how the **Identifier field** varies in both the **client and server**. Describe your observations. The client ID increases by 4 for every packet sent, while the only packet sent by the server has ID 0

Note the value of the **Protocol field**. In this case, which protocol does it refer to? TCP

4. The ping command

The **ping** command (executed from a terminal) estimates the **round-trip time (RTT)** of a packet between a **source station** and a **specified destination station**.

It achieves this by storing the **timestamp** when the packet is **sent** and subtracting it from the **current time** when the **response arrives**.

The **ping command** works using **ICMP type 0 (Echo Reply)** and **ICMP type 8 (Echo Request) messages**.

Other uses of the **ping** command:

- **Check if a destination is operational**, connected to the network, and running TCP/IP protocols.
- **Assess the reliability of the route** between source and destination (by calculating the **percentage of responses received**).

Example:

The ICMP protocol

```
user@rdc14:~$ ping www.uji.es

PING www.uji.es (84.124.83.62) 56(84) bytes of data.

64 bytes from 84.124.83.62.static.user.ono.com (84.124.83.62): icmp_seq=1 ttl=112 time=22.1 ms
64 bytes from 84.124.83.62.static.user.ono.com (84.124.83.62): icmp_seq=2 ttl=112 time=21.6 ms
64 bytes from 84.124.83.62.static.user.ono.com (84.124.83.62): icmp_seq=3 ttl=112 time=21.6 ms
64 bytes from 84.124.83.62.static.user.ono.com (84.124.83.62): icmp_seq=4 ttl=112 time=22.0 ms

     --- www.uji.es ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3004ms

rtt min/avg/max/mdev = 21.646/21.872/22.139/0.217 ms
```

The **ping command** accepts various **options**, some of the most useful ones are listed below for **Windows and Linux**. For a more detailed explanation, refer to the **manual page** of the command.

WINDOWS: ping [-n count] [-l packetsize] [-i ttl] ip destination

Options:

-n count	Number of echo requests to send.
-l size	Number of data bytes.
-i ttl	Time to Live

LINUX: ping [-b][-c count] [-s packetsize] [-t ttl] ip destination

Options:

-b	Allows broadcasting to a broadcast address
-c count	Number of echo requests to send.
-s size	Number of data bytes.
-t ttl	Time to Live

To interrupt the execution of the ping program: press Ctrl-C.

Exercise 3

Run the following **ping** commands using -n 3 (or -c 3 in Linux) to send **three requests** to each server:

- ping -n `www.uv.es` Wweb server of the University of Valencia)
- ping -n `www.uvigo.gal` (Web server of the University of Vigo),
- ping -n `www.berkeley.edu` (Web server of the University of California at Berkeley)
- ping -n `www.uq.edu.au` (Web server at the University of Queensland in Australia)

The -c 3 option configures the ping command to make only three attempts. Record the results in the following table:

	Round Trip Time (ms)		
	Min	**Avg**	**Max**
www.uv.es	1ms	1ms	2ms
www.uvigo.gal	17ms	17ms	17ms
www.berkeley.edu	52ms	52ms	52ms
www.uq.edu.au	305ms	308ms	313ms

- Analyse the variation in results across different destinations.
- Why do the results differ between destinations?
  Because if the destination is further away, the time to reach it will be longer.

The results obtained using the **ping command** can sometimes be **difficult to interpret**. The user receives **limited information** about **why** the RTT (Round Trip Time) varies between destinations.

Even when there is **no response** to a **ping request**, it is **unclear** whether:

- The **destination is offline**,
- There is **no route** from the source to the destination,
- The **network congestion** is so high that no response is received within a reasonable time.

Additionally, for **security reasons**, **network administrators** may configure **firewalls** to **filter ping messages** or **disable ICMP responses** on their computers.

Despite these limitations, **ping** remains **one of the most widely used tools** by **network administrators and users** to diagnose network connectivity.

Exercise 4

We will perform a **Wireshark capture** to analyse the **ICMP packets generated by two consecutive ping requests**.

Before starting the capture, read the instructions carefully.

- **Apply a capture filter** (**not a display filter**) to capture only the **ICMP packets** generated by executing `ping -n 3 www.uv.es`, **Linux equivalent:** ping -c 3 www.uv.es.
- Run the **ping command twice**.
- Stop the capture **after all six attempts** are completed.

Now, analyse how many **ICMP messages** were generated, paying special attention to the following **ICMP header fields**: 12

- **Type** Requests are type 8, replies are type 0
- **Code** They are all 0
- **Data bytes** 32 bytes of data

Compare the **ICMP Echo Request** and **ICMP Echo Reply** messages.

Additionally, analyse the **IP headers** of each packet, focusing on the following fields:

- **Header Length** 20 bytes
- **Total Length** 60 bytes
- **Data Bytes** 32 bytes

Compare the **Protocol field** value with what you observed in **Exercise 2**. Protocol is now ICMP?

Regarding ICMP messages:

- Why don't ICMP messages have source and destination port numbers?
- What is the purpose of the Sequence Number field in the ICMP header?

1) Because they belong to the network protocol, not to the transport protocol.
2) To pair requests with responses.

5. The tracert/traceroute Command

The **tracert** command in **Windows** or **traceroute** in **Linux** (executed from a terminal) allows us to determine the **path** (sequence of routers) that a packet follows from the **source station** to the **destination station**.

Its operation is based on **properly managing a parameter** in the **IP datagram header**—the **TTL (Time To Live) field**—and on the **ICMP messages** generated by routers when they receive a **datagram with an expired TTL**.

Each time a datagram **passes through a router**, it is considered a **hop**. We can say that **tracert/traceroute** calculates and **describes the number of hops** along a given route.

Generally, the **TTL field** is **8 bits long** and is initialized by the sender to a specific value. According to the **RFC 1700 (Assigned Numbers RFC)**, the **recommended default value is 64**. Every router that **processes the datagram** must **decrease the TTL by 1**.

If a router receives a **datagram with TTL = 1** and **decrements it**, the resulting **TTL = 0**, causing the **router to discard the datagram** and send an **ICMP Type 11 (Time Exceeded) message** back to the **originating source**.

The key to the **tracert/traceroute** command is that this **ICMP message contains the IP address** of the router that sent it.

How tracert/traceroute Works

1. The **first IP datagram** is sent to the destination with a **TTL = 1**.
    - If the **destination is not on the same network**, the **first router** encountered **decrements** the TTL to **zero**, discards the datagram, and sends a **Time Exceeded (ICMP Type 11) message** to the source.
    - This identifies the **first router** in the route.
2. A second datagram is sent with **TTL = 2**, allowing the discovery of the **second router**.
3. This process continues **incrementing the TTL** until the **destination is reached**.

When the datagram reaches the **destination**, it needs to **send a response** to stop the process. There are **two different approaches** for this:

- **Tracert (Windows):** Sends **ICMP Echo Request messages**.
    - The response upon reaching the destination is an **Echo Reply message**.
- **Traceroute (Linux/Unix):** Sends **UDP messages** to an **arbitrarily high port** (typically **33434** or similar).
    - If the **port is closed**, the destination responds with an **ICMP Port Unreachable** message.
    - If the **port is open**, no response is received, meaning the destination cannot be detected.

By default, **tracert/traceroute sends three datagrams per hop**, measuring the **round-trip time** for each. If no response is received within a **configurable time limit**, an **asterisk (*)** is displayed.

Key Considerations

- **Routes can change:** There is no guarantee that the **same path** will be used in every execution.
- **Return paths may differ:** The **return path** of a packet **is not necessarily the same** as its **forward path**.
    - For example, if a packet takes **1 second** to reach a router, but **3 seconds** to return, **tracert/traceroute** will report a **total of 4 seconds**, but it is impossible to separate **forward and return times** individually.
- **Router IP addresses:** The IP address shown in the **ICMP response** is the **incoming interface** of the router (the one that received the packet).

Exercise 5

Run the **tracert** (**traceroute** in Linux) command for the following destinations and record the **number of hops**.

	hops
www.upv.es	
www.ua.es	
www.uq.edu.au	

- Note that if the **destination is reached**, the last displayed **line** corresponds to the **final destination (a web server)** and **not to a router**.

Analyse the results:

- What could be the reason for the response when running **tracert www.ua.es**?
- In the **tracert to www.uq.edu.au**, significant differences in **link delays** are observed.
    - **What do you think causes this?**

Exercise 6

1. Open a browser and go to **www.pingtool.org**.
2. Enter a **destination IP address** (in this case, your computer's **IP address in the Network Lab**).
3. The **server at pingtool.org** will perform a **tracert from its location** to your machine.

By doing this, we will obtain **all the routers** that a datagram **passes through** when traveling from the **pingtool server** to our network.

Observation:

- The **tracert from pingtool to your machine does not reach the UPV network**, due to the **security policies** in place.

Now, perform a **tracert/traceroute** to **www.pingtool.org** from your machine to obtain the **reverse path**.
Compare both results:

- **Is the path from your machine to www.pingtool.org the same as the reverse route?**
- You may notice that some routers have **similar names** but **different IP addresses** in both cases.
  - **Why do you think this happens?**

Exercise 7

Capture the IP packets generated by running: `tracert -d www.uv.es` **Linux equivalent:** `traceroute -n www.uv.es`.

To capture the **protocol-specific packets** involved in this command, use the **capture filter**:
ip and dst host X.X.X.X where **X.X.X.X** is your computer's **IP address**.

For the packets sent by your computer:

- What type of messages does your machine send?
- What is the destination address of these messages?
  - Is it always the same?
  - Who owns this address?
- **In the IP header:**
  - How does the TTL field change?
  - What is the TTL value of the last message your machine sends?
  - How many hops did you obtain in the tracert execution?

For the response packets:

- What different types of ICMP messages does your machine receive?
- Observe the source IP addresses of the received ICMP messages.
  - Relate these addresses to the results obtained in the tracert execution.
- Explain why ICMP error packets contain information about the IP and ICMP headers.