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.

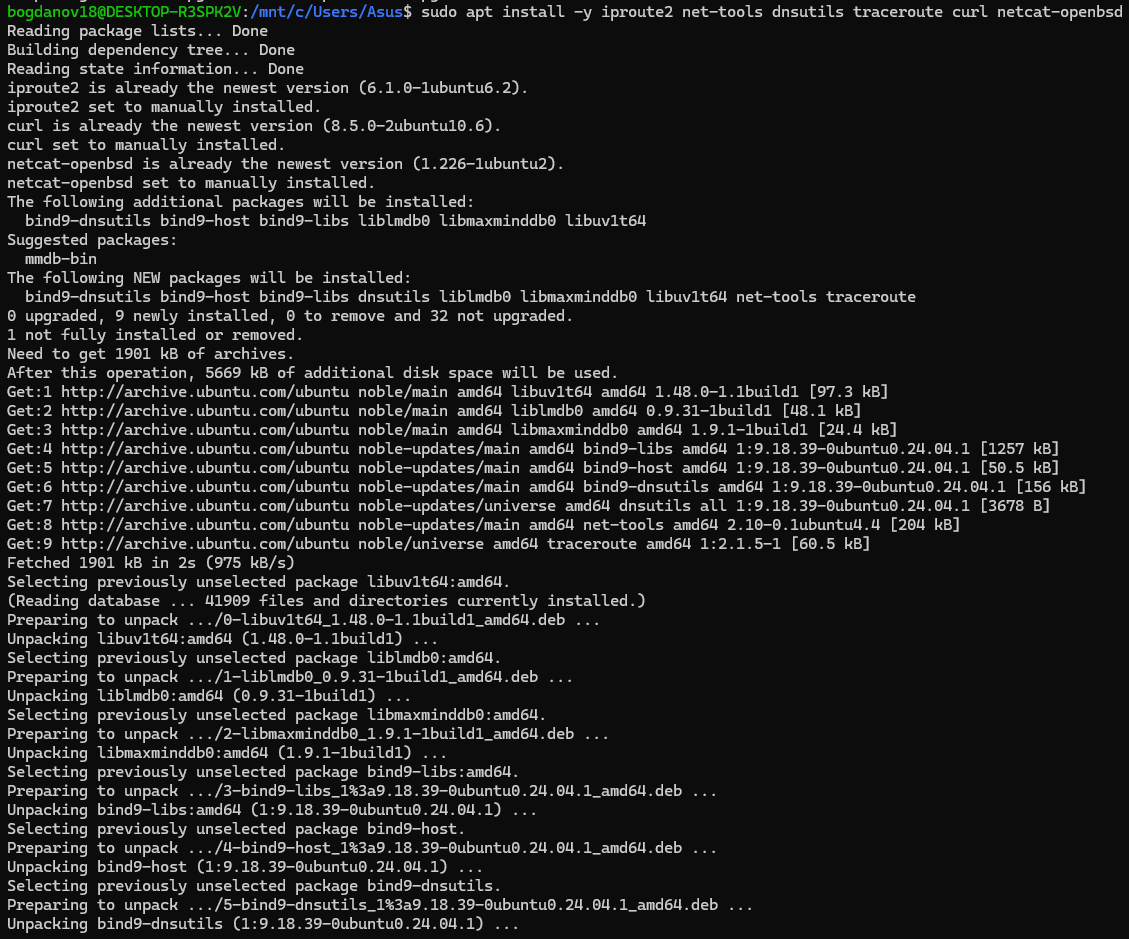
• MAC-адрес.

• Маршрутизация: таблица маршрутов.

• DNS: преобразование имён в IP-адреса.

• Утилиты: ip, ping, traceroute, ss, netstat, dig, curl, nc.

C:\Users\Asus\OneDrive\Изображения\Снимки экрана\Снимок экрана 2025-09-19 084818.png**Подготовка окружения**

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.

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.

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.

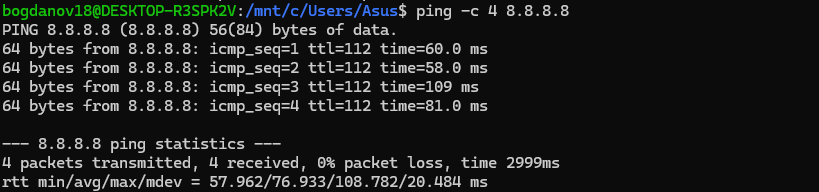
Команда показывает состояние сетевых интерфейсов и их MAC-адреса. Интерфейс lo не имеет физического адреса, так как он виртуальный, а интерфейс eth0 имеет MAC-адрес 00:15:5d:7a:f1:a5.

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.

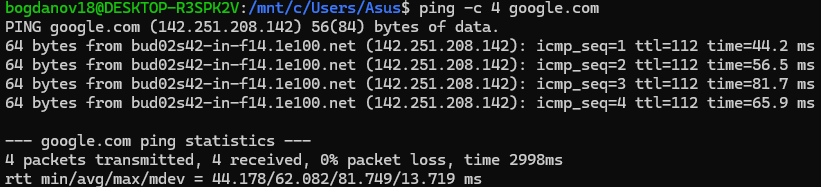
3. Проверим доступность узла 8.8.8.8 и сайта google.com с помощью **ping**.

- **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).

**- 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.



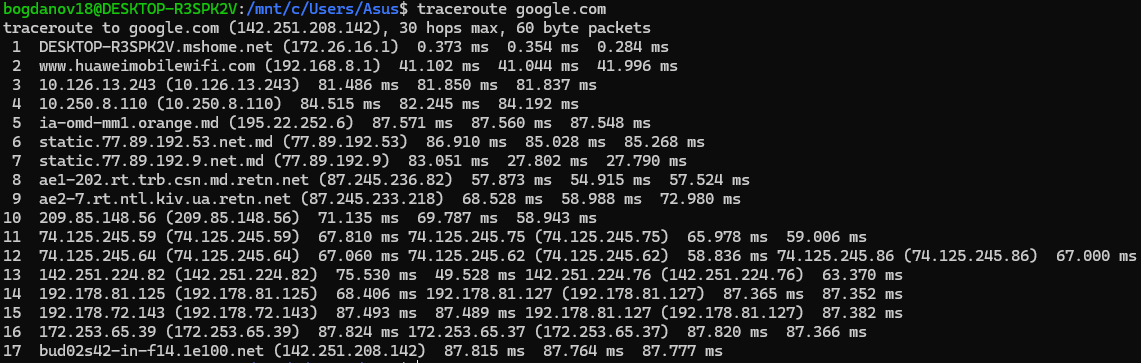
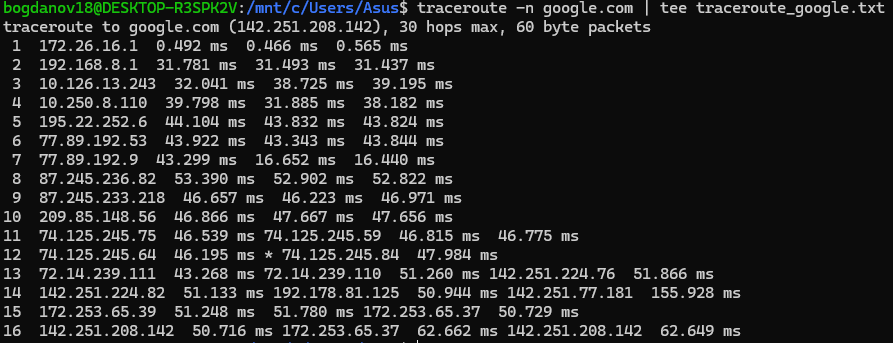
4. Сравните результаты: что произойдёт, если DNS не работает?

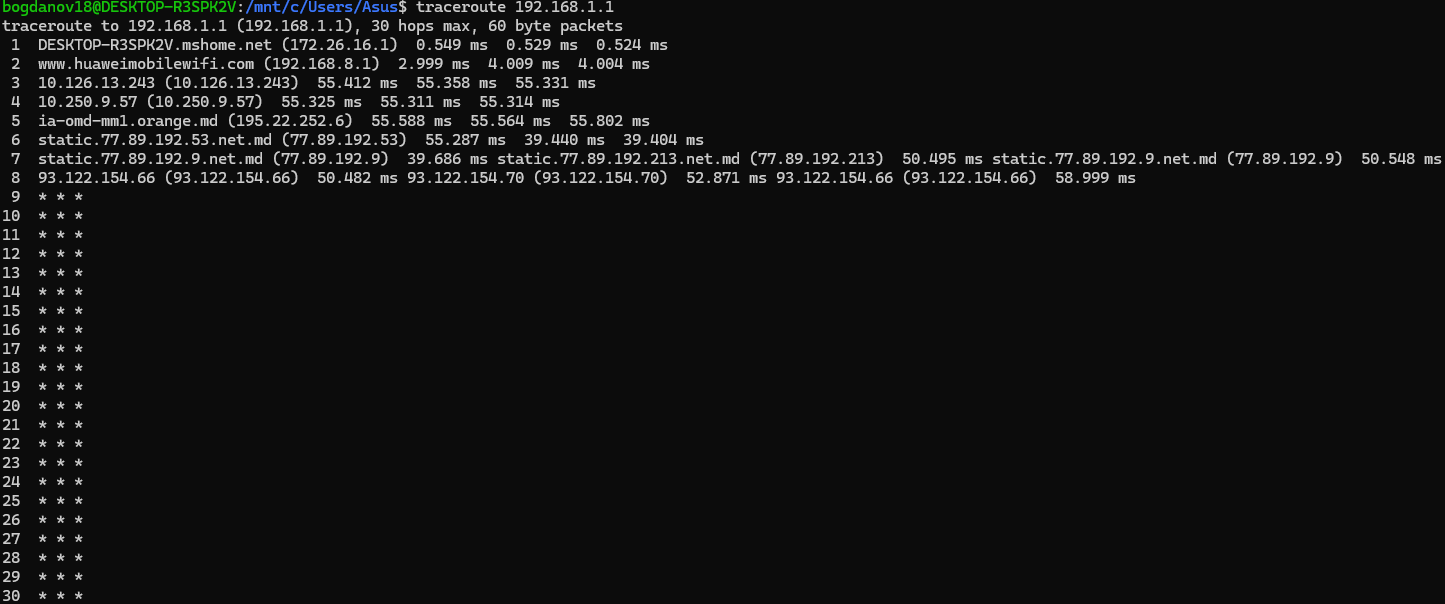
### Сравнение результатов

* 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 (**traceroute**) до google.com.

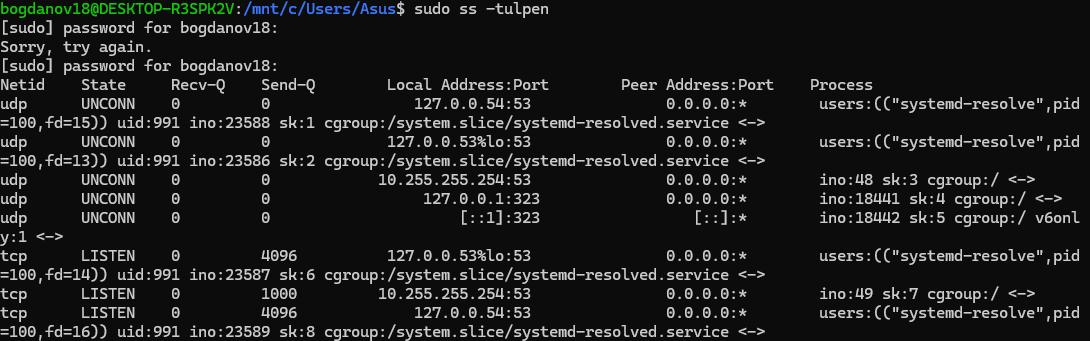
Утилита **traceroute** показывает, по каким промежуточным маршрутизаторам проходят пакеты, пока не достигнут конечного адресата. Каждый «hop» в выводе соответствует сетевому устройству, через которое проходят данные. В моём случае видно 17 шагов от локального компьютера до IP-адреса google.com (142.251.208.142). 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.  
  
2. Сохраним список промежуточных узлов с помощью команды traceroute ***-n google.com | tee traceroute\_google.txt***.  
   
Здесь ключ **-n** отключает преобразование IP-адресов в доменные имена, что ускоряет выполнение, а конструкция **| tee traceroute\_google.txt** одновременно выводит результат на экран и записывает его в файл **traceroute\_google.txt**.  
  
  
3. Попробуем трассировку до локального сервера в сети.

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

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**

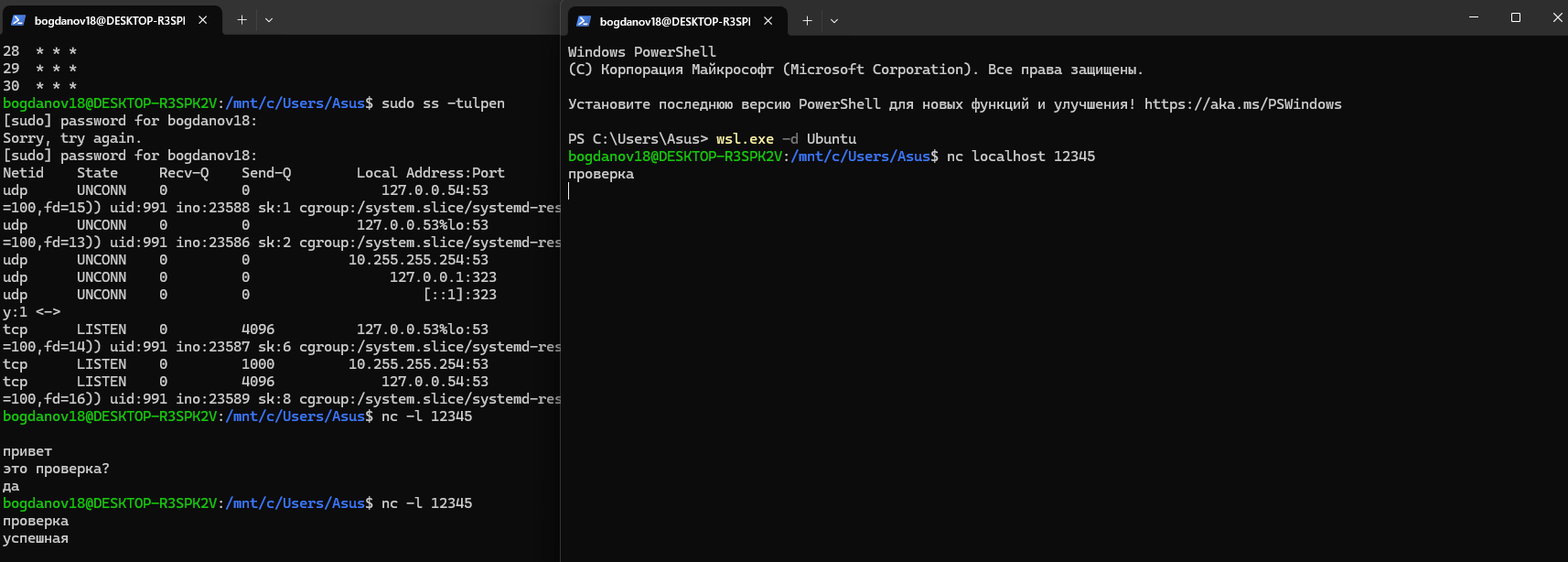
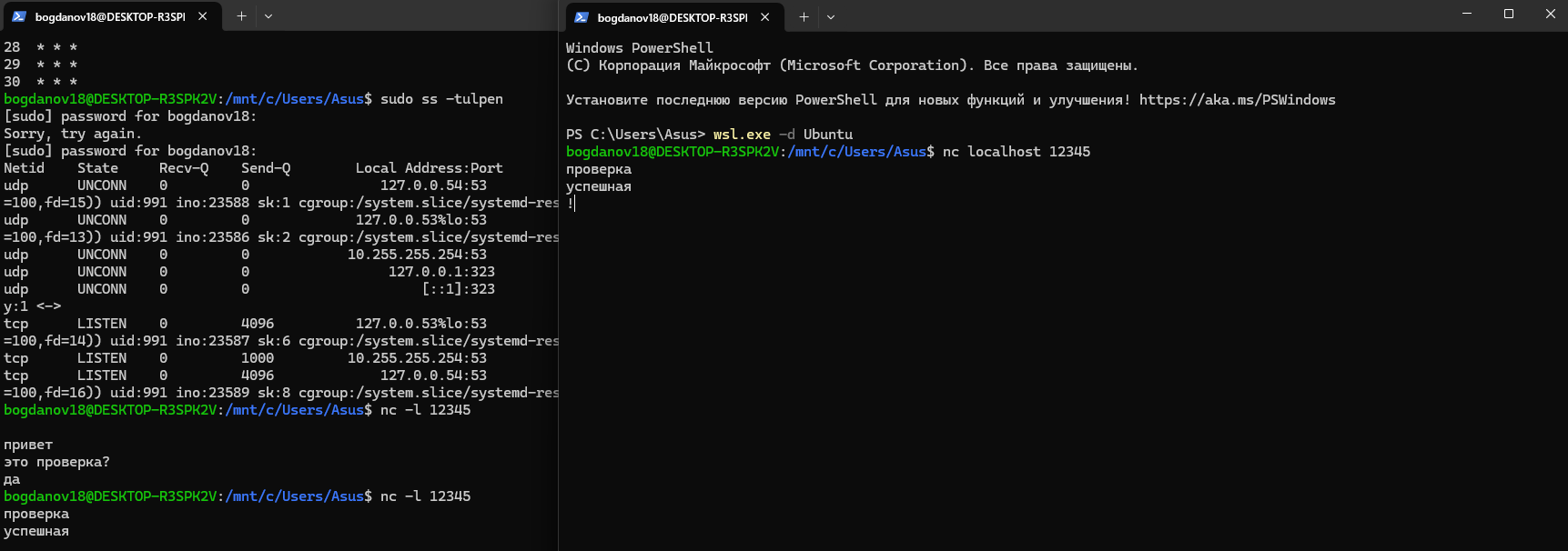
1. Определим, какие порты слушает система:

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.

2. Запустим локальный сервер для теста:

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

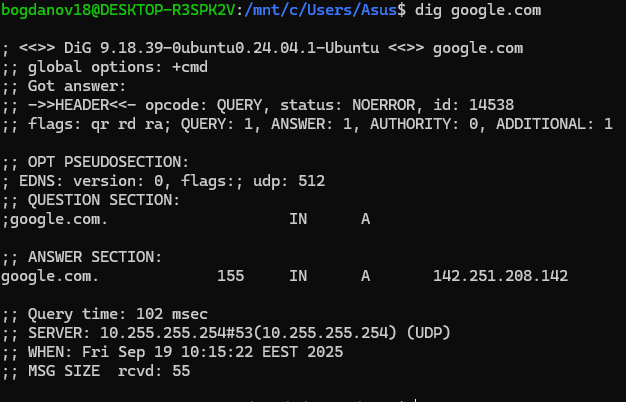
Утилита **nc** (netcat) позволяет открывать соединения. В первой вкладке она была запущена в режиме сервера (**-l**), который слушает порт **12345**. Во второй вкладке выполнено подключение к этому порту с помощью **nc localhost 12345**. После этого вводимые сообщения передаются между двумя окнами терминала,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

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.

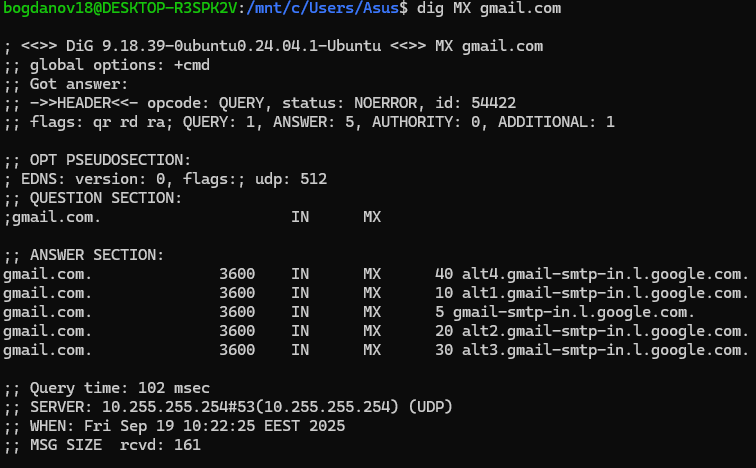
Команда **dig** выполняет DNS-запрос для указанного домена. В разделе ANSWER SECTION видно, что имя google.com было успешно преобразовано в IP-адрес 142.251.208.142. Дополнительно указано время отклика (Query time: 102 msec) и DNS-сервер, который использовался (10.255.255.254).



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

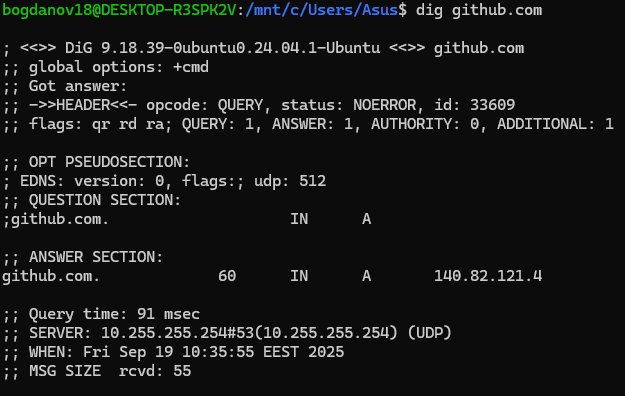
Файл **/etc/resolv.conf** содержит адреса DNS-серверов, которые использует система для преобразования доменных имён в IP-адреса. В моём случае там прописан сервер с адресом 10.255.255.254, что означает: именно к нему операционная система обращается при запросах к доменам (например, google.com).

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.

**Часть 5: Мини-проект «Сетевой отчёт»**

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

**1. IP-адреса и DNS-записи сайта.**  
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.**

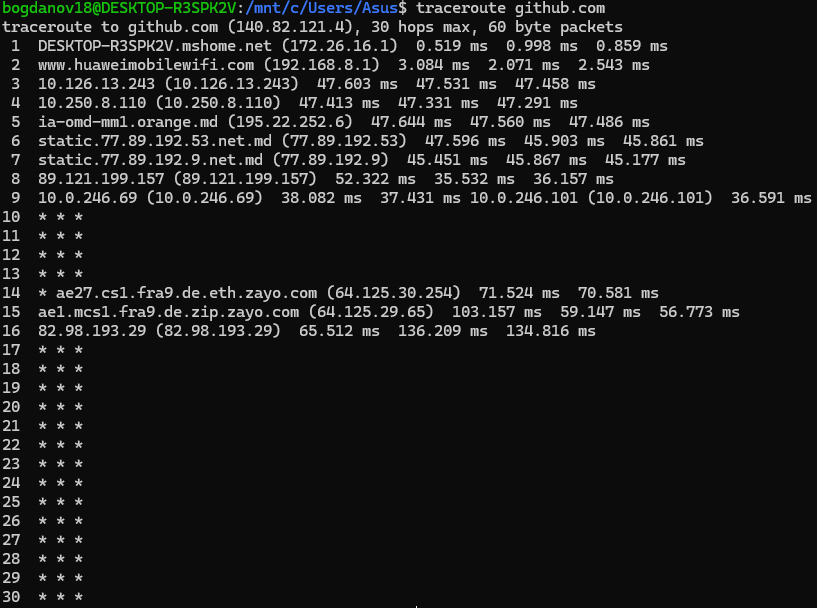
Команда **traceroute** показывает маршрут, по которому пакеты идут от моей машины (172.26.16.1) к серверу **GitHub** (140.82.121.4).

Первые узлы — это локальная сеть и провайдер.

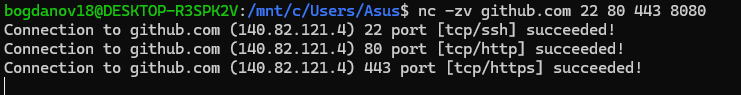
Далее пакеты проходят через магистральные маршрутизаторы (например, ia-omd-mm1.orange.md, zayo.com).

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

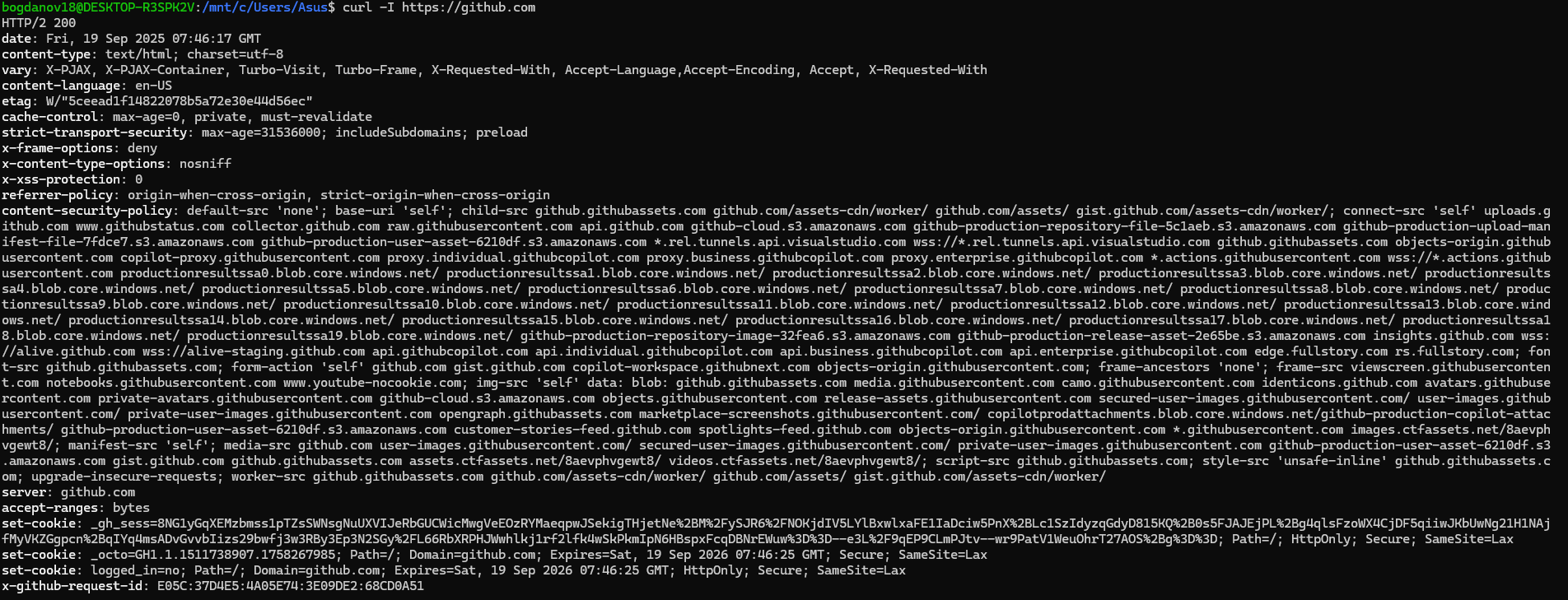
В конце пакеты доходят до сети, где размещены серверы GitHub.



**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.

**4. Заголовки HTTP-ответа.**

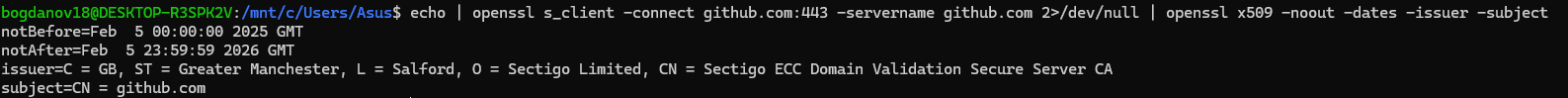
Команда **curl -I https://github.com** выполняет HTTP-запрос методом HEAD и выводит только заголовки ответа сервера. В результате отображается код состояния (например, HTTP/2 200 или 301 Moved Permanently), информация о сервере, тип содержимого, дата ответа и другие служебные параметры. Эти заголовки 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.

**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.



**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.