

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего
профессионального образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им.
Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра «Вычислительные системы и технологии»

Отчет

по лабораторной работе №1

по дисциплине «Сети и телекоммуникации»

РУКОВОДИТЕЛЬ:

Гай В.Е.

СТУДЕНТ:

Ширшов А.А.

19-В-1

Работа защищена «___» _____

С оценкой _____

Нижний Новгород 2022

Задание на лабораторную работу

Работа с анализатором протоколов tcpdump

1. Запустить tcpdump в режиме захвата всех пакетов, проходящих по сети. Количество захватываемых пакетов ограничить 10. Результаты протоколировать в файл.
2. Запустить tcpdump в режиме перехвата широковещательного трафика (фильтр по MAC- адресу). Количество захватываемых пакетов ограничить 5. Включить распечатку пакета в шестнадцатеричной системе (включая заголовок канального уровня).
3. Запустить tcpdump так, чтобы он перехватывал только пакеты протокола ICMP, отправленные на определенный IP-адрес. При этом включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Количество захватываемых пакетов ограничить 3. Для генерирования пакетов воспользоваться утилитой ping.
4. Запустить tcpdump в режиме сохранения данных в двоичном режиме так, чтобы он перехватывал пакеты, созданные утилитой traceroute для определения маршрута к заданному в варианте узлу. Включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Количество захватываемых пакетов ограничить 7. Результат работы программы писать в файл.
5. Прочитать программой tcpdump созданный в предыдущем пункте файл.
6. Придумать три задания для фильтрации пакетов на основе протоколов ARP, TCP, UDP, ICMP.

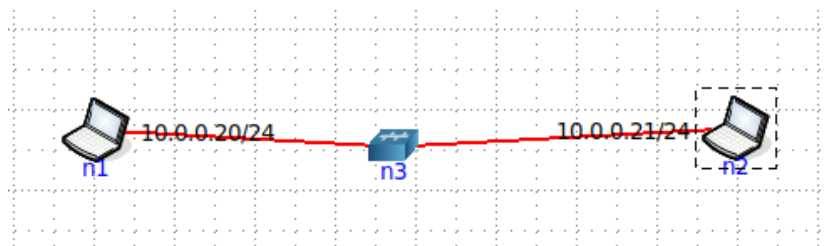
Работа с анализатором протоколов wireshark

1. Захватить 5-7 пакетов широковещательного трафика (фильтр по IP-адресу). Результат сохранить в текстовый файл.
2. Захватить 3-4 пакета ICMP, полученных от определенного узла. Для генерирования пакетов воспользоваться утилитой ping. Результат сохранить в текстовый файл.
3. Перехватить пакеты, созданные утилитой traceroute для определения маршрута к заданному в варианте узлу. По результатам построить диаграмму Flow Graph. Диаграмму сохранить либо в виде текстового файла либо в виде изображения.
4. Прочитать файл, созданный программой tcpdump. Сравнить с тем, что было получено утилитой wireshark.

Работа с анализатором протоколов tcpdump

1. Запустить tcpdump в режиме захвата всех пакетов, проходящих по сети. Количество захватываемых пакетов ограничить 10. Результаты протоколировать в файл.

Схема сети:



На n1 запустим ping

```
root@n1:/tmp/pycore.34503/n1.conf# ping 10.0.0.21
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=64 time=0.139 ms
64 bytes from 10.0.0.21: icmp_seq=2 ttl=64 time=0.105 ms
64 bytes from 10.0.0.21: icmp_seq=3 ttl=64 time=0.086 ms
64 bytes from 10.0.0.21: icmp_seq=4 ttl=64 time=0.083 ms
64 bytes from 10.0.0.21: icmp_seq=5 ttl=64 time=0.074 ms
64 bytes from 10.0.0.21: icmp_seq=6 ttl=64 time=0.112 ms
64 bytes from 10.0.0.21: icmp_seq=7 ttl=64 time=0.096 ms
^C
--- 10.0.0.21 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6147ms
rtt min/avg/max/mdev = 0.074/0.099/0.139/0.021 ms
```

На n2 запустим tcpdump

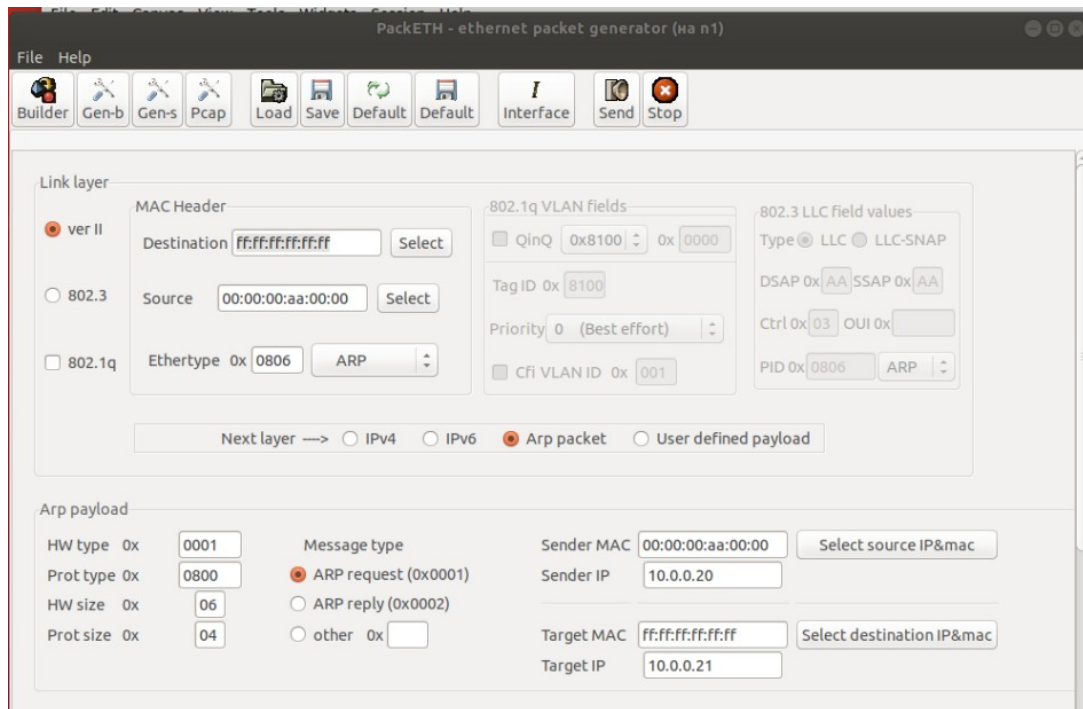
```
root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 10 -l | tee task1.log
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:31:04.400578 ARP, Request who-has n2 tell 10.0.0.20, length 28
20:31:04.400613 ARP, Reply n2 is-at 00:00:00:aa:00:01 (oui Ethernet), length 28
20:31:04.400630 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 1, length 64
20:31:04.400652 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 1, length 64
20:31:05.436071 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 2, length 64
20:31:05.436117 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 2, length 64
20:31:06.471458 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 3, length 64
20:31:06.471483 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 3, length 64
20:31:07.488663 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 4, length 64
20:31:07.488689 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 4, length 64
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

Результат записан также в файл:

```
task1.log
/tmp/pycore.34503/n2.conf
20:31:04.400578 ARP, Request who-has n2 tell 10.0.0.20, length 28
20:31:04.400613 ARP, Reply n2 is-at 00:00:00:aa:00:01 (oui Ethernet), length 28
20:31:04.400630 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 1, length 64
20:31:04.400652 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 1, length 64
20:31:05.436071 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 2, length 64
20:31:05.436117 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 2, length 64
20:31:06.471458 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 3, length 64
20:31:06.471483 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 3, length 64
20:31:07.488663 IP 10.0.0.20 > n2: ICMP echo request, id 27, seq 4, length 64
20:31:07.488689 IP n2 > 10.0.0.20: ICMP echo reply, id 27, seq 4, length 64
```

2. Запустить tcpdump в режиме перехвата широковещательного трафика (фильтр по MAC-адресу). Количество захватываемых пакетов ограничить 5. Включить распечатку пакета в шестнадцатеричной системе (включая заголовок канального уровня).

Для создания широковещательного трафика с помощью программы Packeth будем отправлять ARP-запросы со стороны n1:



На n2 запустим tcpdump

```
<onf# tcpdump -c 5 -xx 'ether dst ff:ff:ff:ff:ff:ff'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:29:42.358077 ARP, Request who-has n2 (Broadcast) tell 10.0.0.20, length 46
  0x0000:  ffff ffff ffff 0000 00aa 0000 0806 0001
  0x0010:  0800 0604 0001 0000 00aa 0000 0a00 0014
  0x0020:  ffff ffff ffff 0a00 0015 0000 0000 0000
  0x0030:  0000 0000 0000 0000 0000 0000
20:29:43.984944 ARP, Request who-has n2 (Broadcast) tell 10.0.0.20, length 46
  0x0000:  ffff ffff ffff 0000 00aa 0000 0806 0001
  0x0010:  0800 0604 0001 0000 00aa 0000 0a00 0014
  0x0020:  ffff ffff ffff 0a00 0015 0000 0000 0000
  0x0030:  0000 0000 0000 0000 0000 0000
20:29:44.439453 ARP, Request who-has n2 (Broadcast) tell 10.0.0.20, length 46
  0x0000:  ffff ffff ffff 0000 00aa 0000 0806 0001
  0x0010:  0800 0604 0001 0000 00aa 0000 0a00 0014
  0x0020:  ffff ffff ffff 0a00 0015 0000 0000 0000
  0x0030:  0000 0000 0000 0000 0000 0000
20:29:44.825145 ARP, Request who-has n2 (Broadcast) tell 10.0.0.20, length 46
  0x0000:  ffff ffff ffff 0000 00aa 0000 0806 0001
  0x0010:  0800 0604 0001 0000 00aa 0000 0a00 0014
  0x0020:  ffff ffff ffff 0a00 0015 0000 0000 0000
  0x0030:  0000 0000 0000 0000 0000 0000
20:29:45.380011 ARP, Request who-has n2 (Broadcast) tell 10.0.0.20, length 46
  0x0000:  ffff ffff ffff 0000 00aa 0000 0806 0001
  0x0010:  0800 0604 0001 0000 00aa 0000 0a00 0014
  0x0020:  ffff ffff ffff 0a00 0015 0000 0000 0000
  0x0030:  0000 0000 0000 0000 0000 0000
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

3. Запустить tcpdump так, чтобы он перехватывал только пакеты протокола ICMP, отправленные на определенный IP-адрес. При этом включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Количество захватываемых пакетов ограничить 3. Для генерирования пакетов воспользоваться утилитой ping.

Запустим ping с n1 на n2:

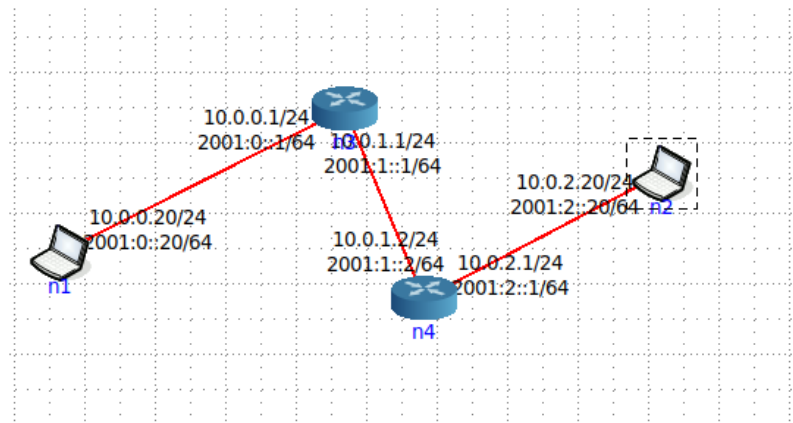
```
root@n1:/tmp/pycore.34503/n1.conf# ping 10.0.0.21
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 10.0.0.21: icmp_seq=2 ttl=64 time=0.095 ms
64 bytes from 10.0.0.21: icmp_seq=3 ttl=64 time=0.094 ms
64 bytes from 10.0.0.21: icmp_seq=4 ttl=64 time=0.049 ms
^C
--- 10.0.0.21 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.049/0.074/0.095/0.020 ms
```

Результат:

```
root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 3 -XX 'dst host 10.0.0.21 and ip proto \icmp'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:39:23.137073 IP 10.0.0.20 > n2: ICMP echo request, id 29, seq 1, length 64
    0x0000:  0000 00aa 0001 0000 00aa 0000 0800 4500  .....E.
    0x0010:  0054 3ed1 4000 4001 e7af 0a00 0014 0a00  .T>.@.@.....
    0x0020:  0015 0800 f4e9 001d 0001 cbab 1f62 0000  .....b..
    0x0030:  0000 5717 0200 0000 0000 1011 1213 1415  ..W.....
    0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  .....! "$%
    0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
    0x0060:  3637                                     67
20:39:24.149279 IP 10.0.0.20 > n2: ICMP echo request, id 29, seq 2, length 64
    0x0000:  0000 00aa 0001 0000 00aa 0000 0800 4500  .....E.
    0x0010:  0054 3f65 4000 4001 e71b 0a00 0014 0a00  .T?e@.@.....
    0x0020:  0015 0800 61b9 001d 0002 ccab 1f62 0000  ....a.....b..
    0x0030:  0000 e946 0200 0000 0000 1011 1213 1415  ...F.....
    0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  .....! "$%
    0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
    0x0060:  3637                                     67
20:39:25.178752 IP 10.0.0.20 > n2: ICMP echo request, id 29, seq 3, length 64
    0x0000:  0000 00aa 0001 0000 00aa 0000 0800 4500  .....E.
    0x0010:  0054 3fd6 4000 4001 e6aa 0a00 0014 0a00  .T?.@.@.....
    0x0020:  0015 0800 3d45 001d 0003 cdab 1f62 0000  ....=E.....b..
    0x0030:  0000 0cba 0200 0000 0000 1011 1213 1415  .....
    0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  .....! "$%
    0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
    0x0060:  3637                                     67
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

4. Запустить tcpdump в режиме сохранения данных в двоичном режиме так, чтобы он перехватывал пакеты, созданные утилитой traceroute для определения маршрута к заданному в варианте узлу. Включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Количество захватываемых пакетов ограничить 7. Результат работы программы писать в файл.

Схема сети:



Запустим утилиту traceroute с n1

```

root@n1:/tmp/pycore.34503/n1.conf# traceroute 10.0.2.20
traceroute to 10.0.2.20 (10.0.2.20), 30 hops max, 60 byte packets
 1 _gateway (10.0.0.1)  0.074 ms  0.021 ms  0.016 ms
 2 10.0.1.2 (10.0.1.2)  0.038 ms  0.023 ms  0.024 ms
 3 10.0.2.20 (10.0.2.20)  0.050 ms  0.033 ms  0.032 ms
  
```

Результат:

```

root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 7 -XX -w task4.log 'ip proto \udp or ip proto \icmp'
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
7 packets captured
16 packets received by filter
0 packets dropped by kernel
  
```



```

p2: /tmp/pyscore_38653/p2_conf# tandump -c 7 -XX -r task4.log

```

```

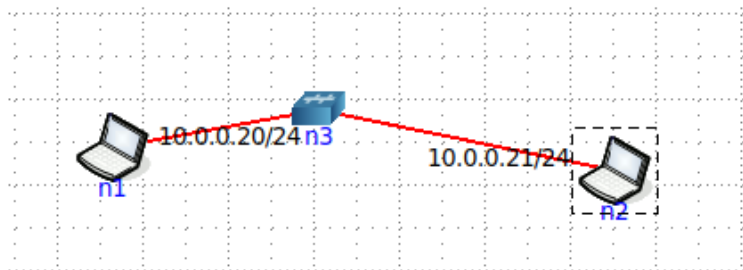
root@n2: /tmp/pycore.task8653/n2.conf# tcpdump -c 7 -XX -r task4.log
reading from file task4.log, link-type EN10MB (Ethernet)
23:13:42.400974 IP 10.0.0.20.45079 > n2.33440: UDP, length 32
    0x0000:  0000 00aa 0005 0000 00aa 0004 0800 4500  ....E.
    0x0010:  003c 3556 0000 0111 6e34 0a00 0014 0a00  .<5V....n4.....
    0x0020:  0214 b017 82a0 0028 c1b9 4041 4243 4445  ....(..@ABCDE
    0x0030:  4647 4849 4a4b 4c4d 4e4f 5051 5253 5455  FGHIJKLMNOPQRSTU
    0x0040:  5657 5859 5a5b 5c5d 5e5f                VWXYZ[\]^_
23:13:42.400995 IP n2 > 10.0.0.20: ICMP n2 udp port 33440 unreachable, length 68
    0x0000:  0000 00aa 0004 0000 00aa 0005 0800 45c0  ....E.
    0x0010:  0058 99ee 0000 4001 c9cf 0a00 0214 0a00  .X....@.....
    0x0020:  0014 0303 135e 0000 0000 4500 003c 3556  ....^....E..<5V
    0x0030:  0000 0111 6e34 0a00 0014 0a00 0214 b017  ....n4.....
    0x0040:  82a0 0028 c1b9 4041 4243 4445 4647 4849  ...(..@ABCDEFGHI
    0x0050:  4a4b 4c4d 4e4f 5051 5253 5455 5657 5859  JKLMNOPQRSTUVWXYZ
    0x0060:  5a5b 5c5d 5e5f                Z[\]^_
23:13:42.401070 IP 10.0.0.20.56098 > n2.33441: UDP, length 32
    0x0000:  0000 00aa 0005 0000 00aa 0004 0800 4500  ....E.
    0x0010:  003c 3557 0000 0111 6e33 0a00 0014 0a00  .<5W....n3.....
    0x0020:  0214 db22 82a1 0028 96ad 4041 4243 4445  ..."....(..@ABCDE
    0x0030:  4647 4849 4a4b 4c4d 4e4f 5051 5253 5455  FGHIJKLMNOPQRSTU
    0x0040:  5657 5859 5a5b 5c5d 5e5f                VWXYZ[\]^_
23:13:42.401085 IP n2 > 10.0.0.20: ICMP n2 udp port 33441 unreachable, length 68
    0x0000:  0000 00aa 0004 0000 00aa 0005 0800 45c0  ....E.
    0x0010:  0058 99ef 0000 4001 c9ce 0a00 0214 0a00  .X....@.....
    0x0020:  0014 0303 135e 0000 0000 4500 003c 3557  ....^....E..<5W
    0x0030:  0000 0111 6e33 0a00 0014 0a00 0214 db22  ....n3....."
    0x0040:  82a1 0028 96ad 4041 4243 4445 4647 4849  ...(..@ABCDEFGHI
    0x0050:  4a4b 4c4d 4e4f 5051 5253 5455 5657 5859  JKLMNOPQRSTUVWXYZ
    0x0060:  5a5b 5c5d 5e5f                Z[\]^_
23:13:42.401146 IP 10.0.0.20.46098 > n2.33442: UDP, length 32
    0x0000:  0000 00aa 0005 0000 00aa 0004 0800 4500  ....E.
    0x0010:  003c 3558 0000 0111 6e32 0a00 0014 0a00  .<5X....n2.....
    0x0020:  0214 b412 82a2 0028 bdbc 4041 4243 4445  ....(..@ABCDE
    0x0030:  4647 4849 4a4b 4c4d 4e4f 5051 5253 5455  FGHIJKLMNOPQRSTU
    0x0040:  5657 5859 5a5b 5c5d 5e5f                VWXYZ[\]^_
23:13:42.401154 IP n2 > 10.0.0.20: ICMP n2 udp port 33442 unreachable, length 68
    0x0000:  0000 00aa 0004 0000 00aa 0005 0800 45c0  ....E.
    0x0010:  0058 99f0 0000 4001 c9cd 0a00 0214 0a00  .X....@.....
    0x0020:  0014 0303 135e 0000 0000 4500 003c 3558  ....^....E..<5X
    0x0030:  0000 0111 6e32 0a00 0014 0a00 0214 b412  ....n2.....
    0x0040:  82a2 0028 bdbc 4041 4243 4445 4647 4849  ...(..@ABCDEFGHI
    0x0050:  4a4b 4c4d 4e4f 5051 5253 5455 5657 5859  JKLMNOPQRSTUVWXYZ
    0x0060:  5a5b 5c5d 5e5f                Z[\]^_

```

6. Придумать три задания для фильтрации пакетов на основе протоколов ARP, TCP, UDP, ICMP.

1) Запустить tcpdump так, чтобы он перехватывал только UDP пакеты, включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Количество захватываемых пакетов ограничить 3.

Схема сети:



Запустим чат с помощью утилиты netcat, сервер на n1 по UDP.

```
root@n1:/tmp/pycore.34503/n1.conf# nc -u -l 3000
hello
hi
how are you

```

n2 подключается к нему

```
root@n2:/tmp/pycore.34503/n2.conf# nc -u 10.0.0.20 3000
hello
hi
how are you

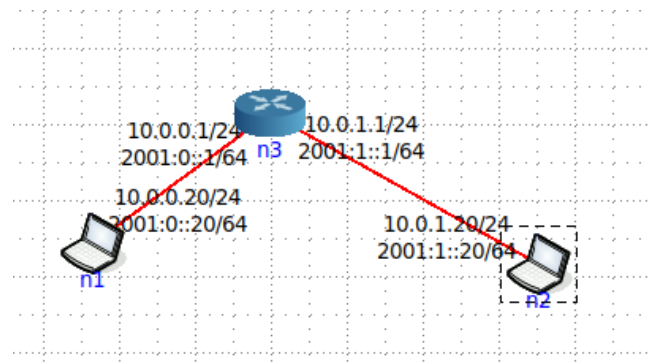
```

Результат tcpdump:

```
root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 3 -XX 'ip proto \udp'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:08:44.477125 IP n2.56605 > 10.0.0.20.3000: UDP, length 6
    0x0000:  0000 00aa 0000 0000 00aa 0001 0800 4500  .....E.
    0x0010:  0022 d48a 4000 4011 5218 0a00 0015 0a00  .."...@.R.....
    0x0020:  0014 dd1d 0bb8 000e bef7 6865 6c6c 6f0a  .....hello.
21:08:50.443569 IP 10.0.0.20.3000 > n2.56605: UDP, length 3
    0x0000:  0000 00aa 0001 0000 00aa 0000 0800 4500  .....E.
    0x0010:  001f c2c2 4000 4011 63e3 0a00 0014 0a00  ....@.c.....
    0x0020:  0015 0bb8 dd1d 000b 9070 6869 0a      .....phi.
21:08:56.166466 IP n2.56605 > 10.0.0.20.3000: UDP, length 12
    0x0000:  0000 00aa 0000 0000 00aa 0001 0800 4500  .....E.
    0x0010:  0028 dae4 4000 4011 4bb8 0a00 0015 0a00  .(..@.K.....
    0x0020:  0014 dd1d 0bb8 0014 6e2b 686f 7720 6172  .....n+how.ar
    0x0030:  6520 796f 750a      e.you.
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```


2) Запустить tcpdump так, чтобы он перехватывал только ICMP пакеты, включить распечатку пакета в ASCII-формате (включая заголовок канального уровня). Создавать пакеты с помощью traceroute. Количество захватываемых пакетов ограничить 5.

Схема сети:



Запустим traceroute на n1:

```

root@n1:/tmp/pycore.34503/n1.conf# traceroute -I 10.0.1.20
traceroute to 10.0.1.20 (10.0.1.20), 30 hops max, 60 byte packets
 1 _gateway (10.0.0.1)  0.131 ms  0.018 ms  0.015 ms
 2 10.0.1.20 (10.0.1.20)  0.101 ms  0.025 ms  0.022 ms
root@n1:/tmp/pycore.34503/n1.conf#
  
```

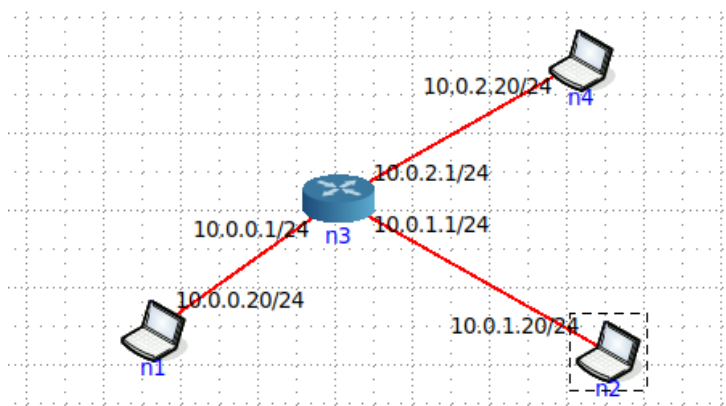
Результат tcpdump:

```

root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 5 -A 'ip proto \icmp'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:23:28.451868 IP 10.0.0.20 > n2: ICMP echo request, id 31, seq 4, length 40
E..<.....
...
.....W....HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
21:23:28.451908 IP n2 > 10.0.0.20: ICMP echo reply, id 31, seq 4, length 40
E..<.%..@..t
...
.....W....HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
21:23:28.451939 IP 10.0.0.20 > n2: ICMP echo request, id 31, seq 5, length 40
E..<.....
...
.....V....HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
21:23:28.451946 IP n2 > 10.0.0.20: ICMP echo reply, id 31, seq 5, length 40
E..<.&..@..s
...
.....V....HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
21:23:28.451967 IP 10.0.0.20 > n2: ICMP echo request, id 31, seq 6, length 40
E..<.....
...
.....U....HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
5 packets captured
26 packets received by filter
19 packets dropped by kernel
  
```

3) Запустить tcpdump так, чтобы он перехватывал только ICMP пакеты, отправленные из определенной сети. Включить распечатку пакета в шестнадцатеричной системе и ASCII-формате (включая заголовок канального уровня). Создавать пакеты с помощью ping. Количество захватываемых пакетов ограничить 2.

Добавим в схему еще одну подсеть:



Запускаем ping с n1 на n2

```

root@n1:/tmp/pycore.34503/n1.conf# ping 10.0.1.20
PING 10.0.1.20 (10.0.1.20) 56(84) bytes of data.
64 bytes from 10.0.1.20: icmp_seq=1 ttl=63 time=0.146 ms
64 bytes from 10.0.1.20: icmp_seq=2 ttl=63 time=0.129 ms
64 bytes from 10.0.1.20: icmp_seq=3 ttl=63 time=0.139 ms
^C
--- 10.0.1.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
rtt min/avg/max/mdev = 0.129/0.138/0.146/0.007 ms
root@n1:/tmp/pycore.34503/n1.conf#

```

Результат tcpdump:

```

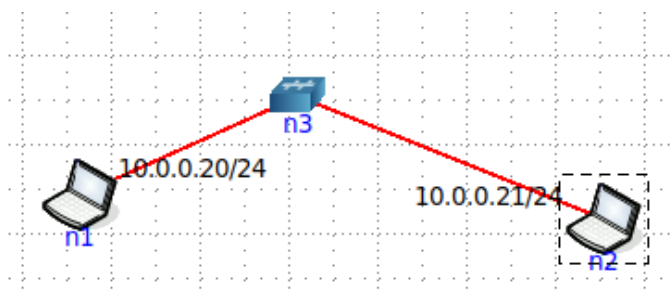
root@n2:/tmp/pycore.34503/n2.conf# tcpdump -c 2 -XX 'src net 10.0.0.0/24 and ip proto \icmp'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:33:09.770790 IP 10.0.0.20 > n2: ICMP echo request, id 33, seq 1, length 64
    0x0000:  0000 00aa 0003 0000 00aa 0002 0800 4500  .....E.
    0x0010:  0054 2259 4000 3f01 0429 0a00 0014 0a00  .T"Y@.?.).....
    0x0020:  0114 0800 0f2e 0021 0001 65b8 1f62 0000  ....(!..f..b..
    0x0030:  0000 99c2 0b00 0000 0000 1011 1213 1415  .....
    0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  .....!""#$%
    0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
    0x0060:  3637                                     67
21:33:10.771860 IP 10.0.0.20 > n2: ICMP echo request, id 33, seq 2, length 64
    0x0000:  0000 00aa 0003 0000 00aa 0002 0800 4500  .....E.
    0x0010:  0054 2275 4000 3f01 040d 0a00 0014 0a00  .T"u@.?.).....
    0x0020:  0114 0800 e028 0021 0002 66b8 1f62 0000  ....(!..f..b..
    0x0030:  0000 c7c6 0b00 0000 0000 1011 1213 1415  .....
    0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  .....!""#$%
    0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
    0x0060:  3637                                     67
2 packets captured
2 packets received by filter
0 packets dropped by kernel

```

Работа с анализатором протоколов wireshark

1. Захватить 5-7 пакетов широковещательного трафика (фильтр по IP-адресу).
Результат сохранить в текстовый файл.

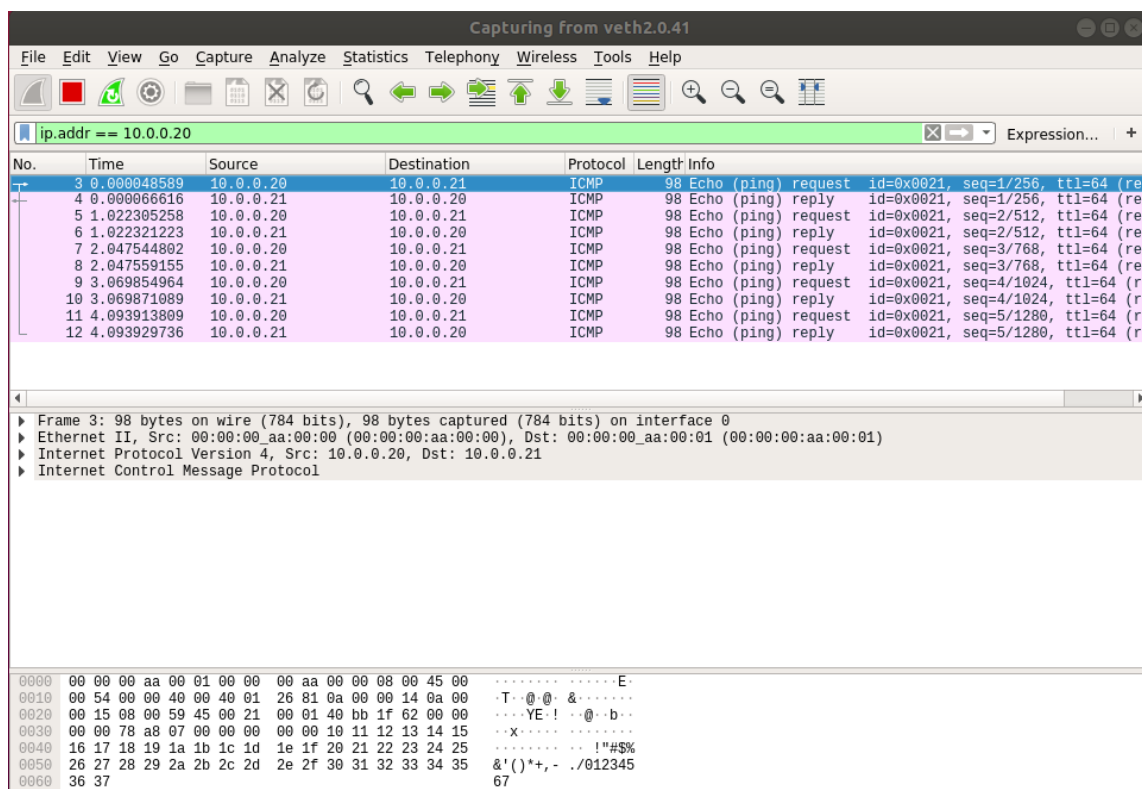
Схема сети:



Запускаем ping:

```
root@n1:/tmp/pycore.34503/n1.conf# ping -b 10.0.0.21
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=64 time=0.132 ms
64 bytes from 10.0.0.21: icmp_seq=2 ttl=64 time=0.049 ms
64 bytes from 10.0.0.21: icmp_seq=3 ttl=64 time=0.047 ms
64 bytes from 10.0.0.21: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 10.0.0.21: icmp_seq=5 ttl=64 time=0.047 ms
^C
--- 10.0.0.21 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4093ms
rtt min/avg/max/mdev = 0.047/0.065/0.132/0.033 ms
```

Wireshark на n2, фильтр по IP:



Результат в файле:

```
task1.txt
+-----+
18:45:20,501,944  ETHER
|0  |ff|ff|ff|ff|ff|ff|00|00|00|aa|00|00|08|06|00|01|08|00|06|04|00|01|00|
00|00|aa|00|00|0a|00|00|14|00|00|00|00|00|00|0a|00|00|15|

+-----+
18:45:20,501,985  ETHER
|0  |00|00|00|aa|00|00|00|00|00|aa|00|01|08|06|00|01|08|00|06|04|00|02|00|
00|00|aa|00|01|0a|00|00|15|00|00|00|aa|00|00|0a|00|00|14|

+-----+
18:45:20,501,993  ETHER
|0  |00|00|00|aa|00|01|00|00|00|aa|00|00|08|00|45|00|00|54|00|00|40|00|40|
01|26|81|0a|00|00|14|0a|00|00|15|08|00|59|45|00|21|00|01|40|bb|1f|62|00|00|
00|00|78|a8|07|00|00|00|00|00|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|

+-----+
18:45:20,502,011  ETHER
|0  |00|00|00|aa|00|00|00|00|00|aa|00|01|08|00|45|00|00|54|74|db|00|00|40|
01|f1|a5|0a|00|00|15|0a|00|00|14|00|00|61|45|00|21|00|01|40|bb|1f|62|00|00|
00|00|78|a8|07|00|00|00|00|00|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|

+-----+
18:45:21,524,250  ETHER
|0  |00|00|00|aa|00|01|00|00|00|aa|00|00|08|00|45|00|00|54|00|00|40|00|40|
01|26|81|0a|00|00|14|0a|00|00|15|08|00|14|ed|00|21|00|02|41|bb|1f|62|00|00|
00|00|bb|ff|07|00|00|00|00|00|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|

+-----+
18:45:21,524,266  ETHER
|0  |00|00|00|aa|00|00|00|00|00|aa|00|01|08|00|45|00|00|54|75|87|00|00|40|
01|f0|f9|0a|00|00|15|0a|00|00|14|00|00|1c|ed|00|21|00|02|41|bb|1f|62|00|00|
00|00|bb|ff|07|00|00|00|00|00|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|
```

2. Захватить 3-4 пакета ICMP, полученных от определенного узла. Для генерирования пакетов воспользоваться утилитой ping. Результат сохранить в текстовый файл.

Запустим ping:

```
root@n1:/tmp/pycore.34503/n1.conf# ping 10.0.0.21
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=64 time=0.056 ms
64 bytes from 10.0.0.21: icmp_seq=2 ttl=64 time=0.049 ms
^C
--- 10.0.0.21 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1014ms
rtt min/avg/max/mdev = 0.049/0.052/0.056/0.008 ms
```

Фильтр по icmp

veth2.0.41

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|-----------|-------------|----------|--------|--|
| 1 | 0.000000000 | 10.0.0.20 | 10.0.0.21 | ICMP | 98 | Echo (ping) request id=0x0022, seq=1/256, ttl=64 (req) |
| 2 | 0.000019459 | 10.0.0.21 | 10.0.0.20 | ICMP | 98 | Echo (ping) reply id=0x0022, seq=1/256, ttl=64 (req) |
| 3 | 1.014888036 | 10.0.0.20 | 10.0.0.21 | ICMP | 98 | Echo (ping) request id=0x0022, seq=2/512, ttl=64 (req) |
| 4 | 1.014902746 | 10.0.0.21 | 10.0.0.20 | ICMP | 98 | Echo (ping) reply id=0x0022, seq=2/512, ttl=64 (req) |

Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
 Ethernet II, Src: 00:00:00:aa:00:00 (00:00:00:aa:00:00), Dst: 00:00:00:aa:00:01 (00:00:00:aa:00:01)
 Internet Protocol Version 4, Src: 10.0.0.20, Dst: 10.0.0.21
 Internet Control Message Protocol

```

0000  00 00 00 aa 00 01 00 00 00 aa 00 00 08 00 45 00  .....E
0010  00 54 9f b7 40 00 40 01 86 c9 0a 00 00 14 0a 00  .T..@.
0020  00 15 08 00 18 6c 00 22 00 01 d6 bc 1f 62 00 00  ....l..b..
0030  00 00 22 7f 08 00 00 00 00 00 10 11 12 13 14 15  ....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  .... !"#%$
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37                                     67
  
```

task2.txt

```

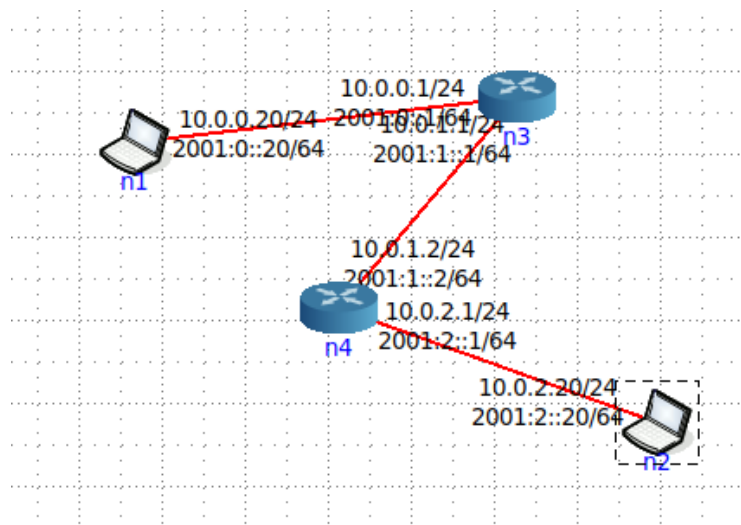
+-----+
18:52:06,556,867  ETHER
|0|00|00|00|aa|00|01|00|00|00|aa|00|00|08|00|45|00|00|54|9f|b7|40|00|40|
01|86|c9|0a|00|00|14|0a|00|00|15|08|00|18|6c|00|22|00|01|d6|bc|1f|62|00|00|
00|00|22|7f|08|00|00|00|00|00|10|11|12|13|14|15|16|17|18|19|1a|1b|1c|1d|1e|
1f|20|21|22|23|24|25|26|27|28|29|2a|2b|2c|2d|2e|2f|30|31|32|33|34|35|36|37|

+-----+
18:52:06,556,886  ETHER
|0|00|00|00|aa|00|00|00|00|00|aa|00|01|08|00|45|00|00|54|34|db|00|00|40|
01|31|a6|0a|00|00|15|0a|00|00|14|00|00|20|6c|00|22|00|01|d6|bc|1f|62|00|00|
00|00|22|7f|08|00|00|00|00|00|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|

+-----+
18:52:07,571,755  ETHER
|0|00|00|00|aa|00|01|00|00|00|aa|00|00|08|00|45|00|00|54|a0|90|40|00|40|
01|85|f0|0a|00|00|14|0a|00|00|15|08|00|ee|30|00|22|00|02|d7|bc|1f|62|00|00|
00|00|4b|b9|08|00|00|00|00|00|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|

+-----+
18:52:07,571,769  ETHER
|0|00|00|00|aa|00|00|00|00|00|aa|00|01|08|00|45|00|00|54|35|c0|00|00|40|
01|30|c1|0a|00|00|15|0a|00|00|14|00|00|f6|30|00|22|00|02|d7|bc|1f|62|00|00|
00|00|4b|b9|08|00|00|00|00|00|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|
  
```


Схема сети:



```
root@n1:/tmp/pycore.34503/n1.conf# traceroute 10.0.2.20
traceroute to 10.0.2.20 (10.0.2.20), 30 hops max, 60 byte packets
 1  _gateway (10.0.0.1)  0.068 ms  0.005 ms  0.004 ms
 2  10.0.1.2 (10.0.1.2)  0.046 ms  0.007 ms  0.006 ms
 3  10.0.2.20 (10.0.2.20)  0.032 ms  0.009 ms  0.009 ms
```

The screenshot displays the Wireshark network protocol analyzer interface. The title bar indicates the capture source is 'veth2.0.41'. The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains icons for various functions like opening files, saving, and zooming.

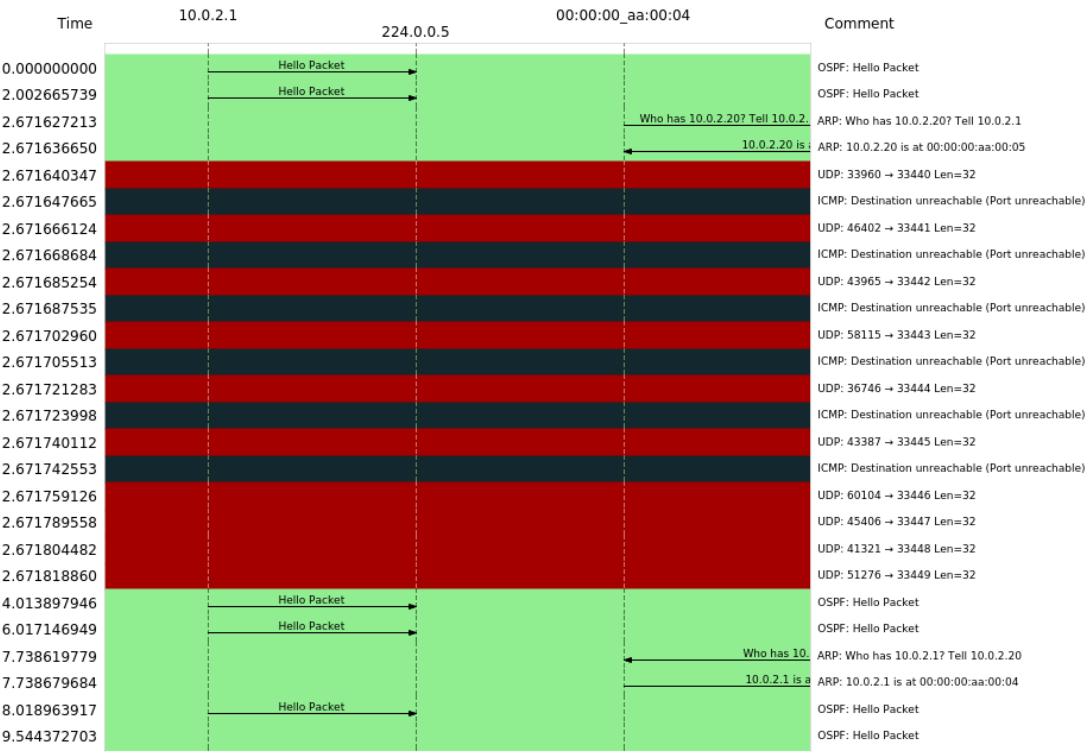
The packet list pane shows a series of 20 packets. The first packet is a UDP packet from 10.0.0.20 to 10.0.0.20, port 33960 to 33440, length 74. The subsequent packets are ICMP 'Destination unreachable' messages, indicating that the destination port is unreachable. The packets alternate between UDP and ICMP.

The packet details pane for the selected packet (Frame 5) shows the following structure:

- Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
- Ethernet II, Src: 00:00:00:aa:00:04 (00:00:00:aa:00:04), Dst: 00:00:00:aa:00:05 (00:00:00:aa:00:05)
- Internet Protocol Version 4, Src: 10.0.0.20, Dst: 10.0.2.20
- User Datagram Protocol, Src Port: 33960, Dst Port: 33440
- Data (32 bytes)

The packet bytes pane shows the raw data in hexadecimal and ASCII. The data is a 32-byte string: '.....E.....<.....(.....@ABCDEFGHIJKLM NOPQRSTU VWXYZ[\]^_'. The ASCII column shows the corresponding characters, with some non-printable characters represented by dots.

Flow graph:

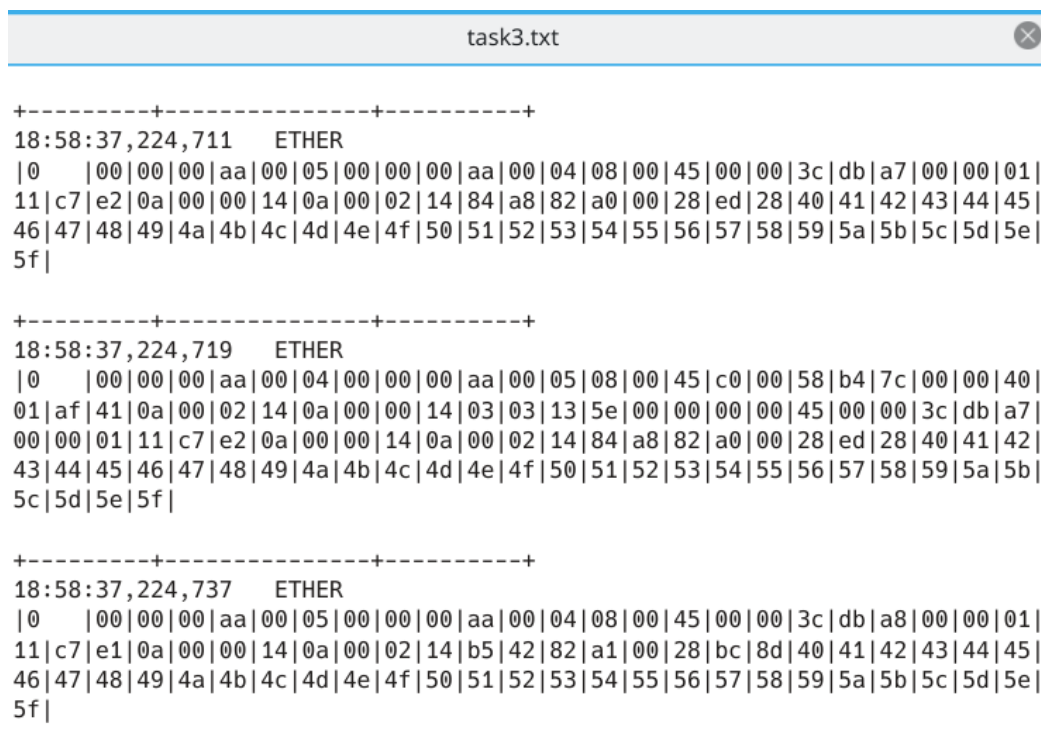


4. Прочитать файл, созданный программой tcpdump. Сравнить с тем, что было получено утилитой Wireshark.

Файл, созданный программой tcpdump:

```
root@n2:/tmp/pycore.38653/n2.conf# tcpdump -c 7 -XX -r task4.log
reading from file task4.log, link-type EN10MB (Ethernet)
23:13:42.400974 IP 10.0.0.20.45079 > n2.33440: UDP, length 32
    0x0000: 0000 00aa 0005 0000 00aa 0004 0800 4500 .....E.
    0x0010: 003c 3556 0000 0111 6e34 0a00 0014 0a00 .<5V....n4.....
    0x0020: 0214 b017 82a0 0028 c1b9 4041 4243 4445 .....(..@ABCDE
    0x0030: 4647 4849 4a4b 4c4d 4e4f 5051 5253 5455 FGHIJKLMNOPQRSTU
    0x0040: 5657 5859 5a5b 5c5d 5e5f VWXYZ[\]^_
23:13:42.400995 IP n2 > 10.0.0.20: ICMP n2 udp port 33440 unreachable, length 68
    0x0000: 0000 00aa 0004 0000 00aa 0005 0800 45c0 .....E.
    0x0010: 0058 99ee 0000 4001 c9cf 0a00 0214 0a00 .X....@.....
    0x0020: 0014 0303 135e 0000 0000 4500 003c 3556 .....^....E..<5V
    0x0030: 0000 0111 6e34 0a00 0014 0a00 0214 b017 ....n4.....
    0x0040: 82a0 0028 c1b9 4041 4243 4445 4647 4849 ...(..@ABCDEFGHI
    0x0050: 4a4b 4c4d 4e4f 5051 5253 5455 5657 5859 JKLMNOPQRSTUVWXYZ
    0x0060: 5a5b 5c5d 5e5f Z[\]^_
23:13:42.401070 IP 10.0.0.20.56098 > n2.33441: UDP, length 32
    0x0000: 0000 00aa 0005 0000 00aa 0004 0800 4500 .....E.
    0x0010: 003c 3557 0000 0111 6e33 0a00 0014 0a00 .<5W....n3.....
    0x0020: 0214 db22 82a1 0028 96ad 4041 4243 4445 ..."....(..@ABCDE
    0x0030: 4647 4849 4a4b 4c4d 4e4f 5051 5253 5455 FGHIJKLMNOPQRSTU
    0x0040: 5657 5859 5a5b 5c5d 5e5f VWXYZ[\]^_
23:13:42.401085 IP n2 > 10.0.0.20: ICMP n2 udp port 33441 unreachable, length 68
```

Результат в Wireshark



```
task3.txt
+-----+-----+-----+
18:58:37,224,711  ETHER
|0|00|00|00|aa|00|05|00|00|00|aa|00|04|08|00|45|00|00|3c|db|a7|00|00|01|
11|c7|e2|0a|00|00|14|0a|00|02|14|84|a8|82|a0|00|28|ed|28|40|41|42|43|44|45|
46|47|48|49|4a|4b|4c|4d|4e|4f|50|51|52|53|54|55|56|57|58|59|5a|5b|5c|5d|5e|
5f|

+-----+-----+-----+
18:58:37,224,719  ETHER
|0|00|00|00|aa|00|04|00|00|00|aa|00|05|08|00|45|c0|00|58|b4|7c|00|00|40|
01|af|41|0a|00|02|14|0a|00|00|14|03|03|13|5e|00|00|00|00|45|00|00|3c|db|a7|
00|00|01|11|c7|e2|0a|00|00|14|0a|00|02|14|84|a8|82|a0|00|28|ed|28|40|41|42|
43|44|45|46|47|48|49|4a|4b|4c|4d|4e|4f|50|51|52|53|54|55|56|57|58|59|5a|5b|
5c|5d|5e|5f|

+-----+-----+-----+
18:58:37,224,737  ETHER
|0|00|00|00|aa|00|05|00|00|00|aa|00|04|08|00|45|00|00|3c|db|a8|00|00|01|
11|c7|e1|0a|00|00|14|0a|00|02|14|b5|42|82|a1|00|28|bc|8d|40|41|42|43|44|45|
46|47|48|49|4a|4b|4c|4d|4e|4f|50|51|52|53|54|55|56|57|58|59|5a|5b|5c|5d|5e|
5f|
```

Данные в целом схожи. Однако распечатка tcpdump дает больше информации о протоколе, направлении, длине, кроме того, видно эти же данные в ASCII формате.