

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**



**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт

компьютерных наук

Кафедра

автоматизированных систем управления

**ЛАБОРАТОРНАЯ РАБОТА №6**

По дисциплине "Операционные системы Linux"

На тему "Работа с SSH"

Студент

ПИ-22-1

подпись, дата

Клименко Н.Д.

Руководитель

канд.техн.наук, доцент

ученая степень, ученое звание

подпись, дата

Кургасов В.В.

Липецк, 2024 г.

## **Оглавление**

<b>Цель работы.....</b>	<b>3</b>
<b>Ход работы.....</b>	<b>4</b>
<b>Часть I - Telnet.....</b>	<b>4</b>
<b>Часть II – SSH .....</b>	<b>9</b>
<b>Вывод.....</b>	<b>16</b>
<b>Контрольные вопросы .....</b>	<b>17</b>

## **Цель работы**

Практическое ознакомление с программным обеспечением удаленного доступа к распределенным системам обработки данных.

## Ход работы

### Часть I - Telnet

Перед началом работы было настроено сетевое соединение между хостовой и виртуальной машинами в VirtualBox посредством сетевого моста. Такой подход позволяет виртуальной машине взаимодействовать с хостовой системой так же, как если бы она была отдельным устройством в локальной сети.

С помощью команды `ip addr` был определен IP-адрес виртуальной машины. Его можно найти в строке интерфейса `enp0s3`, где указан адрес в формате `inet <IP-адрес>`. Для проверки доступности виртуальной машины из хостовой выполнена команда `ping <IP-адрес>`. Успешный результат выполнения команды, представленный на рисунке 1, подтверждает корректность сетевых настроек.

```
PS C:\Users\Nikita> ping 192.168.3.40

Обмен пакетами с 192.168.3.40 по с 32 байтами данных:
Ответ от 192.168.3.40: число байт=32 время<1мс TTL=64
Ответ от 192.168.3.40: число байт=32 время<1мс TTL=64
Ответ от 192.168.3.40: число байт=32 время<1мс TTL=64
Ответ от 192.168.3.40: число байт=32 время<1мс TTL=64

Статистика Ping для 192.168.3.40:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 0мсек, Максимальное = 0 мсек, Среднее = 0 мсек
PS C:\Users\Nikita> |
```

Рисунок 1 – Проверка соединения

Для начала обновим список доступных пакетов и установим необходимые программы. Это выполняется следующими командами:

- `sudo apt update`
- `sudo apt install openssh-server telnetd tcpdump screen`

Команда, приведенная выше производит установку ssh-сервера и дополнительных утилит, а именно:

- `openssh-server`: сервер OpenSSH для удаленного доступа;
- `telnetd`: демон Telnet для удаленного подключения;
- `tcpdump`: анализатор сетевого трафика;

- screen: утилита управления сеансами командой строки.

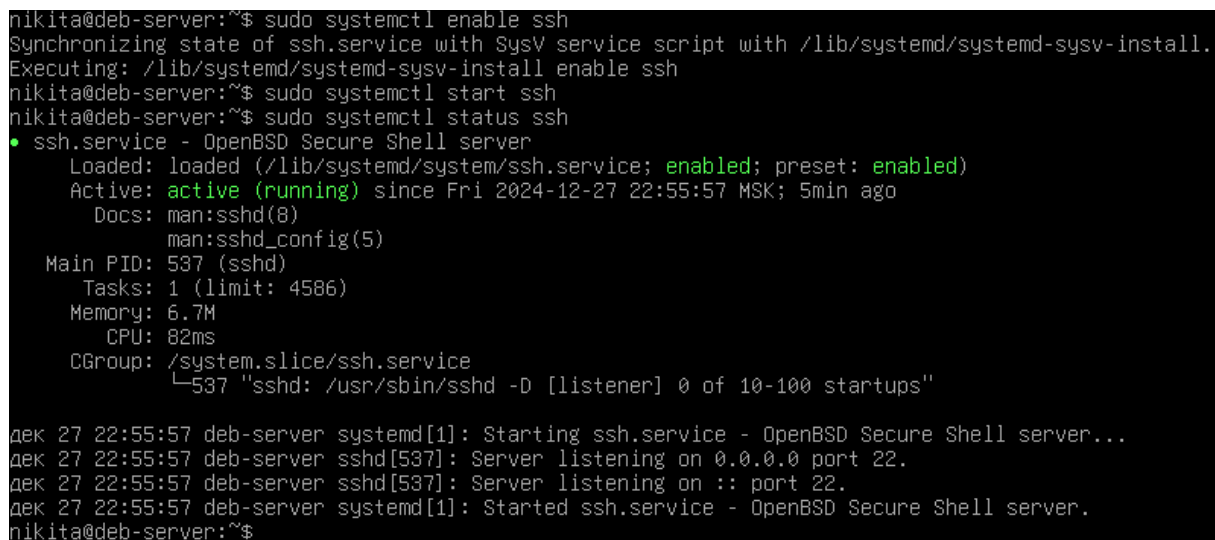
Активируем и запускаем ssh-сервер следующими командами:

- sudo systemctl enable ssh

- sudo systemctl start ssh

- sudo systemctl status ssh

На рисунке 2 представлен результат выполненных команд, ssh-сервер активен.



```
nikita@deb-server:~$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
nikita@deb-server:~$ sudo systemctl start ssh
nikita@deb-server:~$ sudo systemctl status ssh
• ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: active (running) since Fri 2024-12-27 22:55:57 MSK; 5min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 537 (sshd)
     Tasks: 1 (limit: 4586)
    Memory: 6.7M
       CPU: 82ms
   CGroup: /system.slice/ssh.service
           └─537 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

дек 27 22:55:57 deb-server systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
дек 27 22:55:57 deb-server sshd[537]: Server listening on 0.0.0.0 port 22.
дек 27 22:55:57 deb-server sshd[537]: Server listening on :: port 22.
дек 27 22:55:57 deb-server systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
nikita@deb-server:~$
```

Рисунок 2 – SSH-сервер

Для работы демона telnetd необходимо настроить суперсервер inetd, который отвечает за управление сетевыми службами. Проверим наличие inetd и установим его в случае отсутствия:

- sudo apt install openbsd-inetd -y

После установки отредактируем конфигурационный файл inetd. Для этого откроем файл командой:

- sudo nano /etc/inetd.conf

В файле необходимо раскомментировать строку:

- telnet stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.telnetd

Это действие позволяет активировать Telnet-сервис. Раскомментирование строки сообщает inetd, что при подключении клиента по протоколу Telnet нужно запускать in.telnetd через утилиту tcpd для дополнительной проверки. Эта проверка включает в себя контроль доступа на основе правил, заданных в файлах

/etc/hosts.allow и /etc/hosts.deny, а также логирование информации о подключениях, таких как IP-адрес клиента, время подключения и используемая служба. Такой подход повышает безопасность, позволяя ограничить доступ только для доверенных клиентов и отслеживать все попытки подключения.

```
#:INTERNAL: Internal services
#discard      stream  tcp6    nowait  root    internal
#discard      dgram   udp6    wait    root    internal
#daytime       stream  tcp6    nowait  root    internal
#time          stream  tcp6    nowait  root    internal

#:STANDARD: These are standard services.
telnet stream tcp    nowait  root    /usr/sbin/tcpd  /usr/sbin/telnetd

#:BSD: Shell, login, exec and talk are BSD protocols.
```

Рисунок 3 – Файл конфигурации inetd

Командой `sudo systemctl restart inetd` перезапустим суперсервер для применения внесенных изменений. Убедимся, что порт 23 активен для Telnet, данный результат представлен на рисунке 4. Из вывода команды `ss -lt` видно, что порт 23 (Telnet) прослушивается и готов к подключениям (0.0.0.0:telnet и [::]:telnet).

```
nikita@deb-server:~$ sudo systemctl restart inetd
nikita@deb-server:~$ ss -lt
```

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	Process
LISTEN	0	244	127.0.0.1:postgresql	0.0.0.0:*	
LISTEN	0	128	0.0.0.0:ssh	0.0.0.0:*	
LISTEN	0	128	0.0.0.0:telnet	0.0.0.0:*	
LISTEN	0	244	:::1:postgresql	:::*	
LISTEN	0	128	:::ssh	:::*	
LISTEN	0	128	:::telnet	:::*	

```
nikita@deb-server:~$
```

Рисунок 4 – Прослушиваемые порты

Для подключения к виртуальной машине через Telnet необходимо активировать клиент Telnet на хостовой системе. Это действие выполняется через панель управления Windows, где включается компонент "Клиент Telnet", действие представлено на рисунке 5.

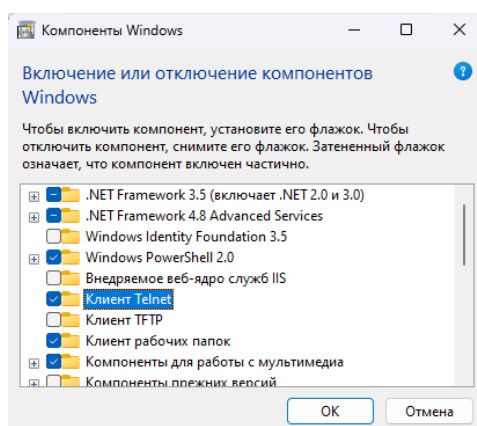


Рисунок 5 – Активация Telnet

На виртуальной машине включаем анализ сетевого трафика с фильтрацией, связанного с портом 23:

```
- sudo tcpdump -l -v -nn tcp and src port 23 or dst port 23 | tee telnet.log
```

### Разбор вышеприведенной команды:

- "-l": вывод в реальном времени;
- "-v": подробный формат вывода;
- "-nn": отключение разрешения имен хостов и служб;
- tcp: фильтрация только трафика TCP;
- src port 23 or dst port 23: анализ трафика с исходным или целевым портом

23;

- tee telnet.log: запись вывода команды в файл telnet.log.

На хостовой машине выполнена команда подключения через Telnet:

- telnet <IP-адрес виртуальной машины>

После успешного подключения была выполнена команда `uname -a`, результат данных действий представлен на рисунке 6.

[illegible]

### Рисунок 6 – Сессия telnet

После завершения сессии Telnet можно остановить анализатор сетевого трафика tcpdump на виртуальной машине и выполнить анализ собранных логов. Для фильтрации пакетов инициализации и завершения соединения применим команду:

- ```
- cat telnet.log | grep -P '\[[SF].*?\]' telnet.log
```

"-P" – использование синтаксиса регулярных выражений Perl.

"\[SF].\*?\]" – поиск пакетов с флагами [S] (установка соединения) и [F] (завершение соединения).

Результат предоставлен на рисунке 7.

```
nikita@deb-server:~$ cat telnet.log | grep -P '\[SF].*?\]'
192.168.3.12.57513 > 192.168.3.40.23: Flags [S], cksum 0x928d (correct), seq 1676285282, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
192.168.3.40.23 > 192.168.3.12.57513: Flags [S.], cksum 0x87ab (incorrect -> 0x8c1c), seq 1686544858, ack 1676285283, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
192.168.3.40.23 > 192.168.3.12.57513: Flags [F.], cksum 0x879f (incorrect -> 0xc1ea), seq 915, ack 103, win 502, length 0
192.168.3.12.57513 > 192.168.3.40.23: Flags [F.], cksum 0xb3de (correct), seq 103, ack 916, win 4097, length 0
nikita@deb-server:~$
```

Рисунок 7 – Пакеты инициализации и завершения сессии



## Часть II – SSH

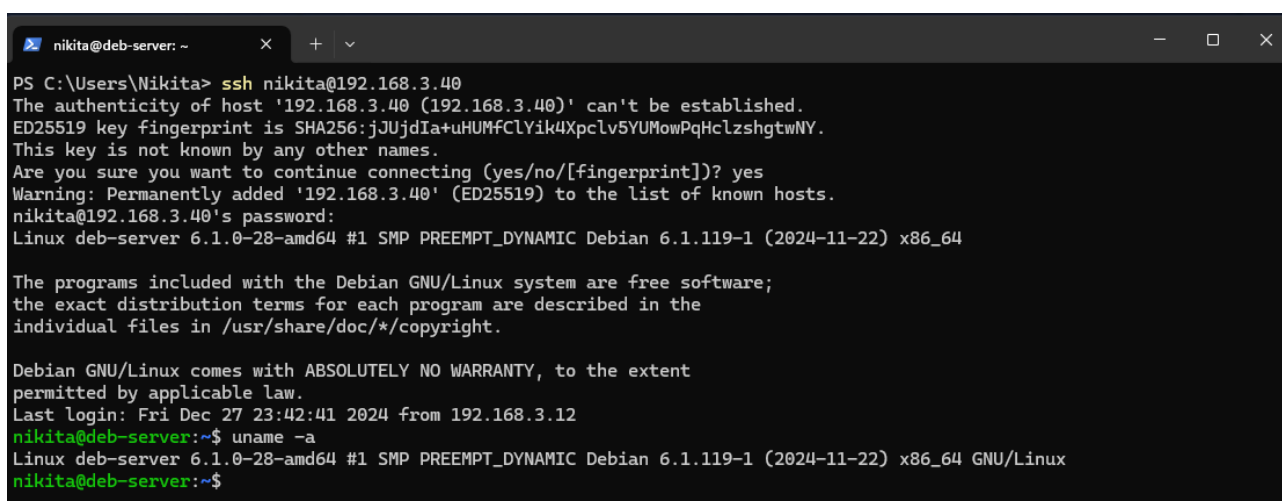
Для анализа сетевого трафика с прослушиванием порта 22 (используемого SSH) на виртуальной машине запустим tcpdump:

- `sudo tcpdump -l -v -nn tcp and src port 22 or dst port 22 | tee ssh.log`

На хостовой машине выполним подключение к виртуальной машине через ssh с помощью команды:

- `ssh <пользователь>@<IP-адрес виртуальной машины>`

После ввода пароля и успешного подключения выполнена команда `uname -a`. Данные действия представлены на рисунке 8.



```
PS C:\Users\Nikita> ssh nikita@192.168.3.40
The authenticity of host '192.168.3.40 (192.168.3.40)' can't be established.
ED25519 key fingerprint is SHA256:jjUjdIa+uHUMfCLYik4Xpclv5YUMowPqHclzshgtwNY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.3.40' (ED25519) to the list of known hosts.
nikita@192.168.3.40's password:
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) 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.

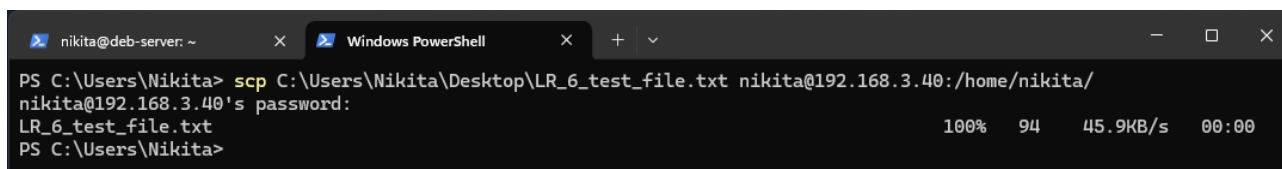
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Dec 27 23:42:41 2024 from 192.168.3.12
nikita@deb-server:~$ uname -a
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64 GNU/Linux
nikita@deb-server:~$
```

Рисунок 8 – Сессия ssh

Создаем текстовый файл на хостовой машине с названием LR\_6\_test\_file.txt, содержащий ФИО и номер лабораторной работы. Далее передаем его на виртуальную машину с использованием `scp`:

- `scp <путь к файлу на хостовой машине> <пользователь>@<IP-адрес виртуальной машины>:/home/<пользователь>/`

После ввода пароля файл успешно передан. Процесс передачи показан на рисунке 9. На рисунке 10 демонстрируется содержимое переданного файла, а также результаты команды `ls`, подтверждающие его наличие.



```
PS C:\Users\Nikita> scp C:\Users\Nikita\Desktop\LR_6_test_file.txt nikita@192.168.3.40:/home/nikita/
nikita@192.168.3.40's password:
LR_6_test_file.txt                               100% 94 45.9KB/s 00:00
PS C:\Users\Nikita>
```

Рисунок 9 – Передача файла

```
Windows PowerShell
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 Dec 27 23:42:41 2024 from 192.168.3.12
nikita@deb-server:~$ uname -a
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64 GNU/Linux
nikita@deb-server:~$ ls -l
итого 116
-rw-r--r-- 1 nikita nikita 58444 дек  9 18:56 composer-setup.php
drwxr-xr-x 18 nikita nikita 4096 дек 14 15:51 demo
drwxr-xr-x  5 nikita nikita 4096 дек 17 20:14 my-demo
-rw-r--r-- 1 nikita nikita 16954 дек 28 00:37 ssh.log
-rw-r--r-- 1 nikita nikita 25970 дек 27 23:42 telnet.log
nikita@deb-server:~$ ls -l
итого 136
-rw-r--r-- 1 nikita nikita 58444 дек  9 18:56 composer-setup.php
drwxr-xr-x 18 nikita nikita 4096 дек 14 15:51 demo
-rw-r--r-- 1 nikita nikita   94 дек 28 00:38 LR_6_test_file.txt
drwxr-xr-x  5 nikita nikita 4096 дек 17 20:14 my-demo
-rw-r--r-- 1 nikita nikita 35946 дек 28 00:38 ssh.log
-rw-r--r-- 1 nikita nikita 25970 дек 27 23:42 telnet.log
nikita@deb-server:~$ cat LR_6_test_file.txt
Клименко Никита Дмитриевич
Лабораторная работа №6
nikita@deb-server:~$ nano LR_6_test_file.txt
nikita@deb-server:~$ exit
выход
Connection to 192.168.3.40 closed.
PS C:\Users\Nikita>
```

Рисунок 10 – Просмотр переданного файла

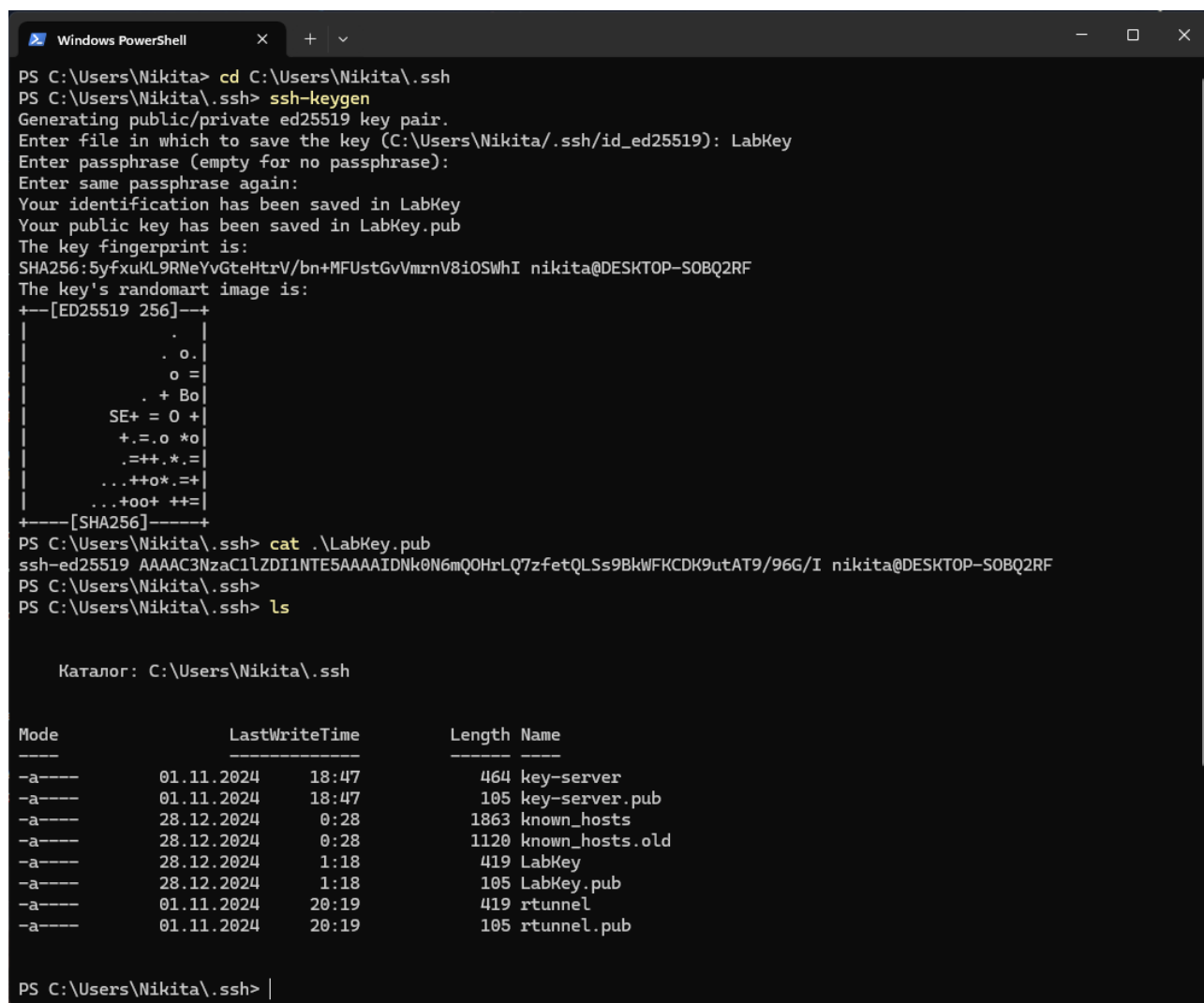
Для безопасного подключения без пароля настроим SSH-ключи. Перед созданием ключей проведем необходимые процедуры на сервере, а именно создадим директорию ".ssh", и файл "authorized\_keys", который будет хранить содержимое ключа. Также были изменены права доступа для каталога (700 – только владелец может читать, записывать и выполнять файлы в директории) и для файла (600 – только владелец может читать и записывать файл). Действия продемонстрированы на рисунке 11.

```
nikita@deb-server:~$ mkdir .ssh
nikita@deb-server:~$ chmod 700 /home/nikita/.ssh
nikita@deb-server:~$ touch /home/nikita/.ssh/authorized_keys
nikita@deb-server:~$ chmod 600 /home/nikita/.ssh/authorized_keys
nikita@deb-server:~$ _
```

Рисунок 11 - Манипуляции перед созданием ключа

Теперь на хостовой машине пропишем команду `ssh-keygen`, для создания пары ключей (публичный, который будет храниться на сервере, а по приватному будет осуществляться вход). По умолчанию используется шифрование ED25519 и хэш sha 256. В результате будет сформирована связка публичного и приватного ключей. Для ключа я дал имя "LabKey", пароль не задавал. С помощью команды

cat можно вывести содержимое публичного ключа, для дальнейшего сравнения, когда передадим его на сервер. Создание ключей показано на рисунке 12.



```
PS C:\Users\Nikita> cd C:\Users\Nikita\.ssh
PS C:\Users\Nikita\.ssh> ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\Nikita\.ssh/id_ed25519): LabKey
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in LabKey
Your public key has been saved in LabKey.pub
The key fingerprint is:
SHA256:5yfxuKL9RNeYvGteHtrV/bn+MFUstGvVmrnV8i0SWhI nikita@DESKTOP-SOBQ2RF
The key's randomart image is:
+--[ED25519 256]--+
|
| . o |
| o = |
| . + Bo |
| SE+ = 0 + |
| +. . o *o |
| . = + . * . = |
| ... + o * . = + |
| ... + oo + + + = |
+-----[SHA256]-----+
PS C:\Users\Nikita\.ssh> cat .\LabKey.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDNk0N6mQOHrLQ7zfetQLSs9BkwFKCDK9utAT9/96G/I nikita@DESKTOP-SOBQ2RF
PS C:\Users\Nikita\.ssh>
PS C:\Users\Nikita\.ssh> ls

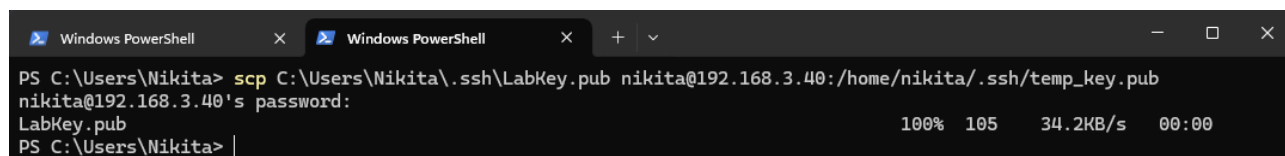
Каталог: C:\Users\Nikita\.ssh

Mode                LastWriteTime         Length Name
----                -
-a-----         01.11.2024    18:47           464 key-server
-a-----         01.11.2024    18:47           105 key-server.pub
-a-----         28.12.2024     0:28          1863 known_hosts
-a-----         28.12.2024     0:28          1120 known_hosts.old
-a-----         28.12.2024     1:18           419 LabKey
-a-----         28.12.2024     1:18           105 LabKey.pub
-a-----         01.11.2024    20:19           419 rtunnel
-a-----         01.11.2024    20:19           105 rtunnel.pub

PS C:\Users\Nikita\.ssh> |
```

Рисунок 12 – Создание ключей

Затем с помощью уже освоенной команды передачи scp передадим публичный ключ на сервер (в качестве временного ключа). Процесс передачи показан на рисунке 13.



```
PS C:\Users\Nikita> scp C:\Users\Nikita\.ssh\LabKey.pub nikita@192.168.3.40:/home/nikita/.ssh/temp_key.pub
nikita@192.168.3.40's password:
LabKey.pub
PS C:\Users\Nikita> |
```

Рисунок 13 – Передача ключа

На сервере переместим содержимое переданного публичного ключа (temp\_key.pub) в созданный файл authorized\_keys. Также можно вывести

содержимое и сравнить его с публичным ключом на хостовой машине, оно будет идентично.

```
nikita@deb-server:~$ mkdir .ssh
nikita@deb-server:~$ chmod 700 /home/nikita/.ssh
nikita@deb-server:~$ touch /home/nikita/.ssh/authorized_keys
nikita@deb-server:~$ chmod 600 /home/nikita/.ssh/authorized_keys
nikita@deb-server:~$ cd .ssh
nikita@deb-server:~/ssh$ ls -l
итого 4
-rw----- 1 nikita nikita  0 дек 28 01:12 authorized_keys
-rw-r--r-- 1 nikita nikita 105 дек 28 01:23 temp_key.pub
nikita@deb-server:~/ssh$ cat /home/nikita/.ssh/temp_key.pub >> /home/nikita/.ssh/authorized_keys
nikita@deb-server:~/ssh$ rm /home/nikita/.ssh/temp_key.pub
nikita@deb-server:~/ssh$ ls -l
итого 4
-rw----- 1 nikita nikita 105 дек 28 01:24 authorized_keys
nikita@deb-server:~/ssh$ cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDNk0N6mQ0HrLQ7zfetQLSs9BkWFKCDK9utAT9/96G/I nikita@DESKTOP-S0BQ2RF
nikita@deb-server:~/ssh$
```

Рисунок 14 – Просмотр публичного ключа на сервере

Теперь выполняем подключение через SSH с явным указанием приватного ключа (так как у меня имеется несколько пар ключей на хостовой машине):

- ssh <пользователь>@<IP-адрес виртуальной машины> -i <путь к ключу>

Процесс входа, показанный на рисунке 15, подтверждает, что пароль больше не требуется.

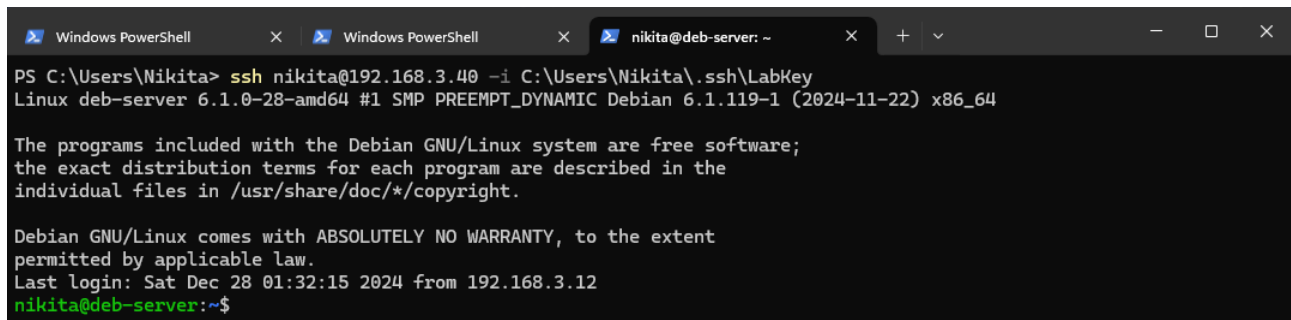


Рисунок 15 – Сессия ssh через ключ

Снова попробуем передать созданный файл, изменив его имя и содержимое. Процесс передачи осуществляется той же командой, но также требуется явно указать приватный ключ, чтобы авторизация происходила по ключу, а не по паролю.

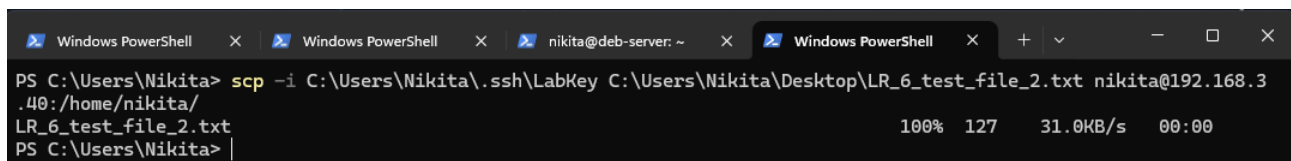


Рисунок 16 – Передача файла по ключу

На рисунке 17 продемонстрировано содержимое переданного файла.

```
Windows PowerShell X Windows PowerShell X nikita@deb-server: ~ X Windows PowerShell X + - □ X
PS C:\Users\Nikita> ssh nikita@192.168.3.40 -i C:\Users\Nikita\.ssh\LabKey
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) 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: Sat Dec 28 01:32:15 2024 from 192.168.3.12
nikita@deb-server:~$ ls -l
итого 116
-rw-r--r-- 1 nikita nikita 58444 дек 9 18:56 composer-setup.php
drwxr-xr-x 18 nikita nikita 4096 дек 14 15:51 demo
-rw-r--r-- 1 nikita nikita 94 дек 28 00:38 LR_6_test_file.txt
drwxr-xr-x 5 nikita nikita 4096 дек 17 20:14 my-demo
-rw-r--r-- 1 nikita nikita 14809 дек 28 01:34 ssh.log
-rw-r--r-- 1 nikita nikita 25970 дек 27 23:42 telnet.log
nikita@deb-server:~$ ls -l
итого 136
-rw-r--r-- 1 nikita nikita 58444 дек 9 18:56 composer-setup.php
drwxr-xr-x 18 nikita nikita 4096 дек 14 15:51 demo
-rw-r--r-- 1 nikita nikita 127 дек 28 01:35 LR_6_test_file_2.txt
-rw-r--r-- 1 nikita nikita 94 дек 28 00:38 LR_6_test_file.txt
drwxr-xr-x 5 nikita nikita 4096 дек 17 20:14 my-demo
-rw-r--r-- 1 nikita nikita 32497 дек 28 01:36 ssh.log
-rw-r--r-- 1 nikita nikita 25970 дек 27 23:42 telnet.log
nikita@deb-server:~$ cat LR_6_test_file_2.txt
Клименко Никита Дмитриевич
Лабораторная работа №6
Тест передачи - 2
nikita@deb-server:~$ |
```

Рисунок 17 – Содержимое переданного файла

Останавливаем анализатор сетевого трафика tcpdump и смотрим файл ssh.log.

```
nikita@deb-server:~$ cat ssh.log | grep -P '\[[SF].*\]'
192.168.3.12.58008 > 192.168.3.40.22: Flags [S], cksum 0x2fcc (correct), seq 1749053889, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
192.168.3.40.22 > 192.168.3.12.58008: Flags [S.], cksum 0x87ab (incorrect -> 0x9e26), seq 1029329979, ack 1749053890, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
192.168.3.12.58040 > 192.168.3.40.22: Flags [S], cksum 0x6a2b (correct), seq 1503217159, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
192.168.3.40.22 > 192.168.3.12.58040: Flags [S.], cksum 0x87ab (incorrect -> 0x88f5), seq 2410303867, ack 1503217160, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
192.168.3.12.58056 > 192.168.3.40.22: Flags [S], cksum 0x15d6 (correct), seq 1675857922, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
192.168.3.40.22 > 192.168.3.12.58056: Flags [S.], cksum 0x87ab (incorrect -> 0xd0f5), seq 3681380706, ack 1675857923, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
192.168.3.12.58059 > 192.168.3.40.22: Flags [S], cksum 0xc0bd (correct), seq 2642848628, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
192.168.3.40.22 > 192.168.3.12.58059: Flags [S.], cksum 0x87ab (incorrect -> 0x2d55), seq 3412445170, ack 2642848629, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
192.168.3.40.22 > 192.168.3.12.58059: Flags [F.], cksum 0x879f (incorrect -> 0x4ba1), seq 3829, ack 3214, win 501, length 0
192.168.3.12.58059 > 192.168.3.40.22: Flags [F.], cksum 0x4b96 (correct), seq 3214, ack 3830, win 511, length 0
nikita@deb-server:~$ _
```

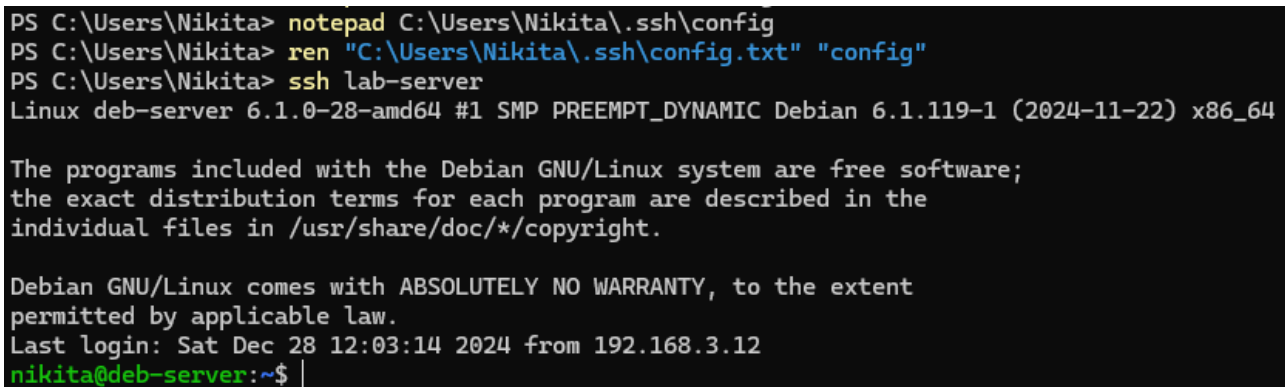
Рисунок 18 – Пакеты инициализации и завершения сессии

Процесс инициализации и завершения TCP соединения состоит из отправки соответствующего пакета и получения ответа сервера.

Чтобы избежать постоянного указания пути к ключу, создаем файл конфигурации ~/.ssh/config на хостовой машине со следующим содержимым:

```
Host lab-server
  HostName 192.168.3.40
  User nikita
  Port 22
  IdentityFile C:\Users\Nikita\.ssh\LabKey
```

Сохраняем файл в кодировке UTF-8 и переименовываем, убрав расширение .txt. На рисунке 19 показан успешный вход на сервер по ключу, без лишних параметров.



```
PS C:\Users\Nikita> notepad C:\Users\Nikita\.ssh\config
PS C:\Users\Nikita> ren "C:\Users\Nikita\.ssh\config.txt" "config"
PS C:\Users\Nikita> ssh lab-server
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) 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: Sat Dec 28 12:03:14 2024 from 192.168.3.12
nikita@deb-server:~$ |
```

Рисунок 19 – Сессия ssh

Для повышения безопасности отключим вход по паролю и запретим доступ для root. На виртуальной машине отредактируем файл конфигурации SSH:

- `sudo nano /etc/ssh/sshd_config`

Раскомментируем и изменим параметры:

- `PermitRootLogin no` (запрет входа для root)
- `PubkeyAuthentication yes` (включение авторизации по ключам)
- `PasswordAuthentication no` (отключение авторизации по паролю)
- `PermitEmptyPassword no` (запрет входа без пароля)

Сохраняем изменения и перезапускаем службу с системой:

- ssh командой `sudo systemctl restart ssh`
- `sudo reboot`

Пробуем войти на хостовой машине командой `ssh <пользователь>@<IP-адрес виртуальной машины>`, чтобы сервер запросил пароль, но как видно на рисунке 20 ошибку `Permisin denied (publickey)`, подтверждающую невозможность входа по паролю. Пробуем войти по ключу – все успешно.

```
nikita@deb-server: ~  
PS C:\Users\Nikita> ssh nikita@192.168.3.40  
nikita@192.168.3.40: Permission denied (publickey).  
PS C:\Users\Nikita> ssh lab-server  
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) 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: Sat Dec 28 12:19:55 2024  
nikita@deb-server:~$ uname -a  
Linux deb-server 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64  
GNU/Linux  
nikita@deb-server:~$ |
```

Рисунок 20 – Отключение авторизации по паролю

## **Вывод**

В ходе выполнения данной лабораторной работы приобрел навыки работы с программным обеспечением удаленного доступа к распределенным системам обработки данных. Изучил способы авторизации пользователя на удаленном сервере и настройку сервера для приема соединений с помощью программного обеспечения Telnet и SSH.



## **Контрольные вопросы**

### **1. Определите основные цели и задачи решаемые с помощью ПО удаленного доступа?**

Предоставление возможности администрирования, управления и технической поддержки серверов и сетевых устройств без физического доступа к ним. Такие решения позволяют пользователям взаимодействовать с удаленными системами для выполнения рабочих задач, разработки программного обеспечения, управления файлами и данными, а также устранения неисправностей. Удаленный доступ также упрощает работу в распределенных командах и способствует экономии времени и ресурсов.

### **2. Выделите отличительные особенности между режимами работы удаленного доступа по протоколам Telnet и SSH?**

Telnet передает данные в открытом виде, что делает его уязвимым к перехвату и использованию злоумышленниками. Подключение осуществляется через порт 23, а для аутентификации используется логин и пароль, которые передаются без шифрования.

SSH обеспечивает защищенное соединение за счет асимметричного шифрования. Все данные, включая процесс аутентификации, шифруются, исключая возможность перехвата. SSH использует порт 22 и позволяет применять дополнительные методы аутентификации, например, по ключу.

### **3. Опишите способы установления соединения при использовании протокола SSH? Охарактеризуйте положительные и отрицательные аспекты приведенных методов**

- Аутентификация по паролю: пользователь вводит имя и пароль для подключения, после чего создается защищенный канал связи. Из плюсов: простота настройки и использования, доступность. В тоже время из минусов: подверженность атакам на пароль, необходимость периодической смены паролей.

- Аутентификация по ключу. Клиент генерирует пару ключей (приватный и публичный), публичный ключ передается серверу для авторизации. Если клиент предоставляет соответствующий приватный ключ, соединение устанавливается.

Из плюсов: высокий уровень безопасности, отсутствие необходимости передачи пароля. В тоже время из минусов: сложность настройки и ответственность пользователя за сохранность ключей.

**4. Основываясь на заданиях лабораторной работы, приведите практический пример использования систем удаленного доступа?**

Примером может служить подключение к виртуальной машине для выполнения системных команд, передачи файлов или настройки сервисов. В лабораторной работе SSH использовался для подключения к Debian, выполнения команд управления системой и передачи текстовых файлов через зашифрованный канал с помощью утилиты scp. Это демонстрирует, как удаленный доступ упрощает взаимодействие с системами без физического присутствия.

**5. Перечислите распространенные сетевые службы, основанные на использовании шифрованного соединения по протоколу SSH? Приведите пример использования службы передачи файлов по безопасному туннелю?**

OpenSSH реализует протокол SSH, содержит утилиты

- ssh;
- scp (односторонняя передача данных);
- sftp (двусторонний безопасный канал).

Пример использования: `scp file.txt user@192.168.3.40:/home/user/`