Middle East Technical University | Department of Computer Engineering

CENG 435

Data Communications and Networking

Fall '2021-2022

Homework 3

Student Name and Surname: Berke Can Ünlü
Student Number: 2381028

1 ICMP With Minimal Topology

1.1 Answers

1.1.1 1)

The address' when **request** sent are **Source: 10.0.0.1** and **Destination: 10.0.0.2**.
The address' when **reply** sent are **Source: 10.0.0.2** and **Destination: 10.0.0.1**.

icmp

No.	Time	Source	Destination	Protocol	Length	Info
6	2.341067443	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) request
7	2.342437927	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply
9	3.341491518	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) request
10	3.341747682	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply
11	4.350395748	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) request
12	4.350459246	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply

1.1.2 2)

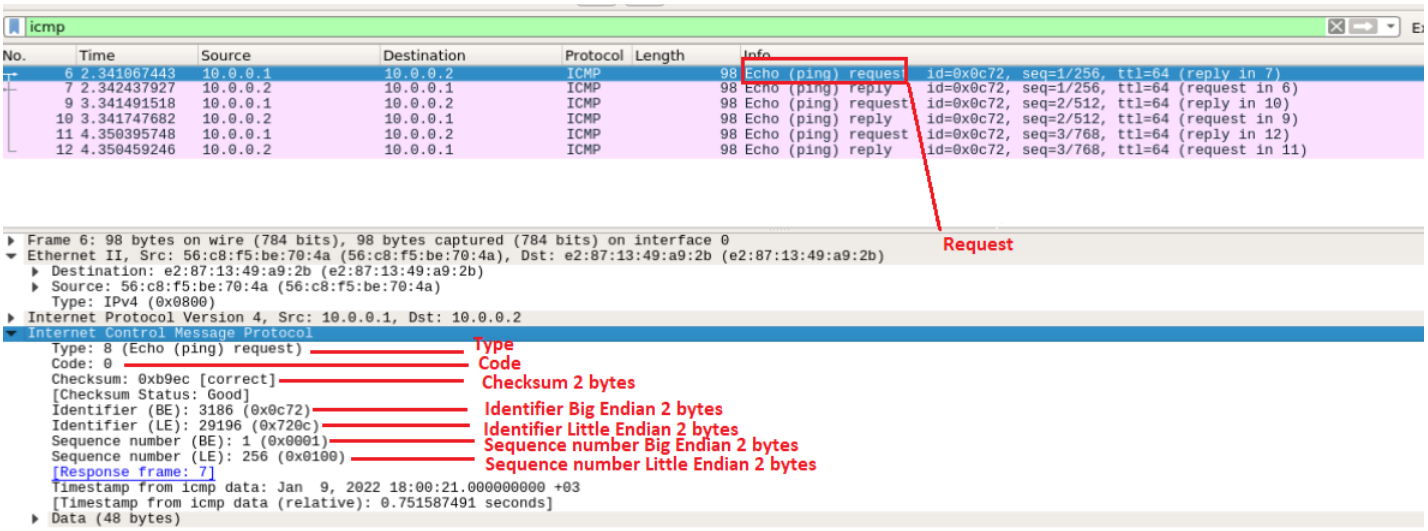
ICMP packets does not have port number since it was designed in order to communicate network-layer information between hosts and routers. It was not designed to communicate between application layer processes. It uses type and code instead of port number.[1]

1.1.3 3)

ICMP **type** is **8**. 8 stands for "**Echo**".[2]
ICMP **code** is **0**.
Also, there are **checksum**, sequence number (**BE**), sequence number (**LE**), identifier (**BE**), and identifier (**LE**).
The **checksum** has **2 bytes** since its hexadecimal representation in the examined request is **0xb9ec**.
The **identifier fields** has **2 bytes** since their hexadecimal representation in the examined request are

[1] https://www.howtouselinux.com/post/icmp-port-number
[2] https://www.ibm.com/docs/en/qsip/7.4?topic=applications-icmp-type-code-ids

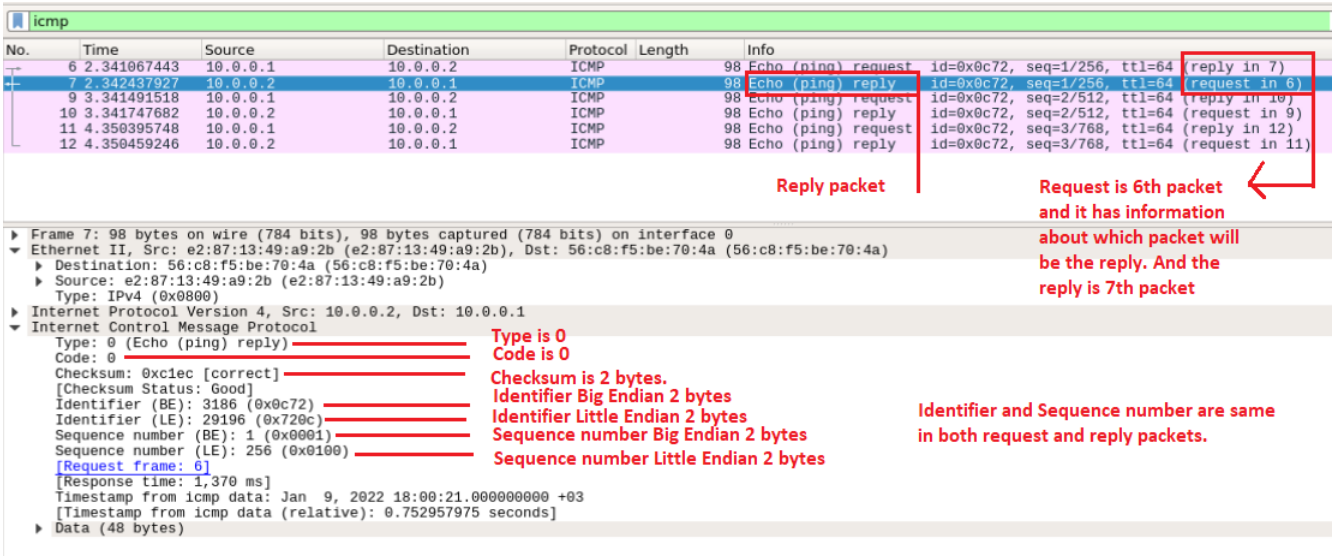
0x0c72 for (BE), for **0x720c** (LE).
The **sequence number fields** has **2 bytes** since their hexadecimal representation in the examined request are **0x0001** for (BE), for **0x0100** (LE).
Note that BE stands for Big endian, and LE stands for Little endian.



1.1.4 4)

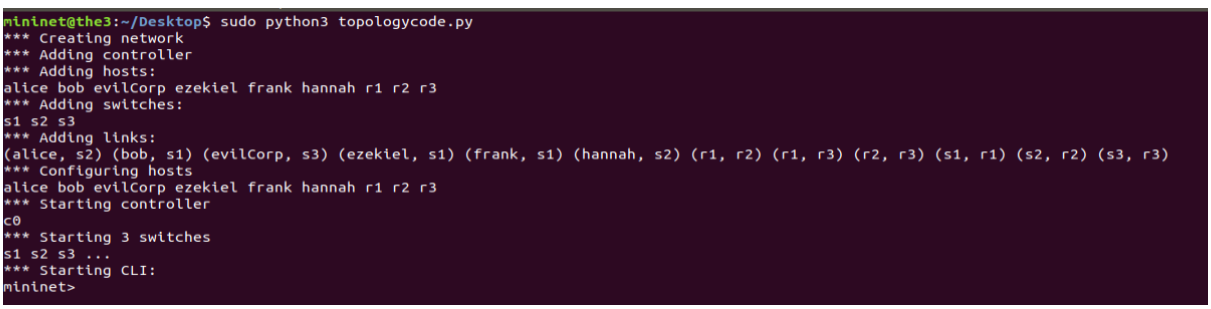
ICMP **type** is **0**. 0 stands for "**Echo reply**".[3]
ICMP **code** is **0**.
Also, there are **checksum**, sequence number (**BE**), sequence number (**LE**), identifier (**BE**), and identifier (**LE**).
The **checksum** has **2 bytes** since its hexadecimal representation in the examined request is **0xc1ec**.
The **identifier fields** has **2 bytes** since their hexadecimal representation in the examined request are **0x0c72** for (BE), for **0x720c** (LE).
The **sequence number fields** has **2 bytes** since their hexadecimal representation in the examined request are **0x0001** for (BE), for **0x0100** (LE).
Note that BE stands for Big endian, and LE stands for Little endian.

[3] https://www.ibm.com/docs/en/qsip/7.4?topic=applications-icmp-type-code-ids



2 PART2 - Creating a Topology With Mininet

2.1 Answers



```
mininet@the3:~/Desktop$ sudo python3 topologycode.py
*** Creating network
*** Adding controller
*** Adding hosts:
alice bob evilCorp ezekiel frank hannah r1 r2 r3
*** Adding switches:
s1 s2 s3
*** Adding links:
(alice, s2) (bob, s1) (evilCorp, s3) (ezekiel, s1) (frank, s1) (hannah, s2) (r1, r2) (r1, r3) (r2, r3) (s1, r1) (s2, r2) (s3, r3)
*** Configuring hosts
alice bob evilCorp ezekiel frank hannah r1 r2 r3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
```

This is the terminal screenshot after executing "**sudo python3 topologycode.py**".

2.1.1 1)

Results after executing "**pingall**".

```
mininet@the3: ~/Desktop
File  Edit  View  Search  Terminal  Help
mininet@the3:~/Desktop$ sudo python3 topologycode.py
*** Creating network
*** Adding controller
*** Adding hosts:
alice bob evilCorp ezekiel frank hannah r1 r2 r3
*** Adding switches:
s1 s2 s3
*** Adding links:
(alice, s2) (bob, s1) (evilCorp, s3) (ezekiel, s1) (frank, s1) (hannah, s2) (r1, r2) (r1, r3) (r2, r3) (s1, r1) (s2, r2) (s3, r3)
*** Configuring hosts
alice bob evilCorp ezekiel frank hannah r1 r2 r3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
alice -> bob evilCorp ezekiel frank hannah r1 r2 r3
bob -> alice evilCorp ezekiel frank hannah r1 r2 r3
evilCorp -> alice bob ezekiel frank hannah r1 r2 r3
ezekiel -> alice bob evilCorp frank hannah r1 r2 r3
frank -> alice bob evilCorp ezekiel hannah r1 r2 r3
hannah -> alice bob evilCorp ezekiel frank r1 r2 r3
r1 -> alice bob evilCorp ezekiel frank hannah r2 r3
r2 -> alice bob evilCorp ezekiel frank hannah r1 r3
r3 -> alice bob evilCorp ezekiel frank hannah r1 r2
*** Results: 0% dropped (72/72 received)
mininet>
```

2.1.2 2)

Result after executing "**ezekiel traceroute hannah**".

```
mininet> ezekiel traceroute hannah
traceroute to 10.1.0.201 (10.1.0.201), 64 hops max
  1   10.0.0.1   1,606ms  0,764ms  0,597ms
  2   10.100.0.2  1,357ms  1,398ms  1,116ms
  3   10.1.0.201  1,731ms  2,075ms  1,568ms
mininet>
```

2.1.3 3)

Result after executing "**alice traceroute bob**".

```
mininet> alice traceroute bob
traceroute to 10.0.0.251 (10.0.0.251), 64 hops max
  1   10.1.0.1   0,976ms  1,165ms  0,393ms
  2   10.100.0.1  1,640ms  1,107ms  1,773ms
  3   10.0.0.251  1,641ms  1,710ms  1,811ms
mininet>
```

2.1.4 4)

Result after executing "**frank traceroute evilCorp**".

```
mininet> frank traceroute evilCorp
traceroute to 10.0.1.101 (10.0.1.101), 64 hops max
  1   10.0.0.1  1,751ms  0,630ms  0,687ms
  2   10.200.0.2  1,008ms  1,323ms  3,921ms
  3   10.0.1.101  2,302ms  1,624ms  3,933ms
mininet>
```

2.1.5 5)

Result after executing "**evilCorp traceroute frank**".

```
mininet> evilCorp traceroute frank
traceroute to 10.0.0.250 (10.0.0.250), 64 hops max
  1   10.0.1.1  1,556ms  0,634ms  0,469ms
  2   10.200.0.1  1,208ms  0,847ms  0,887ms
  3   10.0.0.250  2,439ms  1,911ms  1,827ms
```

2.1.6 6)

Result after executing "**evilCorp traceroute alice**".

```
mininet> evilCorp traceroute alice
traceroute to 10.1.0.144 (10.1.0.144), 64 hops max
  1   10.0.1.1  1,033ms  0,545ms  0,571ms
  2   10.150.0.1  1,491ms  0,769ms  1,212ms
  3   10.1.0.144  2,779ms  1,286ms  9,947ms
```

2.1.7 7)

Result after executing "**hannah traceroute evilCorp**".

```
mininet> hannah traceroute evilCorp
traceroute to 10.0.1.101 (10.0.1.101), 64 hops max
  1   10.1.0.1  3,391ms  0,550ms  0,444ms
  2   10.150.0.2  1,016ms  0,642ms  1,337ms
  3   10.0.1.101  2,021ms  1,599ms  3,521ms
mininet>
```