

МОЛДАВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ МАТЕМАТИКИ И ИНФОРМАТИКИ
ДЕПАРТАМЕНТ ИНФОРМАТИКИ

Лабораторная работа № 2
по курсу
Sisteme de monitorizare și analiză a infrastructurii IT și a aplicațiilor
Тема:
“Основы сетей”

Выполнил: Bogdanov Iurii,
студент группы I2302
Проверил: D. Borș

Кишинёв, 2025

Цель лабораторной работы:

Освоить базовые сетевые команды и инструменты диагностики в Linux. Научиться анализировать сетевые подключения и маршруты, а также проверять доступность удалённых ресурсов.

Ход работы

Теоретическое введение:

- IP-адрес и маска сети.
- MAC-адрес.
- Маршрутизация: таблица маршрутов.
- DNS: преобразование имён в IP-адреса.
- Утилиты: ip, ping, traceroute, ss, netstat, dig, curl, nc.

Подготовка окружения

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ sudo apt update
```

Обновление списка пакетов в системе с помощью команды **sudo apt update**. Эта операция синхронизирует локальную базу пакетов с репозиториями Ubuntu, чтобы были доступны последние версии утилит.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ sudo apt install -y iproute2 net-tools dnsutils traceroute curl netcat-openbsd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
iproute2 is already the newest version (6.1.0-1ubuntu6.2).
iproute2 set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.6).
curl set to manually installed.
netcat-openbsd is already the newest version (1.226-1ubuntu2).
netcat-openbsd set to manually installed.
The following additional packages will be installed:
  bind9-dnsutils bind9-host bind9-libs liblmbd0 libmaxminddb0 libuv1t64
Suggested packages:
  mmdns-bin
The following NEW packages will be installed:
  bind9-dnsutils bind9-host bind9-libs dnsutils liblmbd0 libmaxminddb0 libuv1t64 net-tools traceroute
0 upgraded, 9 newly installed, 0 to remove and 32 not upgraded.
1 not fully installed or removed.
Need to get 1901 kB of archives.
After this operation, 5669 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu/noble/main amd64 libuv1t64 amd64 1.48.0-1.1build1 [97.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu/noble/main amd64 liblmbd0 amd64 0.9.31-1build1 [48.1 kB]
Get:3 http://archive.ubuntu.com/ubuntu/noble/main amd64 libmaxminddb0 amd64 1.9.1-1build1 [24.4 kB]
Get:4 http://archive.ubuntu.com/ubuntu/noble-updates/main amd64 bind9-libs amd64 1:9.18.39-0ubuntu0.24.04.1 [1257 kB]
Get:5 http://archive.ubuntu.com/ubuntu/noble-updates/main amd64 bind9-host amd64 1:9.18.39-0ubuntu0.24.04.1 [50.5 kB]
Get:6 http://archive.ubuntu.com/ubuntu/noble-updates/main amd64 bind9-dnsutils amd64 1:9.18.39-0ubuntu0.24.04.1 [156 kB]
Get:7 http://archive.ubuntu.com/ubuntu/noble-updates/universe amd64 dnsutils all 1:9.18.39-0ubuntu0.24.04.1 [3678 B]
Get:8 http://archive.ubuntu.com/ubuntu/noble-updates/main amd64 net-tools amd64 2.10-0.1ubuntu4.4 [204 kB]
Get:9 http://archive.ubuntu.com/ubuntu/noble/universe amd64 traceroute amd64 1:2.1.5-1 [60.5 kB]
Fetched 1901 kB in 2s (975 kB/s)
Selecting previously unselected package libuv1t64:amd64.
(Reading database ... 41909 files and directories currently installed.)
Preparing to unpack .../0-libuv1t64_1.48.0-1.1build1_amd64.deb ...
Unpacking libuv1t64:amd64 (1.48.0-1.1build1) ...
Selecting previously unselected package liblmbd0:amd64.
Preparing to unpack .../1-liblmbd0_0.9.31-1build1_amd64.deb ...
Unpacking liblmbd0:amd64 (0.9.31-1build1) ...
Selecting previously unselected package libmaxminddb0:amd64.
Preparing to unpack .../2-libmaxminddb0_1.9.1-1build1_amd64.deb ...
Unpacking libmaxminddb0:amd64 (1.9.1-1build1) ...
Selecting previously unselected package bind9-libs:amd64.
Preparing to unpack .../3-bind9-libs_1:9.18.39-0ubuntu0.24.04.1_amd64.deb ...
Unpacking bind9-libs:amd64 (1:9.18.39-0ubuntu0.24.04.1) ...
Selecting previously unselected package bind9-host.
Preparing to unpack .../4-bind9-host_1:9.18.39-0ubuntu0.24.04.1_amd64.deb ...
Unpacking bind9-host (1:9.18.39-0ubuntu0.24.04.1) ...
Selecting previously unselected package bind9-dnsutils.
Preparing to unpack .../5-bind9-dnsutils_1:9.18.39-0ubuntu0.24.04.1_amd64.deb ...
Unpacking bind9-dnsutils (1:9.18.39-0ubuntu0.24.04.1) ...
```

Установка необходимых сетевых инструментов (**iproute2**, **net-tools**, **dnsutils**, **traceroute**, **curl**, **netcat-openbsd**) для выполнения лабораторной работы. Эти пакеты обеспечивают доступ к командам: **ip**, **ss**, **netstat**, **dig**, **traceroute**, **curl**, **nc**, которые будут использоваться для диагностики и анализа сети.

Часть 1: Базовая диагностика

1. Определим IP-адреса и MAC-адреса всех сетевых интерфейсов нашей машины.

- Определение IP-адресов интерфейсов с помощью команды **ip -br address**

Утилита **ip** используется для работы с сетевыми настройками. Ключ **-br** означает *brief* (сжатый, краткий вывод), а подкоманда **address** показывает IP-адреса всех интерфейсов. Благодаря этому можно быстро узнать, какой IP назначен машине, а также увидеть служебные адреса.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ ip -br address
lo                UNKNOWN      127.0.0.1/8 10.255.255.254/32 ::1/128
eth0              UP           172.26.20.5/20 fe80::215:5dff:fe7a:f1a5/64
```

Эта команда выводит список сетевых интерфейсов и их IP-адреса. В результате видно, что интерфейс **lo** имеет адреса 127.0.0.1 и ::1, а рабочий интерфейс **eth0** получил адрес 172.26.20.5/20.

- Определение MAC-адресов интерфейсов с помощью команды **ip -br link**

Снова используется утилита **ip** с ключом **-br** для компактного вывода, а подкоманда **link** выводит состояние интерфейсов и их физические адреса (MAC). Таким образом можно определить уникальные идентификаторы сетевых карт.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ ip -br link
lo                UNKNOWN      00:00:00:00:00:00 <LOOPBACK,UP,LOWER_UP>
eth0              UP           00:15:5d:7a:f1:a5 <BROADCAST,MULTICAST,UP,LOWER_UP>
```

Команда показывает состояние сетевых интерфейсов и их MAC-адреса. Интерфейс **lo** не имеет физического адреса, так как он виртуальный, а интерфейс **eth0** имеет MAC-адрес 00:15:5d:7a:f1:a5.

2. Выведем таблицу маршрутизации с помощью команды *ip route*

Подкоманда **route** указывает утилите **ip** вывести таблицу маршрутизации. Она показывает, какие сети доступны напрямую и через какой шлюз отправляются пакеты в интернет. В моём случае пакеты в интернет идут через шлюз 172.26.16.1 по интерфейсу eth0, а сеть 172.26.16.0/20 доступна напрямую.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ ip route
default via 172.26.16.1 dev eth0 proto kernel
172.26.16.0/20 dev eth0 proto kernel scope link src 172.26.20.5
```

3. Проверим доступность узла 8.8.8.8 и сайта google.com с помощью **ping**.

- Проверка доступности узла по IP с помощью команды *ping -c 4 8.8.8.8*

Команда проверяет, доступен ли узел по указанному IP-адресу, отправляя ему пакеты. Это помогает понять, есть ли связь с внешней сетью. В моём случае сервер Google DNS (8.8.8.8) ответил на все пакеты без потерь, средняя задержка составила около 81 мс. Ключ **-c 4** указывает количество пакетов (четыре).

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ ping -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=112 time=60.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=112 time=58.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=112 time=109 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=112 time=81.0 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 57.962/76.933/108.782/20.484 ms
```

- Проверка доступности сайта по имени с помощью команды *ping -c 4 google.com*

Команда проверяет доступность ресурса по доменному имени. Для работы сначала используется DNS, который переводит имя сайта в IP-адрес. В моём случае google.com был преобразован в 142.251.208.142, сайт ответил на все пакеты, задержка также около 81 мс.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ ping -c 4 google.com
PING google.com (142.251.208.142) 56(84) bytes of data.
64 bytes from bud02s42-in-f14.1e100.net (142.251.208.142): icmp_seq=1 ttl=112 time=44.2 ms
64 bytes from bud02s42-in-f14.1e100.net (142.251.208.142): icmp_seq=2 ttl=112 time=56.5 ms
64 bytes from bud02s42-in-f14.1e100.net (142.251.208.142): icmp_seq=3 ttl=112 time=81.7 ms
64 bytes from bud02s42-in-f14.1e100.net (142.251.208.142): icmp_seq=4 ttl=112 time=65.9 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 44.178/62.082/81.749/13.719 ms
```

4. Сравните результаты: что произойдёт, если DNS не работает?

Сравнение результатов

- При пинге по IP (8.8.8.8) запрос идёт напрямую без участия DNS.
- При пинге по доменному имени (google.com) сначала происходит обращение к DNS-серверу для преобразования имени в IP.
- Если DNS не работает, пинг по IP всё равно будет успешным, а по имени — завершится ошибкой «Temporary failure in name resolution».

Часть 2: Маршруты и трассировка

1. Выполним трассировку (**tracert**) до google.com.

Утилита **tracert** показывает, по каким промежуточным маршрутизаторам проходят пакеты, пока не достигнут конечного адресата. Каждый «hop» в выводе соответствует сетевому устройству, через которое проходят данные. В моём случае видно 17 шагов от локального компьютера до IP-адреса google.com (142.251.208.142). В выводе можно заметить, что первые узлы относятся к локальной сети и провайдеру, затем пакеты проходят через несколько магистральных маршрутизаторов, а в конце достигают серверов Google.

```
bogdanov18@DESKTOP-R3SPK2V: /mnt/c/Users/Asus$ tracert google.com
tracert to google.com [142.251.208.142], 30 hops max, 60 byte packets
 1  DESKTOP-R3SPK2V.mshome.net (172.26.16.1)  0.373 ms  0.354 ms  0.284 ms
 2  www.huaweimobilewifi.com (192.168.8.1)  41.102 ms  41.044 ms  41.996 ms
 3  10.126.13.243 (10.126.13.243)  81.486 ms  81.850 ms  81.837 ms
 4  10.250.8.110 (10.250.8.110)  84.515 ms  82.245 ms  84.192 ms
 5  ia-omd-mm1.orange.md (195.22.252.6)  87.571 ms  87.560 ms  87.548 ms
 6  static.77.89.192.53.net.md (77.89.192.53)  86.910 ms  85.028 ms  85.268 ms
 7  static.77.89.192.9.net.md (77.89.192.9)  83.051 ms  27.802 ms  27.790 ms
 8  ael-202.rt.trb.csn.md.retn.net (87.245.236.82)  57.873 ms  54.915 ms  57.524 ms
 9  ae2-7.rt.ntl.kiv.ua.retn.net (87.245.233.218)  68.528 ms  58.988 ms  72.980 ms
10  209.85.148.56 (209.85.148.56)  71.135 ms  69.787 ms  58.943 ms
11  74.125.245.59 (74.125.245.59)  67.810 ms  74.125.245.75 (74.125.245.75)  65.978 ms  59.006 ms
12  74.125.245.64 (74.125.245.64)  67.060 ms  74.125.245.62 (74.125.245.62)  58.836 ms  74.125.245.86 (74.125.245.86)  67.000 ms
13  142.251.224.82 (142.251.224.82)  75.530 ms  49.528 ms  142.251.224.76 (142.251.224.76)  63.370 ms
14  192.178.81.125 (192.178.81.125)  68.406 ms  192.178.81.127 (192.178.81.127)  87.365 ms  87.352 ms
15  192.178.72.143 (192.178.72.143)  87.493 ms  87.489 ms  192.178.81.127 (192.178.81.127)  87.382 ms
16  172.253.65.39 (172.253.65.39)  87.824 ms  172.253.65.37 (172.253.65.37)  87.820 ms  87.366 ms
17  bud02s42-in-f14.1e100.net (142.251.208.142)  87.815 ms  87.764 ms  87.777 ms
```

2. Сохраним список промежуточных узлов с помощью команды **tracert -n google.com | tee tracert_google.txt**.

Здесь ключ **-n** отключает преобразование IP-адресов в доменные имена, что ускоряет выполнение, а конструкция **| tee tracert_google.txt** одновременно выводит результат на экран и записывает его в файл **tracert_google.txt**.


```

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ traceroute -n google.com | tee traceroute_google.txt
traceroute to google.com (142.251.208.142), 30 hops max, 60 byte packets
 1 172.26.16.1 0.492 ms 0.466 ms 0.565 ms
 2 192.168.8.1 31.781 ms 31.493 ms 31.437 ms
 3 10.126.13.243 32.041 ms 38.725 ms 39.195 ms
 4 10.250.8.110 39.798 ms 31.885 ms 38.182 ms
 5 195.22.252.6 44.104 ms 43.832 ms 43.824 ms
 6 77.89.192.53 43.922 ms 43.343 ms 43.844 ms
 7 77.89.192.9 43.299 ms 16.652 ms 16.440 ms
 8 87.245.236.82 53.390 ms 52.902 ms 52.822 ms
 9 87.245.233.218 46.657 ms 46.223 ms 46.971 ms
10 209.85.148.56 46.866 ms 47.667 ms 47.656 ms
11 74.125.245.75 46.539 ms 74.125.245.59 46.815 ms 46.775 ms
12 74.125.245.64 46.195 ms * 74.125.245.84 47.984 ms
13 72.14.239.111 43.268 ms 72.14.239.110 51.260 ms 142.251.224.76 51.866 ms
14 142.251.224.82 51.133 ms 192.178.81.125 50.944 ms 142.251.77.181 155.928 ms
15 172.253.65.39 51.248 ms 51.780 ms 172.253.65.37 50.729 ms
16 142.251.208.142 50.716 ms 172.253.65.37 62.662 ms 142.251.208.142 62.649 ms

```

3. Попробуем трассировку до локального сервера в сети.

Трассировка до локального сервера (192.168.1.1) с помощью команды **traceroute 192.168.1.1**

```

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ traceroute 192.168.1.1
traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 60 byte packets
 1 DESKTOP-R3SPK2V.mshome.net (172.26.16.1) 0.549 ms 0.529 ms 0.524 ms
 2 www.huaweimobilewifi.com (192.168.8.1) 2.999 ms 4.009 ms 4.004 ms
 3 10.126.13.243 (10.126.13.243) 55.412 ms 55.358 ms 55.331 ms
 4 10.250.9.57 (10.250.9.57) 55.325 ms 55.311 ms 55.314 ms
 5 ia-omd-mm1.orange.md (195.22.252.6) 55.588 ms 55.564 ms 55.802 ms
 6 static.77.89.192.53.net.md (77.89.192.53) 55.287 ms 39.440 ms 39.404 ms
 7 static.77.89.192.9.net.md (77.89.192.9) 39.686 ms static.77.89.192.213.net.md (77.89.192.213) 50.495 ms static.77.89.192.9.net.md (77.89.192.9) 50.548 ms
 8 93.122.154.66 (93.122.154.66) 50.482 ms 93.122.154.70 (93.122.154.70) 52.871 ms 93.122.154.66 (93.122.154.66) 58.999 ms
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *

```

Команда выполняет трассировку маршрута до локального устройства в сети, в данном случае до адреса 192.168.1.1. В выводе видно, что пакеты проходят через несколько промежуточных маршрутизаторов: сначала локальная сеть, затем устройства провайдера. Начиная с 9-го шага, маршрутизаторы не отвечают на запросы (обозначено символами * * *), что характерно для сетевых устройств, которые могут блокировать ICMP-запросы. В отличие от маршрута до google.com, здесь меньше активных узлов, и маршрут обрывается после устройств провайдера.

Часть 3: Порты и соединения

1. Определим, какие порты слушает система:

Команда **ss** с ключами **-tulpen** показывает список всех TCP- и UDP-портов, которые находятся в состоянии LISTEN. Здесь **-t** — TCP, **-u** — UDP, **-l** — только слушающие порты, **-p** — процесс, **-e** — расширенная информация, **-n** — цифровой формат адресов и портов. В выводе видно, что службы прослушивают порты 53 (DNS) и 323 (служебный протокол NTP). За работу на этих портах отвечает системная служба **systemd-resolved**.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ sudo ss -tulpen
[sudo] password for bogdanov18:
Sorry, try again.
[sudo] password for bogdanov18:
Netid    State    Recv-Q   Send-Q   Local Address:Port   Peer Address:Port   Process
udp      UNCONN   0         0         127.0.0.54:53        0.0.0.0:*           users(("systemd-resolve",pid
=100,fd=15)) uid:991 ino:23588 sk:1 cgroup:/system.slice/systemd-resolved.service <->
udp      UNCONN   0         0         127.0.0.53%lo:53     0.0.0.0:*           users(("systemd-resolve",pid
=100,fd=13)) uid:991 ino:23586 sk:2 cgroup:/system.slice/systemd-resolved.service <->
udp      UNCONN   0         0         10.255.255.254:53    0.0.0.0:*           ino:48 sk:3 cgroup:/ <->
udp      UNCONN   0         0         127.0.0.1:323       0.0.0.0:*           ino:18441 sk:4 cgroup:/ <->
udp      UNCONN   0         0         [::1]:323          [::]:*             ino:18442 sk:5 cgroup:/ v6onl
y:1 <->
tcp      LISTEN   0         4096      127.0.0.53%lo:53    0.0.0.0:*           users(("systemd-resolve",pid
=100,fd=14)) uid:991 ino:23587 sk:6 cgroup:/system.slice/systemd-resolved.service <->
tcp      LISTEN   0         1000     10.255.255.254:53    0.0.0.0:*           ino:49 sk:7 cgroup:/ <->
tcp      LISTEN   0         4096      127.0.0.54:53       0.0.0.0:*           users(("systemd-resolve",pid
=100,fd=16)) uid:991 ino:23589 sk:8 cgroup:/system.slice/systemd-resolved.service <->
```

2. Запустим локальный сервер для теста:

nc -l 12345 и подключимся к нему с другой вкладки терминала (**nc localhost 12345**).

Утилита **nc** (netcat) позволяет открывать соединения. В первой вкладке она была запущена в режиме сервера (**-l**), который слушает порт **12345**. Во второй вкладке выполнено подключение к этому порту с помощью **nc localhost 12345**. После этого вводимые сообщения передаются между двумя окнами терминала, что подтверждает работу TCP-соединения.

```
bogdanov18@DESKTOP-R3SPI x + v bogdanov18@DESKTOP-R3SPI x + v
28 * * * Windows PowerShell
29 * * * (C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.
30 * * * Установите последнюю версию PowerShell для новых функций и улучшения! https://aka.ms/PSWindows

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ sudo ss -tulpen
[sudo] password for bogdanov18:
Sorry, try again.
[sudo] password for bogdanov18:
Netid State Recv-Q Send-Q Local Address:Port
udp UNCONN 0 0 127.0.0.54:53
=100,fd=15)) uid:991 ino:23588 sk:1 cgroup:/system.slice/systemd-res
udp UNCONN 0 0 127.0.0.53%lo:53
=100,fd=13)) uid:991 ino:23586 sk:2 cgroup:/system.slice/systemd-res
udp UNCONN 0 0 10.255.255.254:53
udp UNCONN 0 0 127.0.0.1:323
udp UNCONN 0 0 [::]:323
y:1 <->
tcp LISTEN 0 4096 127.0.0.53%lo:53
=100,fd=14)) uid:991 ino:23587 sk:6 cgroup:/system.slice/systemd-res
tcp LISTEN 0 1000 10.255.255.254:53
tcp LISTEN 0 4096 127.0.0.54:53
=100,fd=16)) uid:991 ino:23589 sk:8 cgroup:/system.slice/systemd-res
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ nc localhost 12345
привет
это проверка?
да
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ nc -l 12345
проверка
успешная

bogdanov18@DESKTOP-R3SPI x + v bogdanov18@DESKTOP-R3SPI x + v
28 * * * Windows PowerShell
29 * * * (C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.
30 * * * Установите последнюю версию PowerShell для новых функций и улучшения! https://aka.ms/PSWindows

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ sudo ss -tulpen
[sudo] password for bogdanov18:
Sorry, try again.
[sudo] password for bogdanov18:
Netid State Recv-Q Send-Q Local Address:Port
udp UNCONN 0 0 127.0.0.54:53
=100,fd=15)) uid:991 ino:23588 sk:1 cgroup:/system.slice/systemd-res
udp UNCONN 0 0 127.0.0.53%lo:53
=100,fd=13)) uid:991 ino:23586 sk:2 cgroup:/system.slice/systemd-res
udp UNCONN 0 0 10.255.255.254:53
udp UNCONN 0 0 127.0.0.1:323
udp UNCONN 0 0 [::]:323
y:1 <->
tcp LISTEN 0 4096 127.0.0.53%lo:53
=100,fd=14)) uid:991 ino:23587 sk:6 cgroup:/system.slice/systemd-res
tcp LISTEN 0 1000 10.255.255.254:53
tcp LISTEN 0 4096 127.0.0.54:53
=100,fd=16)) uid:991 ino:23589 sk:8 cgroup:/system.slice/systemd-res
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ nc localhost 12345
привет
это проверка?
да
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ nc -l 12345
проверка
успешная
```

Передача сообщений подтверждает установку и корректную работу TCP-соединения.

Часть 4: Работа с DNS

1. Используем команду **dig** для запроса IP-адреса домена google.com.

Команда **dig** выполняет DNS-запрос для указанного домена. В разделе ANSWER SECTION видно, что имя google.com было успешно преобразовано в IP-адрес 142.251.208.142. Дополнительно указано время отклика (Query time: 102 msec) и DNS-сервер, который использовался (10.255.255.254).

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ dig google.com
; <<>> DiG 9.18.39-0ubuntu0.24.04.1-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; -->HEADER<< opcode: QUERY, status: NOERROR, id: 14538
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;google.com. IN A

;; ANSWER SECTION:
google.com. 155 IN A 142.251.208.142

;; Query time: 102 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Fri Sep 19 10:15:22 EEST 2025
;; MSG SIZE rcvd: 55
```


2. Определим, какой DNS-сервер используется системой с помощью команды *cat /etc/resolv.conf*.

Файл */etc/resolv.conf* содержит адреса DNS-серверов, которые использует система для преобразования доменных имён в IP-адреса. В моём случае там прописан сервер с адресом 10.255.255.254, что означает: именно к нему операционная система обращается при запросах к доменам (например, google.com).

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ cat /etc/resolv.conf
# This file was automatically generated by WSL. To stop automatic generation of this file, add the following entry to /etc/wsl.conf:
# [network]
# generateResolvConf = false
nameserver 10.255.255.254
```

3. Попробуем запросить MX-записи для домена gmail.com с помощью команды *dig MX gmail.com*.

Ключ **MX** указывает утилите **dig** выполнить запрос почтовых записей для указанного домена. **MX**-записи (Mail eXchanger) показывают, какие серверы обрабатывают входящую почту. У каждой записи есть приоритет: чем меньше число, тем выше приоритет у сервера. Для домена gmail.com настроено пять почтовых серверов Google с разными приоритетами, что обеспечивает отказоустойчивость и балансировку нагрузки.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ dig MX gmail.com

; <<>> DiG 9.18.39-0ubuntu0.24.04.1-Ubuntu <<>> MX gmail.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54422
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;gmail.com.                IN      MX

;; ANSWER SECTION:
gmail.com.                 3600    IN      MX      40 alt4.gmail-smtp-in.l.google.com.
gmail.com.                 3600    IN      MX      10 alt1.gmail-smtp-in.l.google.com.
gmail.com.                 3600    IN      MX      5 gmail-smtp-in.l.google.com.
gmail.com.                 3600    IN      MX      20 alt2.gmail-smtp-in.l.google.com.
gmail.com.                 3600    IN      MX      30 alt3.gmail-smtp-in.l.google.com.

;; Query time: 102 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Fri Sep 19 10:22:25 EEST 2025
;; MSG SIZE rcvd: 161
```

Часть 5: Мини-проект «Сетевой отчёт»

Каждый студент выбирает один сайт (например, github.com) и готовит:

1. IP-адреса и DNS-записи сайта.

```
bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ dig github.com

; <<>> DiG 9.18.39-0ubuntu0.24.04.1-Ubuntu <<>> github.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33609
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;github.com.                IN      A

;; ANSWER SECTION:
github.com.                 60      IN      A      140.82.121.4

;; Query time: 91 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Fri Sep 19 10:35:55 EEST 2025
;; MSG SIZE rcvd: 55
```

Команда **dig** выполняет DNS-запрос для домена github.com. В ответе мы видим раздел ANSWER SECTION, где указано, что доменное имя **github.com** преобразуется в IP-адрес 140.82.121.4. В нижней части вывода также отображается, что запрос обработал DNS-сервер 10.255.255.254.

2. Трассировка до сервера.

Команда **traceroute** показывает маршрут, по которому пакеты идут от моей машины (172.26.16.1) к серверу **GitHub** (140.82.121.4).

Первые узлы — это локальная сеть и провайдер.

Далее пакеты проходят через магистральные маршрутизаторы (например, ia-omd-mm1.orange.md, zayo.com).

Часть маршрутизаторов не отвечает на ICMP-запросы, поэтому в выводе отображаются * * *. Это не ошибка, а особенность настройки оборудования.

В конце пакеты доходят до сети, где размещены серверы GitHub.

```

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ traceroute github.com
traceroute to github.com (140.82.121.4), 30 hops max, 60 byte packets
 1 DESKTOP-R3SPK2V.mshome.net (172.26.16.1) 0.519 ms 0.998 ms 0.859 ms
 2 www.huaweimobilewifi.com (192.168.8.1) 3.084 ms 2.071 ms 2.543 ms
 3 10.126.13.243 (10.126.13.243) 47.603 ms 47.531 ms 47.458 ms
 4 10.250.8.110 (10.250.8.110) 47.413 ms 47.331 ms 47.291 ms
 5 ia-omd-mm1.orange.md (195.22.252.6) 47.644 ms 47.560 ms 47.486 ms
 6 static.77.89.192.53.net.md (77.89.192.53) 47.596 ms 45.903 ms 45.861 ms
 7 static.77.89.192.9.net.md (77.89.192.9) 45.451 ms 45.867 ms 45.177 ms
 8 89.121.199.157 (89.121.199.157) 52.322 ms 35.532 ms 36.157 ms
 9 10.0.246.69 (10.0.246.69) 38.082 ms 37.431 ms 10.0.246.101 (10.0.246.101) 36.591 ms
10 * * *
11 * * *
12 * * *
13 * * *
14 * ae27.cs1.fra9.de.eth.zayo.com (64.125.30.254) 71.524 ms 70.581 ms
15 ae1.mcs1.fra9.de.zip.zayo.com (64.125.29.65) 103.157 ms 59.147 ms 56.773 ms
16 82.98.193.29 (82.98.193.29) 65.512 ms 136.209 ms 134.816 ms
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *

```

3. Список открытых портов с помощью команды **nc -zv github.com 22 80 443 8080**.

Команда **nc -zv github.com 22 80 443 8080** выполняет проверку доступности основных портов на сервере github.com. В результате видно, что открыты порты 22 (SSH), 80 (HTTP) и 443 (HTTPS), а порт 8080 недоступен. Это означает, что GitHub поддерживает защищённое соединение через HTTPS и работу с репозиториями по протоколу SSH, а также использует HTTP для перенаправления.

```

bogdanov18@DESKTOP-R3SPK2V:/mnt/c/Users/Asus$ nc -zv github.com 22 80 443 8080
Connection to github.com (140.82.121.4) 22 port [tcp/ssh] succeeded!
Connection to github.com (140.82.121.4) 80 port [tcp/http] succeeded!
Connection to github.com (140.82.121.4) 443 port [tcp/https] succeeded!
|

```

4. Заголовки HTTP-ответа.

Команда **curl -I https://github.com** выполняет HTTP-запрос методом HEAD и выводит только заголовки ответа сервера. В результате отображается код состояния (например, HTTP/2 200 или 301 Moved Permanently), информация о сервере, тип содержимого, дата ответа и другие служебные параметры. Эти заголовки позволяют понять, как сервер обрабатывает запрос: используется ли перенаправление, какой протокол и версия HTTP применяются, а также как настраивается безопасность соединения. В случае с github.com сервер отвечает

через HTTPS и использует современные стандарты HTTP/2, что подтверждает защищённое и оптимизированное соединение.

```
bogdanov18@DESKTOP-R35PK2V:/mnt/c/Users/Asus$ curl -I https://github.com
HTTP/2 200
date: Fri, 19 Sep 2025 07:46:17 GMT
content-type: text/html; charset=utf-8
vary: X-PJAX, X-PJAX-Container, Turbo-Visit, Turbo-Frame, X-Requested-With, Accept-Language, Accept-Encoding, Accept, X-Requested-With
content-language: en-US
etag: W/"5ccad1f14822078b5a72e38e44d56ec"
cache-control: max-age=0, private, must-revalidate
strict-transport-security: max-age=31536000; includeSubdomains; preload
x-frame-options: deny
x-content-type-options: nosniff
x-xss-protection: 0
referrer-policy: origin-when-cross-origin, strict-origin-when-cross-origin
content-security-policy: default-src 'none'; base-uri 'self'; child-src github.githubassets.com github.com/assets-cdn/worker/ github.com/assets/ gist.github.com/assets-cdn/worker/; connect-src 'self' uploads.g
ithub.com www.githubstatus.com collector.github.com raw.githubusercontent.com api.github.com github-cloud.s3.amazonaws.com github-production-repository-file-5c1aeb.s3.amazonaws.com github-production-uploa
d-manifest-file-7f6dc7.s3.amazonaws.com github-production-user-asset-6210df.s3.amazonaws.com *.rel.tunnels.api.visualstudio.com github.githubassets.com objects-origin.github
usercontent.com copilot-proxy.githubusercontent.com proxy.individual.githubcopilot.com proxy.business.githubcopilot.com proxy.enterprise.githubcopilot.com *.actions.githubusercontent.com wss://*.actions.githu
busercontent.com productionresultssa8.blob.core.windows.net/ productionresultssa11.blob.core.windows.net/ productionresultssa2.blob.core.windows.net/ productionresultssa3.blob.core.windows.net/ productionresults
sa4.blob.core.windows.net/ productionresultssa5.blob.core.windows.net/ productionresultssa6.blob.core.windows.net/ productionresultssa7.blob.core.windows.net/ productionresultssa8.blob.core.windows.net/ produc
tionresultssa9.blob.core.windows.net/ productionresultssa10.blob.core.windows.net/ productionresultssa11.blob.core.windows.net/ productionresultssa12.blob.core.windows.net/ productionresultssa13.blob.core.wind
ows.net/ productionresultssa14.blob.core.windows.net/ productionresultssa15.blob.core.windows.net/ productionresultssa16.blob.core.windows.net/ productionresultssa17.blob.core.windows.net/ productionresultssa1
8.blob.core.windows.net/ productionresultssa19.blob.core.windows.net/ github-production-repository-image-32feaf.s3.amazonaws.com github-production-release-asset-266b8e.s3.amazonaws.com insights.github.com wss:
//alive.github.com wss://alive-staging.github.com api.githubcopilot.com api.individual.githubcopilot.com api.business.githubcopilot.com api.enterprise.githubcopilot.com edge.fullstory.com rs.fullstory.com; fen
t-src github.githubassets.com; form-action 'self' github.com gist.github.com copilot-workspace.githubnext.com objects-origin.githubusercontent.com; frame-ancestors 'none'; frame-src viewscreen.githubuserconten
t.com notebooks.githubusercontent.com www.youtube-nocookie.com; img-src 'self' data: blob: github.githubassets.com media.githubusercontent.com camo.githubusercontent.com avatars.githubuse
rcontent.com private-avatars.githubusercontent.com github-cloud.s3.amazonaws.com objects.githubusercontent.com release-assets.githubusercontent.com secured-user-images.githubusercontent.com/ user-images.github
usercontent.com/ private-user-images.githubusercontent.com openpgp.githubassets.com marketplace-screenshots.githubusercontent.com/ copilotprodattachments.blob.core.windows.net/github-production-copilot-attac
hments/ github-production-user-asset-6210df.s3.amazonaws.com customer-stories-feed.github.com spotlights-feed.github.com objects-origin.githubusercontent.com *.githubusercontent.com images.ctfassets.net/8aevph
vgw8t8/; manifest-src 'self'; media-src github.com user-images.githubusercontent.com/ secured-user-images.githubusercontent.com/ private-user-images.githubusercontent.com github-production-user-asset-6210df.s3
.amazonaws.com gist.github.com github.githubassets.com assets.ctfassets.net/8aevphvgw8t8/ videos.ctfassets.net/8aevphvgw8t8/; script-src github.githubassets.com; style-src 'unsafe-inline' github.githubassets.c
om; upgrade-insecure-requests; worker-src github.githubassets.com github.com/assets-cdn/worker/ github.com/assets/ gist.github.com/assets-cdn/worker/
server: github.com
accept-ranges: bytes
set-cookie: gh_sess=8NGlyGqXEMzbnslpT7z5WlmgNuUxV1JerbGUCWicHwQVeEOzRYMaegpmJ5ekigHjetNeK2B%2F5JR6%2FNOKjd1V5LVlBxwLxaFELIaDciwSPn%2BLc1SzIdyzoGdyD815K%2B8%5FJAJEjPL%2Bq4qlsFzomXWcJDF5qiiw3KbUwNg21H1NAj
fHYVKGzpcn%2BqYqmsADVGvubI1zs29bwfj3w3RBY3Ep3N25G%2FL66RbXRPkWhLk1jrF2Lfk4wSkpKmpI6N8BspxFcqDBNrEwWw%3D%3D--s3L%2F9qEP9CLmPJtv--w9PatV1MeuOhrT72AOS%2Bg%3D%3D; Path=/; HttpOnly; Secure; SameSite=Lax
set-cookie: octo=GH1.1.1511738997/1758267985; Path=/; Domain=github.com; Expires=Sat, 19 Sep 2026 07:46:25 GMT; Secure; SameSite=Lax
set-cookie: logged_in=no; Path=/; Domain=github.com; Expires=Sat, 19 Sep 2026 07:46:25 GMT; HttpOnly; Secure; SameSite=Lax
x-github-request-id: EBSC-37DAE5:HA8E74:3E9D22:68CD8A51
```

5. SSL-сертификат (действителен ли?).

Проверка SSL-сертификата с помощью команды **echo | openssl s_client -connect github.com:443 -servername github.com 2>/dev/null | openssl x509 -noout -dates -issuer -subject**

Команда **echo | openssl s_client -connect github.com:443 -servername github.com 2>/dev/null | openssl x509 -noout -dates -issuer -subject** выполняет подключение к серверу и выводит данные его сертификата. Часть **echo |** передаёт пустую строку, чтобы соединение сразу завершилось. Утилита **openssl** в режиме **s_client** подключается к серверу по адресу **github.com** и порту **443**, а параметр **-servername github.com** заставляет сервер выдать сертификат именно для этого домена. Перенаправление **2>/dev/null** скрывает лишние сообщения об установке соединения. Далее результат передаётся в **openssl x509**, где ключи **-noout -dates -issuer -subject** выводят срок действия, организацию-издателя и назначение сертификата.

Из вывода видно, что сертификат действует с 5 февраля 2025 года по 5 февраля 2026 года, выдан организацией Sectigo Limited и предназначен для домена **github.com**.

```
bogdanov18@DESKTOP-R35PK2V:/mnt/c/Users/Asus$ echo | openssl s_client -connect github.com:443 -servername github.com 2>/dev/null | openssl x509 -noout -dates -issuer -subject
notBefore=Feb  5 00:00:00 2025 GMT
notAfter=Feb  5 23:59:59 2026 GMT
issuer=C = GB, ST = Greater Manchester, L = Salford, O = Sectigo Limited, CN = Sectigo ECC Domain Validation Secure Server CA
subject=CN = github.com
```

4. Контрольные вопросы

1. Чем отличаются частные и публичные IP-адреса?

Частные IP-адреса используются только внутри локальных сетей (например, 192.168.x.x или 10.x.x.x) и не видны в интернете. Публичные IP-адреса уникальны во всей сети Интернет и позволяют устройству быть доступным извне.

2. Для чего нужны порты и какие протоколы их используют?

Порты позволяют различать разные службы на одном устройстве. Например, веб-сайт работает на 80-м или 443-м порту, почта — на 25-м. Порты используют протоколы TCP (надёжная передача, например HTTPS, SSH) и UDP (быстрая передача без подтверждения, например DNS, DHCP).

3. Как работает DNS?

DNS переводит удобные для человека доменные имена (google.com) в IP-адреса (142.251.208.142). Запрос отправляется на DNS-сервер, который либо знает ответ, либо пересылает запрос дальше. После получения IP-адреса устанавливается соединение с нужным сервером.

4. Как определить, открыт ли порт на удалённом хосте?

Проверить открытые порты можно с помощью утилит: ``nc -zv адрес порт``, ``telnet адрес порт`` или ``nmap``. Если соединение устанавливается — порт открыт, если выдаётся ошибка — порт закрыт.

Вывод

В ходе выполнения лабораторной работы были освоены базовые приёмы диагностики и анализа сетей в Linux. С помощью утилиты `ip` определены IP- и MAC-адреса интерфейсов, а также изучена таблица маршрутизации. Команда `ping` позволила проверить доступность как отдельных IP-адресов, так и доменных имён, что показало важность работы DNS. Утилита `traceroute` продемонстрировала путь прохождения пакетов до внешних и локальных узлов.

Было изучено, какие порты слушает система, а также организовано простое соединение клиент–сервер с помощью `nc`, что на практике показало работу TCP-соединений. С помощью `dig` проведены DNS-запросы для определения IP-адресов и MX-записей, а также выявлен используемый системой DNS-сервер.

В мини-проекте на примере сайта `github.com` получены IP-адреса и DNS-записи, выполнена трассировка маршрута, проверена доступность основных портов, проанализированы заголовки HTTP-ответа и изучен SSL-сертификат. Это подтвердило использование современных стандартов безопасности и отказоустойчивости.

Таким образом, были закреплены навыки работы с сетевыми инструментами, понимание принципов маршрутизации, работы портов, DNS и HTTPS. Полученные знания могут быть использованы для диагностики сетевых проблем, настройки сервисов и повышения безопасности работы в сети.