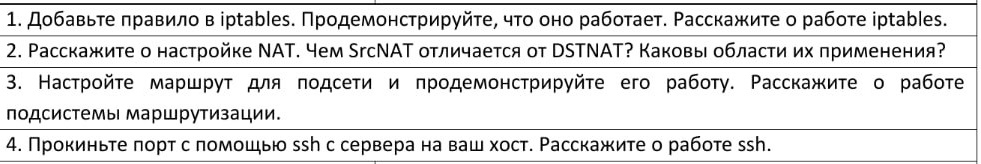
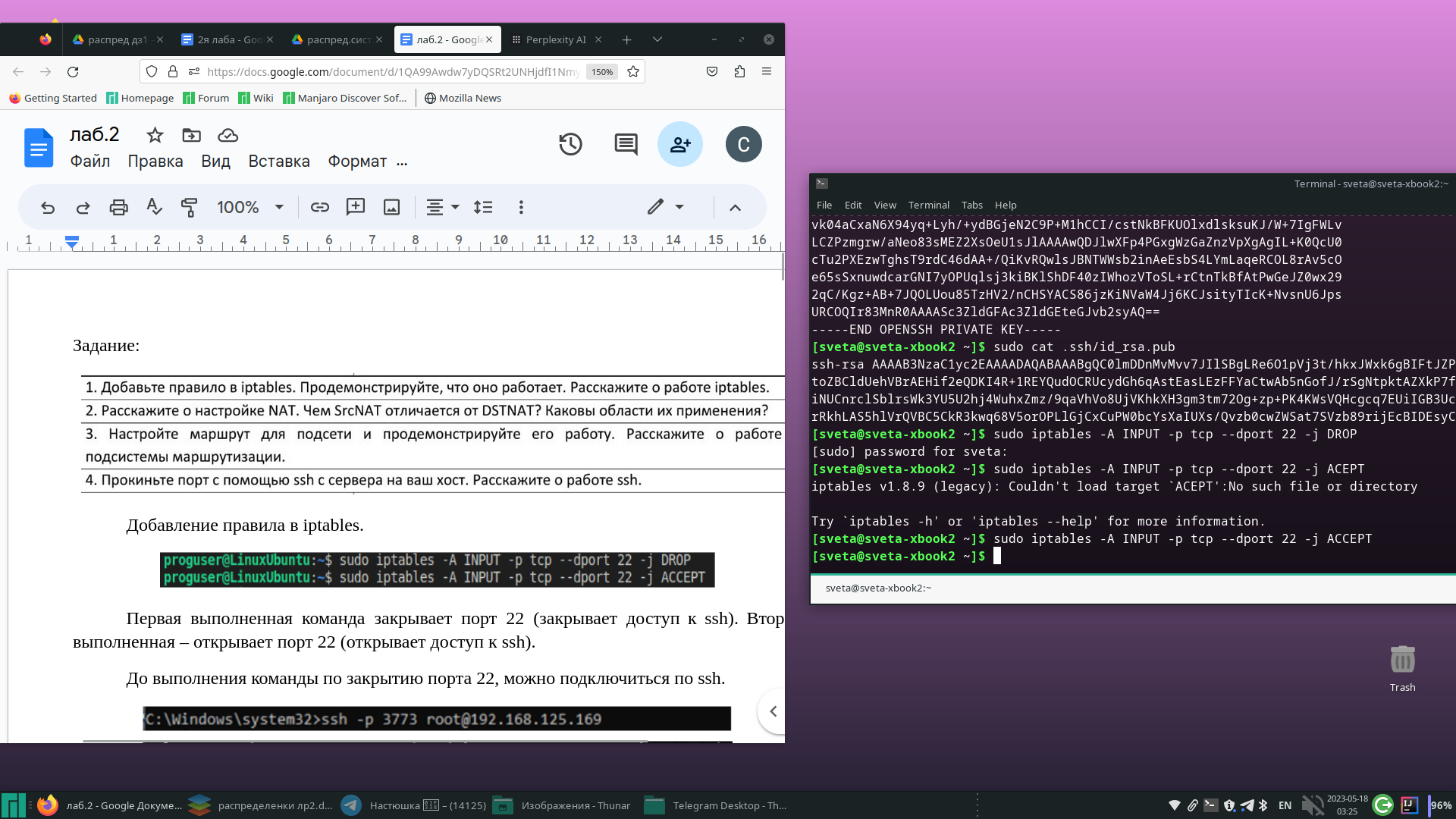
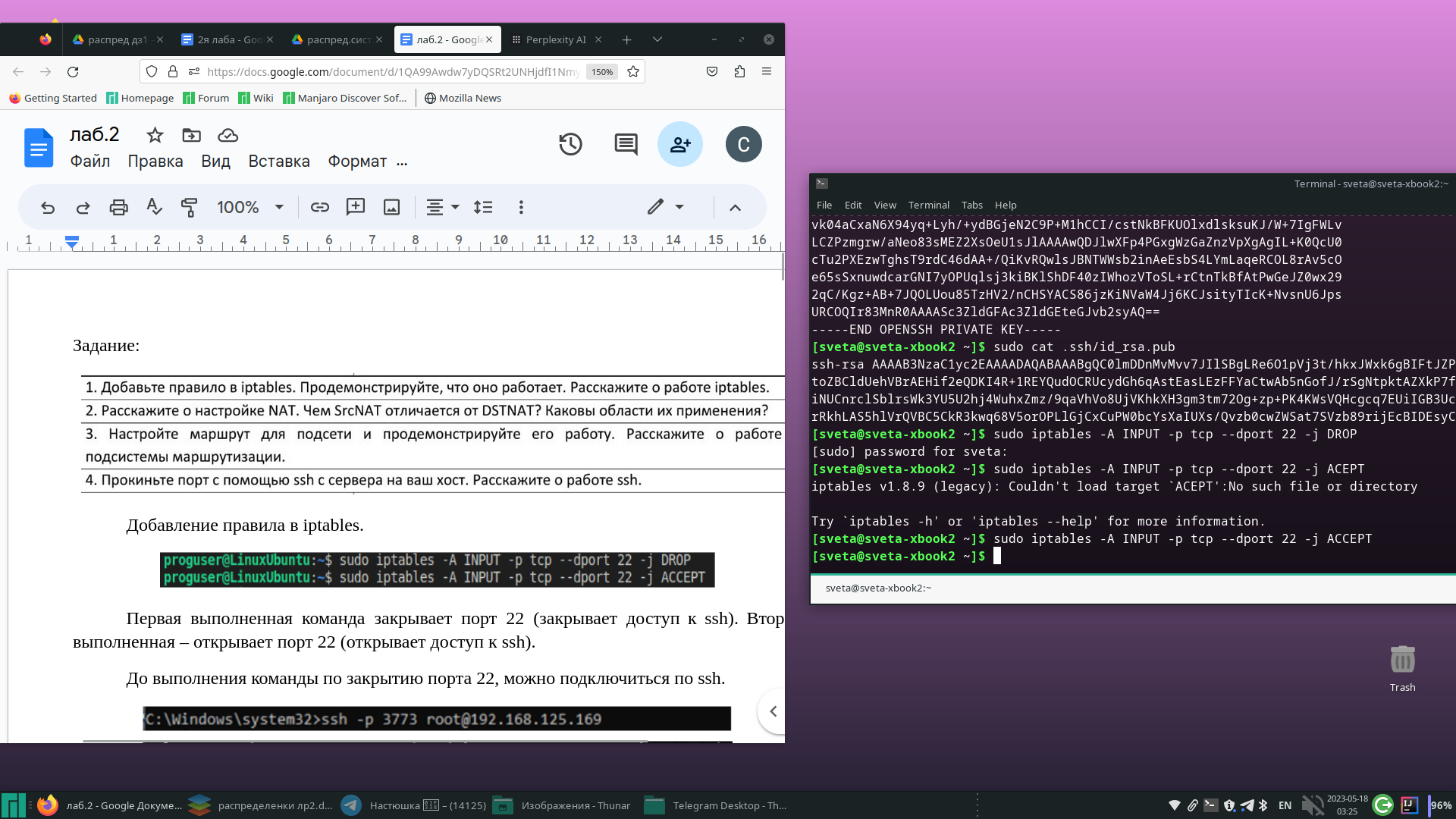
Задание:



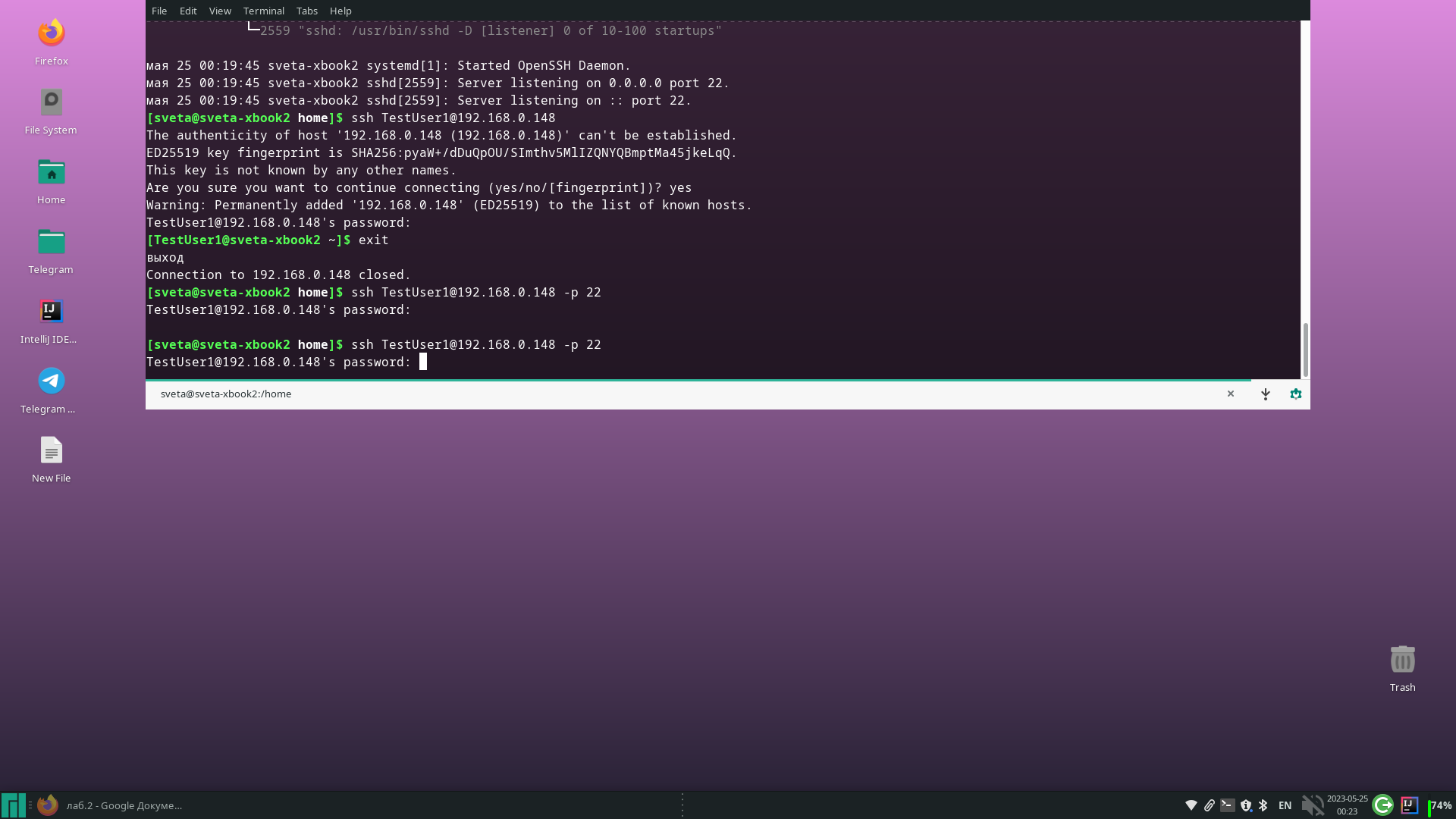
1. Добавление правила в iptables.

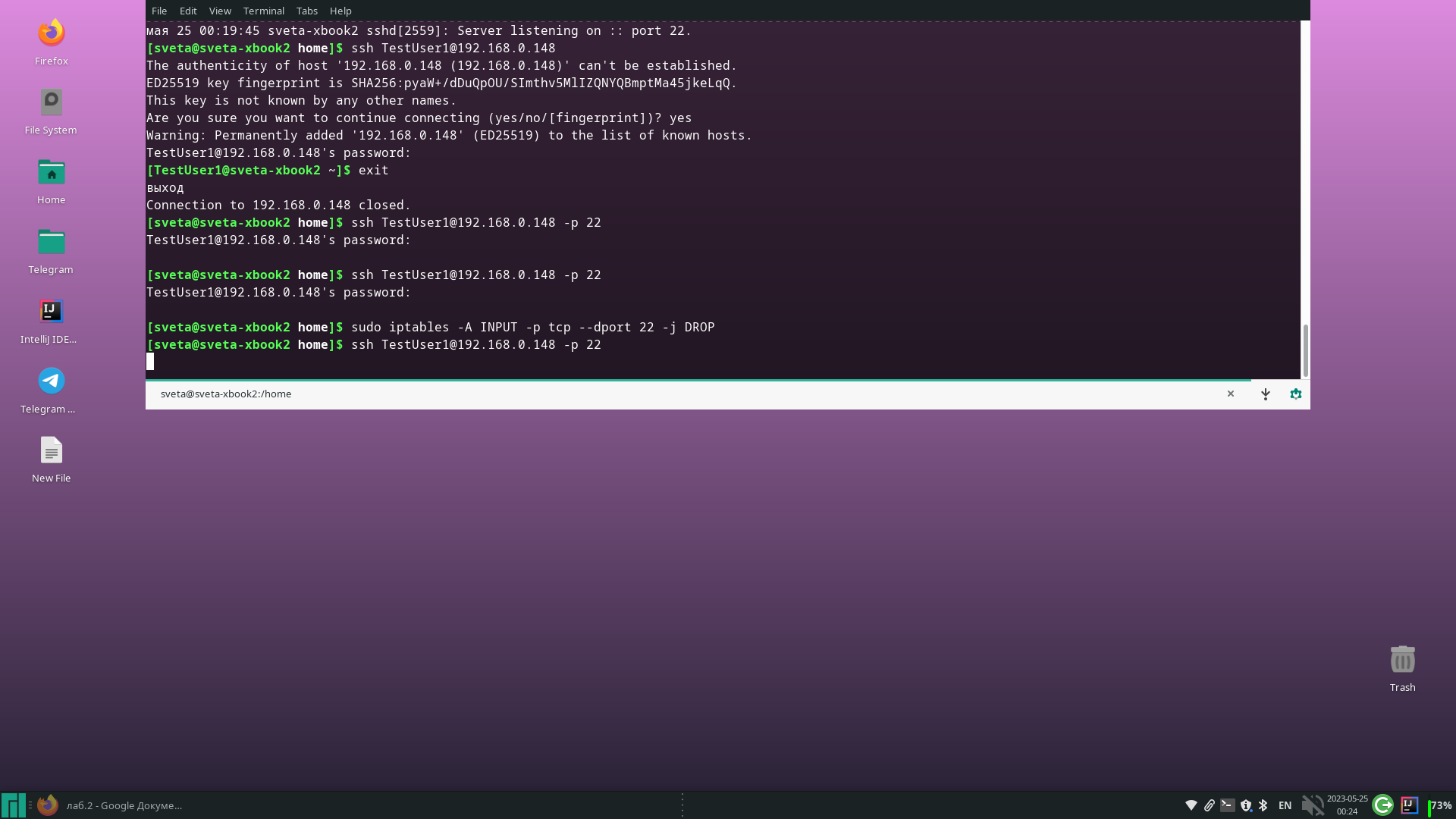


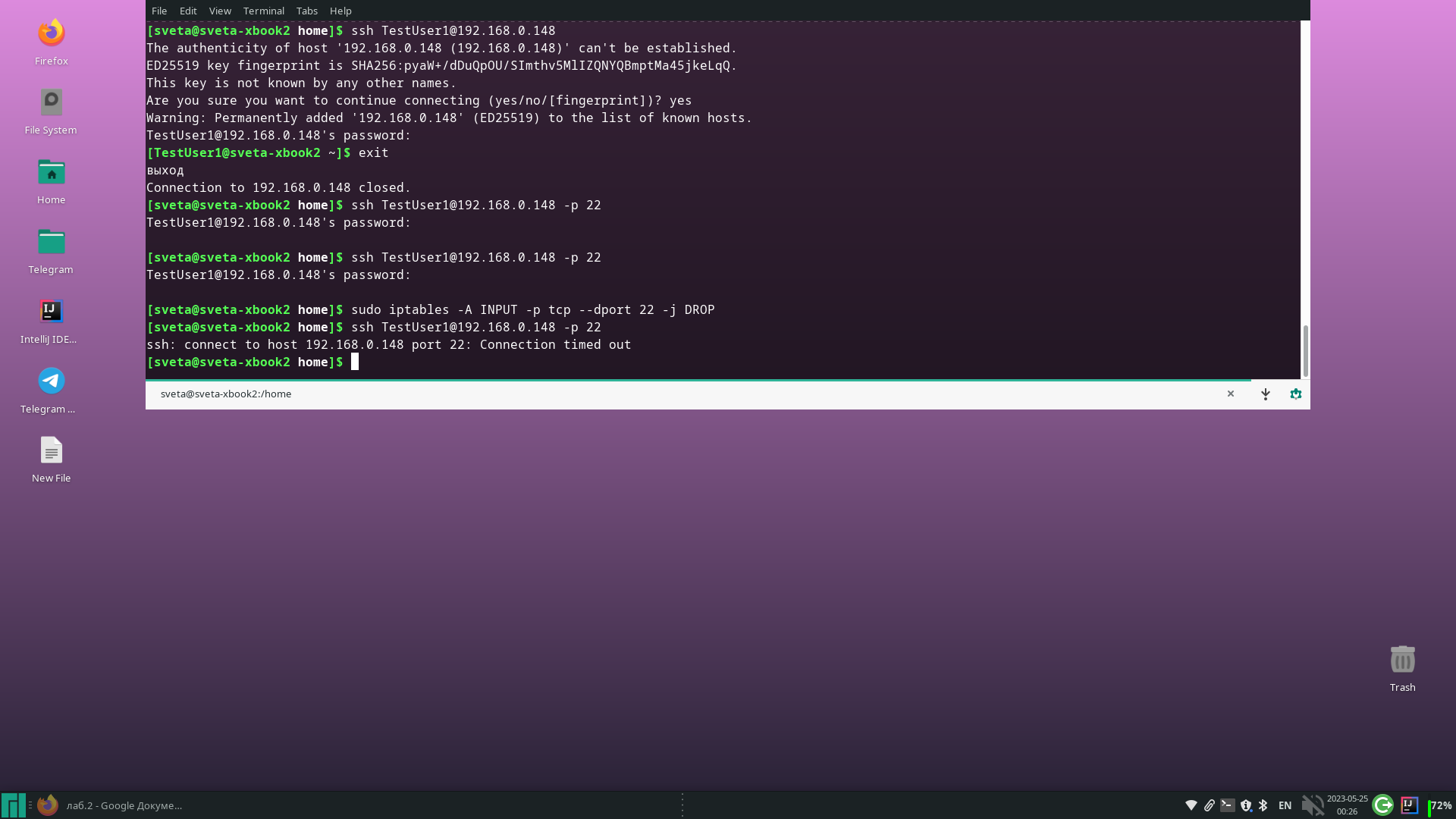


Первая выполненная команда закрывает порт 22 (закрывает доступ к ssh). Вторая выполненная – открывает порт 22 (открывает доступ к ssh).

До выполнения команды по закрытию порта 22, можно подключиться по ssh.







Iptables - это утилита командной строки, которая является стандартным интерфейсом управления работой межсетевого экрана (брандмауэра) netfilter для ядер Linux версий 2.4 и выше. Она используется для настройки правил фильтрации пакетов в Linux. Работа iptables основана на использовании цепочек и правил. Цепочки содержат списки правил, которые определяют, что делать с пакетами, проходящими через цепочку.

Правила могут блокировать или разрешать доступ к определенным портам, IP-адресам и протоколам. Для работы с iptables необходимо знать синтаксис команд и опций. Общий вид управляющей команды: iptables [опции] [цепочка] [действие] [параметры].

Например, чтобы добавить правило в цепочку INPUT для блокировки доступа к порту 22, нужно выполнить команду: iptables -A INPUT -p tcp --dport 22 -j DROP.

Чтобы сохранить настройки iptables после перезагрузки сервера, необходимо использовать утилиту iptables-save.

В фильтре iptables все пакеты делятся на три цепочки:

- input – обрабатывает входящие пакеты и подключения;

- forward – применяется для проходящих соединений;

- output – используется для исходящих пакетов и соединений.

Есть еще две дополнительные цепочки правил:

- prerouting – в эту цепочку пакет попадает перед обработкой iptables, система еще не знает куда он будет отправлен, в input, output или forward;

- postrouting – сюда попадают все проходящие пакеты, которые уже прошли цепочку forward.

Действия в правиле бывают следующие:

- ACCEPT - разрешить прохождение пакета дальше по цепочке правил;

- DROP - удалить пакет;

- REJECT - отклонить пакет, отправителю будет отправлено сообщение, что пакет был отклонен;

- LOG - сделать запись о пакете в лог файл;

- QUEUE - отправить пакет пользовательскому приложению.

Основные действия, которые позволяет выполнить iptables:

- A – добавить правило в цепочку;

- С – проверить все правила;

- D – удалить правило;

- I – вставить правило с нужным номером;

- L – вывести все правила в текущей цепочке;

- S – вывести все правила;

- F – очистить все правила;

- N – создать цепочку;

- X – удалить цепочку;

- P – установить действие по умолчанию.

1. Настройка NAT

Сначала нужно включить поддержку пересылки пакетов в /etc/sysctl.conf, чтобы трафик мог ходить между разными сетевыми интерфейсами.

Проверка текущего состояния:

sysctl net.ipv4.conf.all.forwarding

cat /proc/sys/net/ipv4/ip\_forward

Если оно равно 0, то включить его можно следующей командой:

sysctl -w net.ipv4.conf.all.forwarding=1

Чтобы после перезапуска системы оно не сбросилось, в файл /etc/sysctl.conf добавляется строка:

net.ipv4.conf.all.forwarding=1

При необходимости можно очистить существующие NAT правила:

iptables -t nat –flush

В iptables добавляется правило, например,

iptables -t nat -A POSTROUTING -s 192.168.99.0/24 -j SNAT --to-source 172.16.16.94, где

192.168.99.0/24 внутренняя сеть, а 172.16.16.94 адрес, через который нужен выход в Интернет, аналогично прописываются другие внутренние сети.

Если IP адрес на внешнем сетевом интерфейсе меняется (динамический), тогда вместо SNAT указывается MASQUERADE:

iptables -t nat -A POSTROUTING -s 192.168.99.0/24 -j MASQUERADE

Сохранение добавленных правил iptables.

nano /etc/rc.local

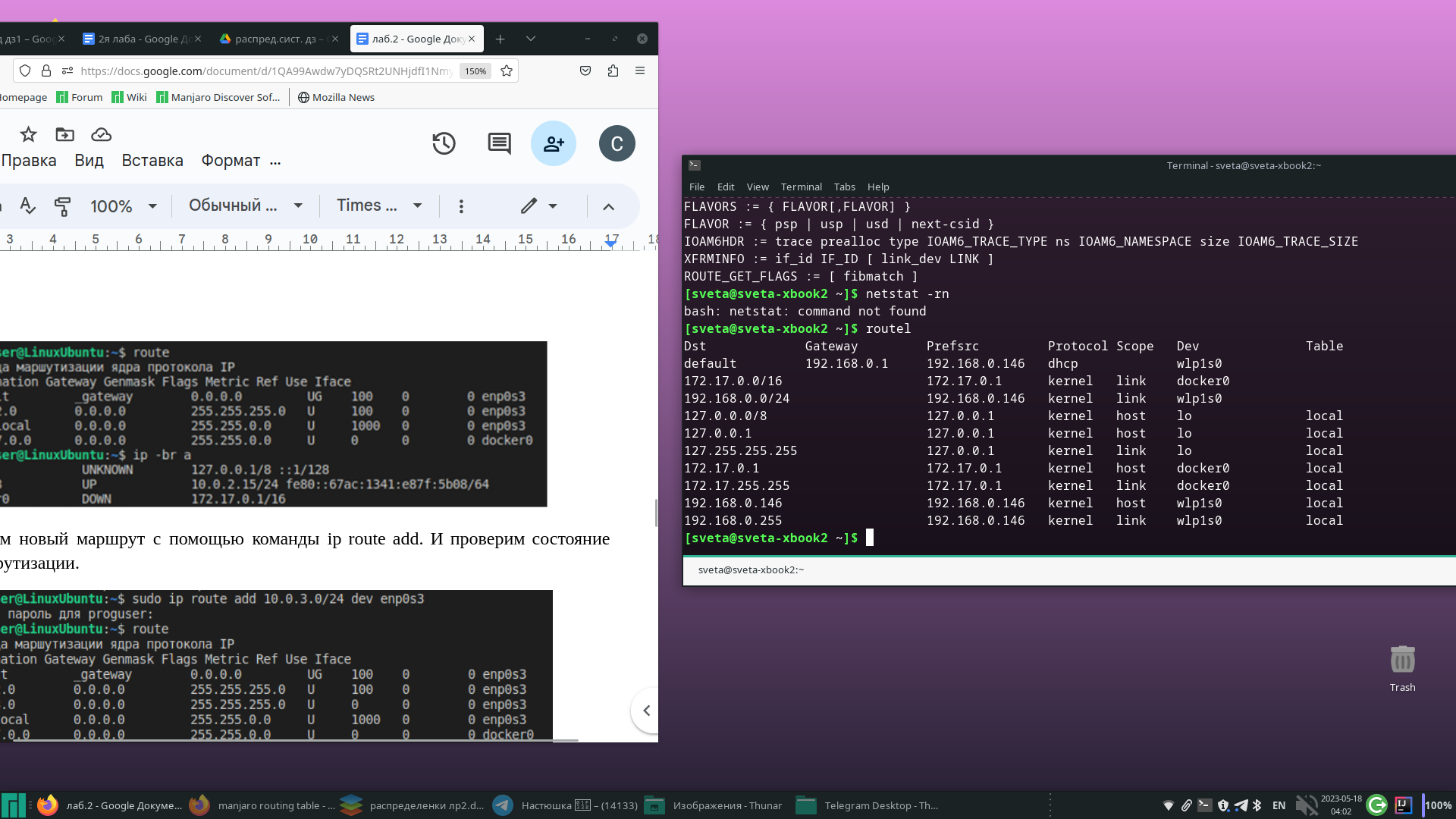
/sbin/iptables -t nat -A POSTROUTING -s 192.168.99.0/24 -o eth3 -j SNAT --to-source 172.16.90.1-172.16.90.5 –persistent

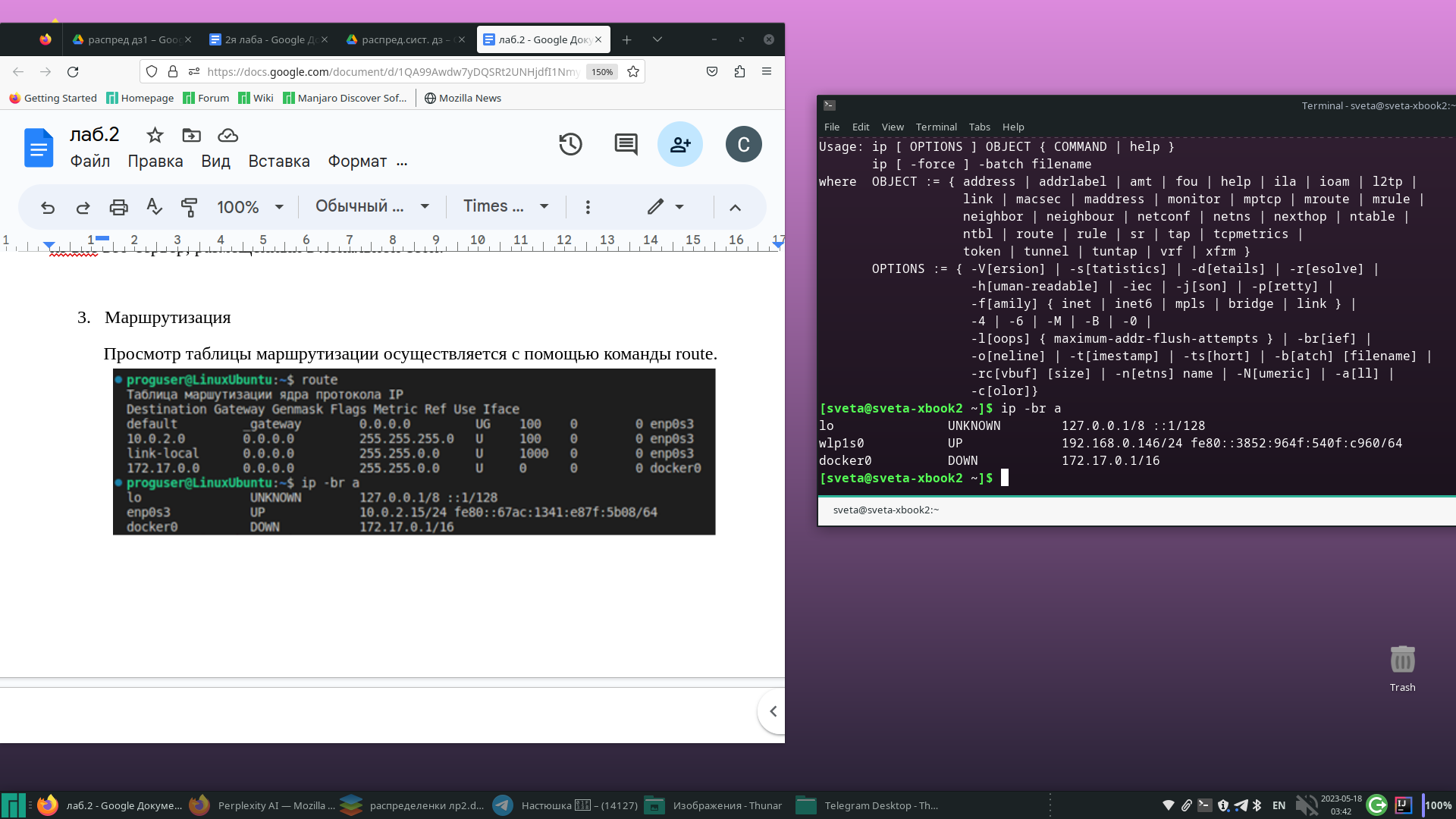
Цепочка SRCNAT (Source NAT) предназначена для трафика, условно названного «наружу». Изменяется адрес источника (source) на адрес внешнего интерфейса. Например, чтобы дать возможность клиентам локальной сети посещать сайты в интернет.

Цепочка DSTNAT (Destination NAT) определяет, как будет проходить «входящий» трафик. Роутер изменит адрес назначения (destination). Например, можно сделать доступным из вне веб-сервер, размещенный в локальной сети.

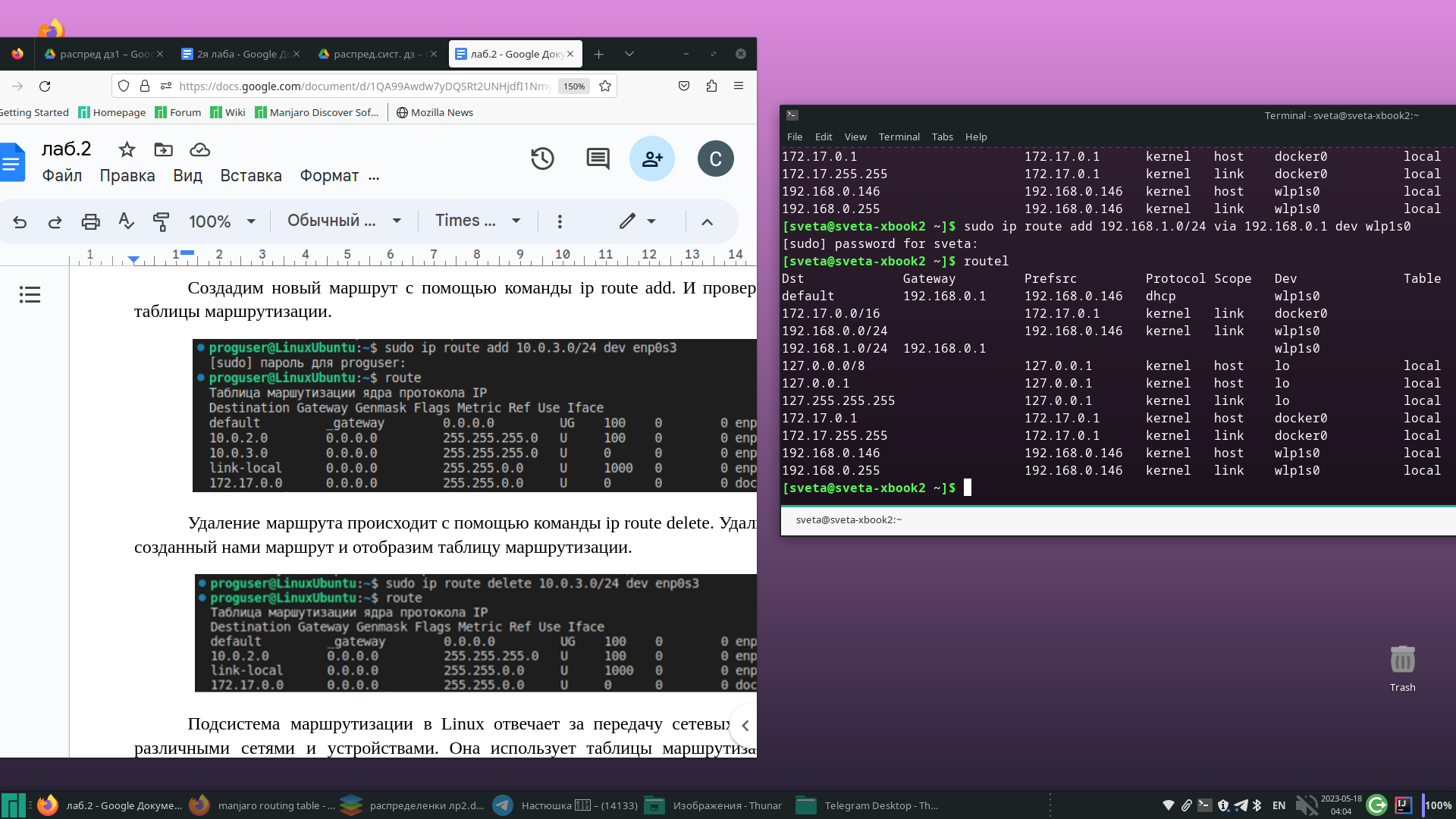
1. Маршрутизация

Просмотр таблицы маршрутизации осуществляется с помощью команды routel.

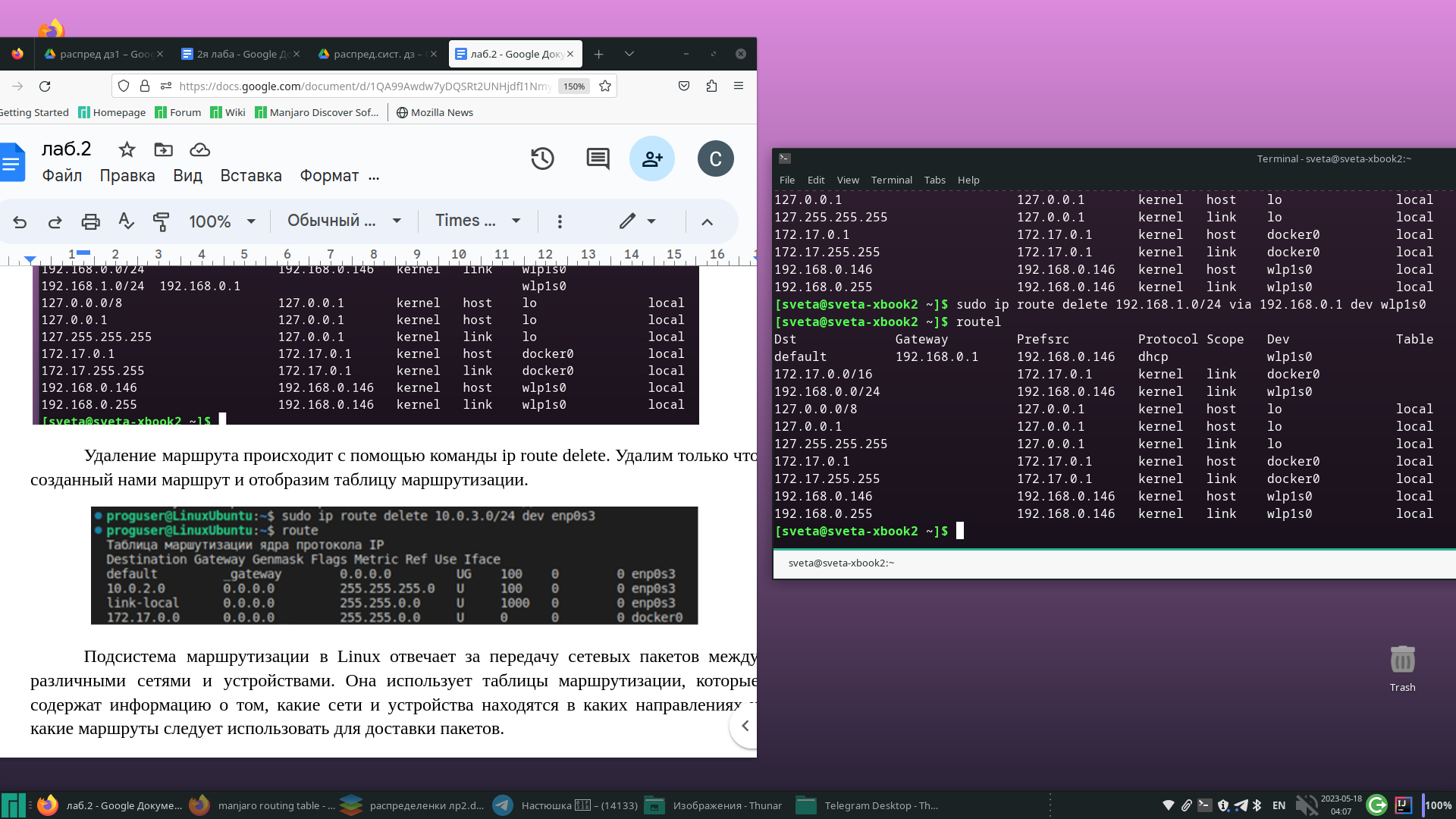




Создадим новый маршрут с помощью команды ip route add. И проверим состояние таблицы маршрутизации.



Удаление маршрута происходит с помощью команды ip route delete. Удалим только что созданный нами маршрут и отобразим таблицу маршрутизации.



Подсистема маршрутизации в Linux отвечает за передачу сетевых пакетов между различными сетями и устройствами. Она использует таблицы маршрутизации, которые содержат информацию о том, какие сети и устройства находятся в каких направлениях и какие маршруты следует использовать для доставки пакетов.

Когда пакет поступает на сетевой интерфейс, ядро Linux проверяет таблицы маршрутизации, чтобы определить, какой маршрут следует использовать для доставки пакета. Если в таблице маршрутизации нет соответствующей записи, ядро отправляет пакет на шлюз по умолчанию.

Для работы с подсистемой маршрутизации в Linux используются различные утилиты командной строки, такие как iproute2 и net-tools.

С помощью этих утилит можно просматривать таблицы маршрутизации, добавлять и удалять маршруты, настраивать параметры сетевых интерфейсов и т.д.

1. Проброс портов через ssh

Чтобы пробросить порт с сервера на локальную машину через ssh, необходимо выполнить следующую команду на локальном ПК.

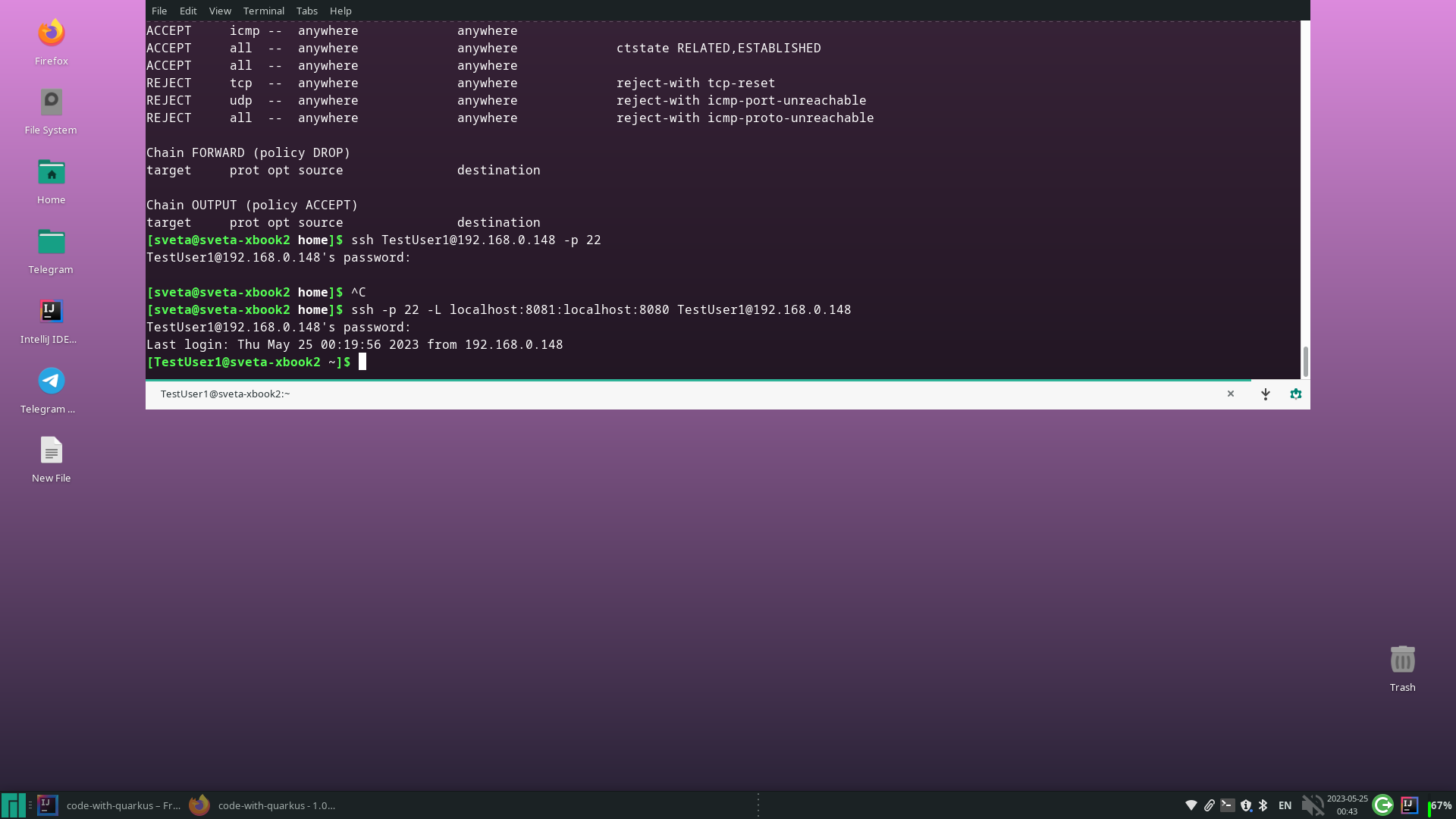
ssh -p 22 -L localhost:8081:localhost:8080 TestUser1@192.168.0.148, где

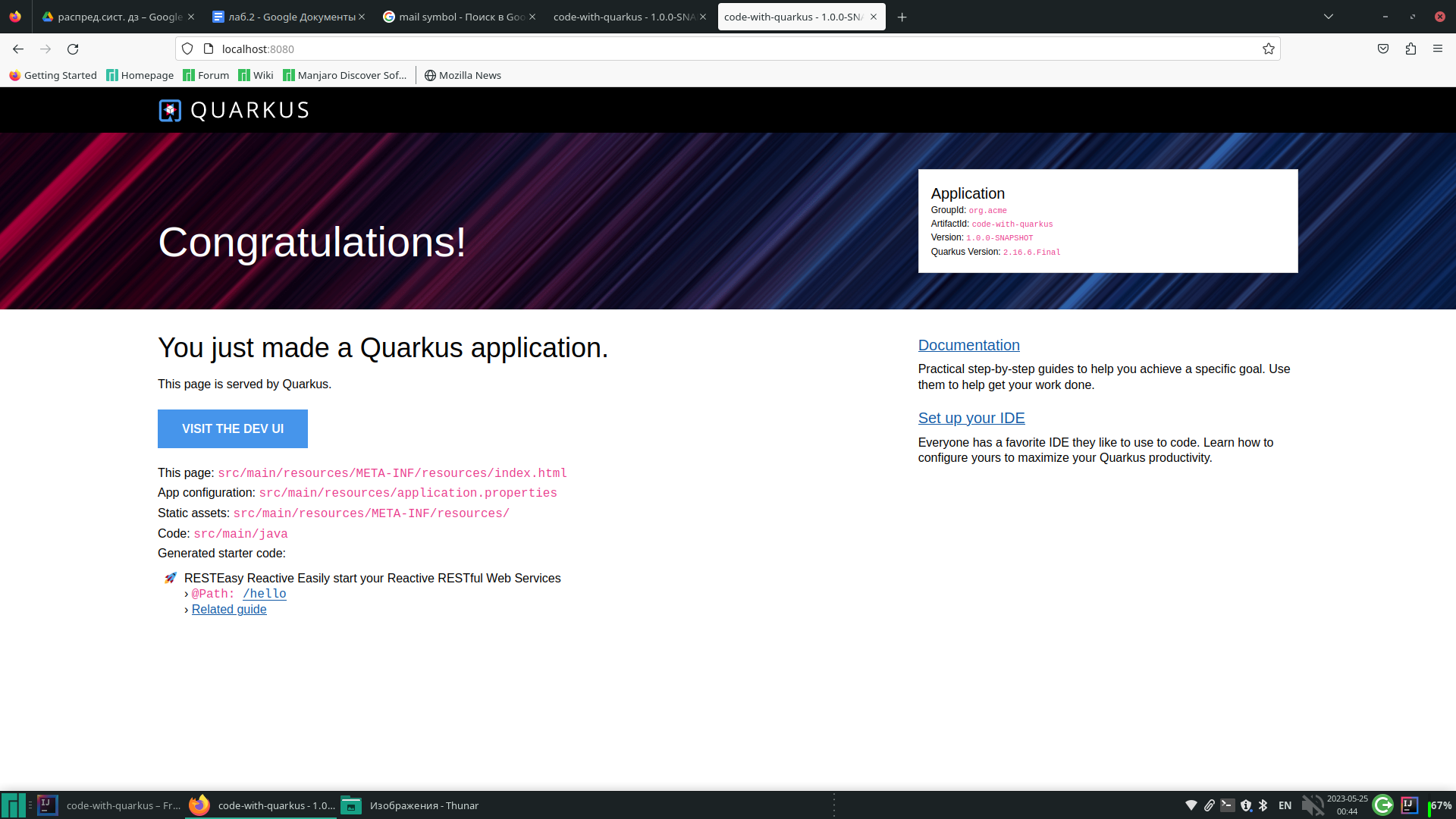
-p 22 – указание порта для подключения по ssh,

localhost:8080 – локальный адрес, на который перенаправляется web-приложение

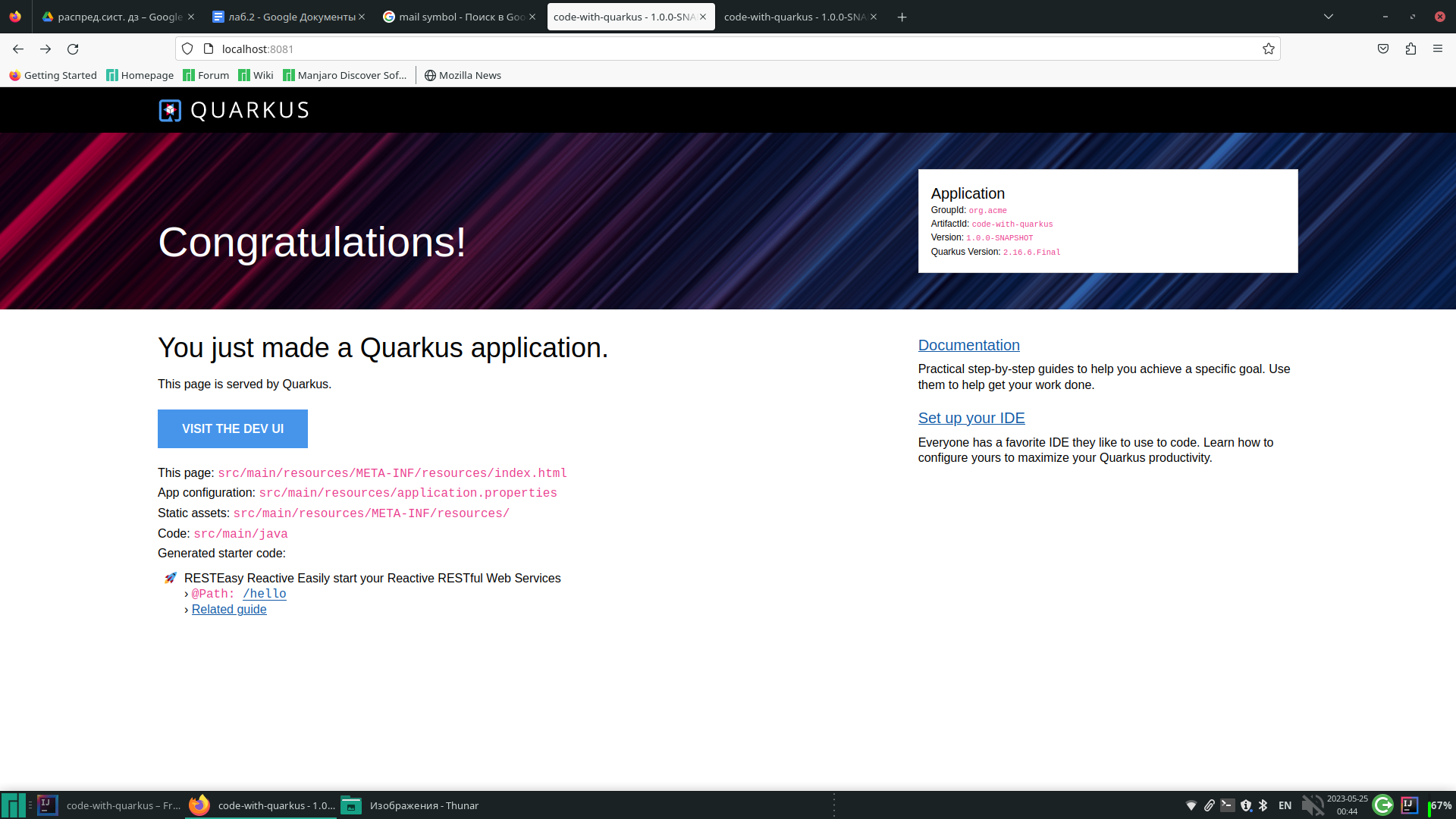
localhost:8081 – удаленный адрес, на котором запущено web-приложение

TestUser1@192.168.0.148 – удаленный сервер





Приложение, которое запущено на сервере на localhost:8080.



Приложение, перенаправлено на локальную машину на localhost:8081.

SSH – это протокол, использующий клиент-серверную модель для аутентификации удаленных систем и обеспечения шифрования данных, обмен которыми происходит в рамках удаленного доступа.

По умолчанию для работы протокола используется TCP-22 порт: на нем сервер (хост) ожидает входящее подключение и после получения команды и проведения аутентификации организует запуск клиента, открывая выбранную пользователем оболочку. При необходимости пользователь может изменять используемый порт.

Для создания подключения SSH клиент должен инициировать соединение с сервером, обеспечив защищенное соединение и подтвердив свой идентификатор (проверяются соответствие идентификатора с предыдущими записями, хранящимися в RSA-файле, и личные данные пользователя для аутентификации).

SSH клиент под Linux можно использовать не только для безопасного подключения к консоли других компьютеров. Эта программа имеет ряд полезных функций, одной из которых является создание защищенных туннелей и эмуляция SOCKS5 прокси на локальной машине.

С помощью SSH-туннелей можно соединиться с сервером и использовать его сервисы через проброс портов и сокетов, и, наоборот, можно пробросить на сервер ресурсы со своего локального ПК, или создать прокси сервер с динамическим портом. И всё это с теми же средствами защиты данных, какие присутствуют при стандартном соединении с сервером по SSH.