

Лабораторная работа 1

Подготовка лабораторного стенда

Лушин Артём Андреевич

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	19
4	Контрольные вопросы	20

Список иллюстраций

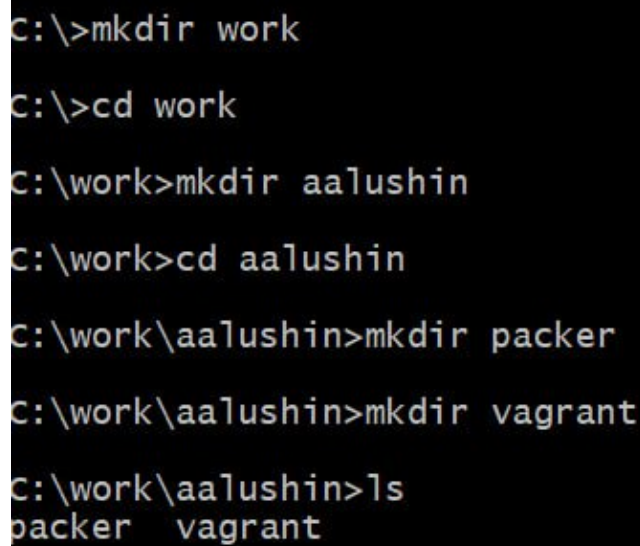
2.1	Создание каталога для проекта	5
2.2	Размещение образа	6
2.3	Размещение файлов в packer	6
2.4	Размещение файлов в vagrant	7
2.5	Размещение полкаталогов в vagrant	7
2.6	01-dummy.sh	7
2.7	01-user.sh	8
2.8	01-hostname.sh	8
2.9	02-forward.sh	9
2.10	02-routing.sh	9
2.11	vagrant-rocky.pkr.hcl	10
2.12	ks.cfg	10
2.13	Vagrantfile	11
2.14	Makefile в packer	12
2.15	Makefile в vagrant	13
2.16	Регистрация образа виртуальной машины	13
2.17	Запуск Server	14
2.18	Запуск client	14
2.19	Сверка машин	15
2.20	Подключение к серверу	15
2.21	Подключение к клиенту	16
2.22	Выключение машин	16
2.23	Запись в Vagrantfile	17
2.24	Фиксация машин	17
2.25	Проверка работоспособности Сервер	18
2.26	Проверка работоспособности Клиент	18

1 Цель работы

Целью данной работы является приобретение практических навыков установки Rocky Linux на виртуальную машину с помощью инструмента Vagrant.

2 Выполнение лабораторной работы

- 1) создал каталог для проекта. Расположил согласно инструкции. Сделал два подкаталога packer и vagrant. Работаю на системе ОС Windows.



```
C:\>mkdir work  
C:\>cd work  
C:\work>mkdir aalushin  
C:\work>cd aalushin  
C:\work\aalushin>mkdir packer  
C:\work\aalushin>mkdir vagrant  
C:\work\aalushin>ls  
packer  vagrant
```

Рис. 2.1: Создание каталога для проекта

- 2) В подкаталоге packer разместил образ виртуальной операционной системы Rocky linux.

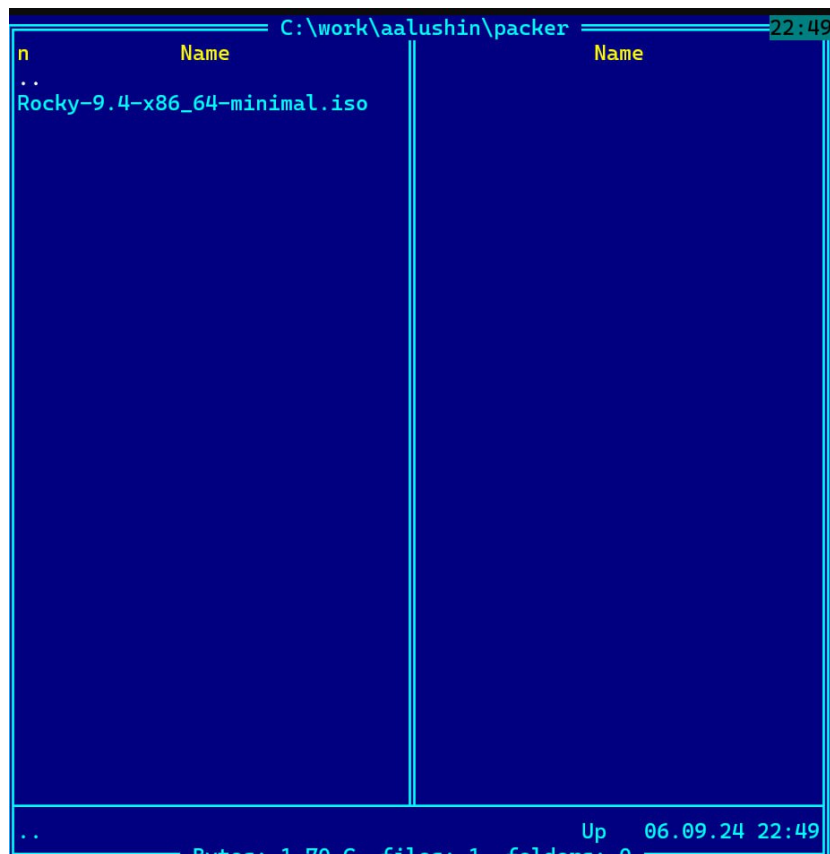


Рис. 2.2: Размещение образа

3) Разместил необходимые подкаталоги и файлы в папках packer и vagrant.

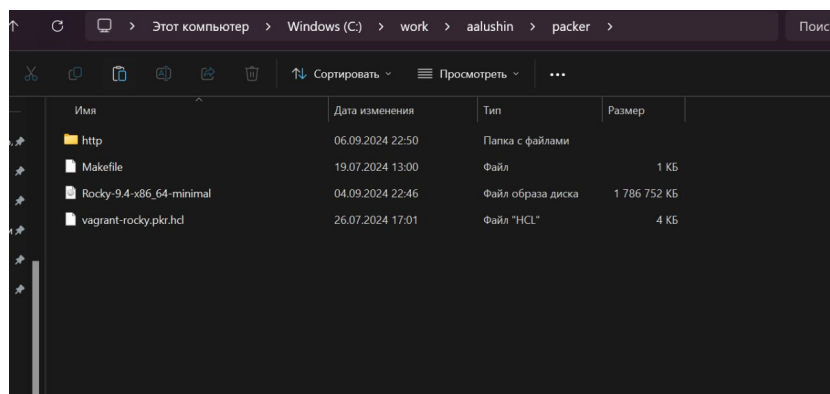


Рис. 2.3: Размещение файлов в packer

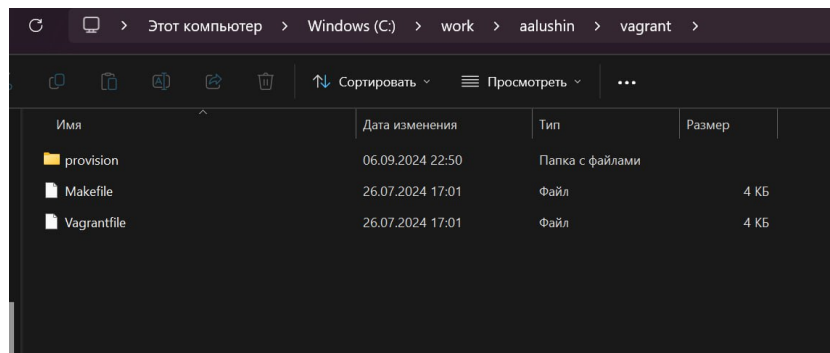


Рис. 2.4: Размещение файлов в vagrant

- 4) В каталоге vagrant создал ещё 3 подкаталога для работы: server, client, default.

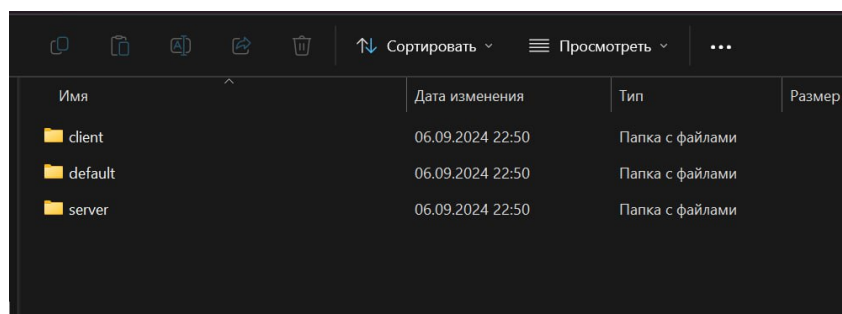


Рис. 2.5: Размещение подкаталогов в vagrant

- 5) В папках server, client, default разместил подготовленный файл 01-dummy.sh

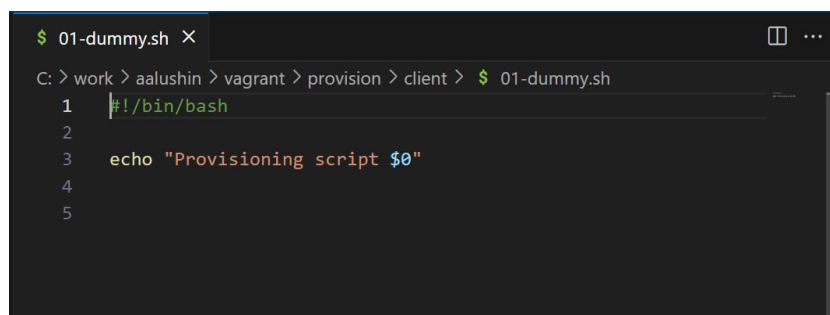


Рис. 2.6: 01-dummy.sh

6) В каталоге default разместил скрипт 01-user.sh. Заменил user на aalushin.

```
C: > work > aalushin > vagrant > provision > default > $ 01-user.sh
1  #!/bin/bash
2
3  echo "Provisioning script $0"
4
5  username=aalushin
6  userpassword=123456
7
8  encpassword=`openssl passwd -1 ${userpassword}`
9
10 id -u $username
11 if [[ $? ]]
12 then
13     adduser -G wheel -p ${encpassword} ${username}
14     homedir=`getent passwd ${username} | cut -d: -f6`
15     echo "export PS1='\u@\H \W]\\$ '" >> ${homedir}/.bashrc
16 fi
17
18
19
```

Рис. 2.7: 01-user.sh

7) В каталоге default разместил скрипт 01-hostname.sh. Заменил user на aalushin.

```
C: > work > aalushin > vagrant > provision > default > $ 01-hostname.sh
1  #!/bin/bash
2
3  username=user
4
5  hostnamectl set-hostname "${HOSTNAME%%.*}.${aalushin}.net"
6
7
8
```

Рис. 2.8: 01-hostname.sh

8) В каталоге server разместил скрипт 02-forward.sh.


```

$ 02-forward.sh X
C: > work > aalushin > vagrant > provision > server > $ 02-forward.sh
1  #!/bin/bash
2
3  echo "Provisioning script $0"
4
5  echo "Enable forwarding"
6  echo "net.ipv4.ip_forward = 1" > /etc/sysctl.d/90-forward.conf
7  sysctl -w net.ipv4.ip_forward=1
8
9  echo "Configure masquerading"
10 firewall-cmd --add-masquerade --permanent
11 firewall-cmd --reload
12
13 restorecon -vR /etc
14

```

Рис. 2.9: 02-forward.sh

9) В каталоге client разместил скрипт 02-routing.sh.

```

C: > work > aalushin > vagrant > provision > client > $ 01-routing.sh
1  #!/bin/bash
2
3  echo "Provisioning script $0"
4
5  nmcli connection modify "System eth1" ipv4.gateway "192.168.1.1"
6  nmcli connection up "System eth1"
7
8  nmcli connection modify eth0 ipv4.never-default true
9  nmcli connection modify eth0 ipv6.never-default true
10
11 nmcli connection down eth0
12 nmcli connection up eth0
13
14 # systemctl restart NetworkManager
15

```

Рис. 2.10: 02-routing.sh

10) Сверил содержимое файла vagrant-rocky.pkr.hcl. Содержимое совпадает с задачей лабораторной работы.

```

C: > work > aalushin > vagrant > Vagrantfile
1  # -*- mode: ruby -*-
2  # vi: set ft=ruby :
3
4  Vagrant.configure("2") do |config|
5
6      ## Common configuration
7      config.vm.provision "common dummy",
8          type: "shell",
9          preserve_order: true,
10         path: "provision/default/01-dummy.sh"
11
12      config.vm.provision "common hostname",
13          type: "shell",
14          preserve_order: true,
15          run: "always",
16          path: "provision/default/01-hostname.sh"
17
18      config.vm.provision "common user",
19          type: "shell",
20          preserve_order: true,

```

Рис. 2.11: vagrant-rocky.pkr.hcl

- 11) Сверил содержимое файла ks.cfg. Содержимое совпадает с задачей лабораторной работы.

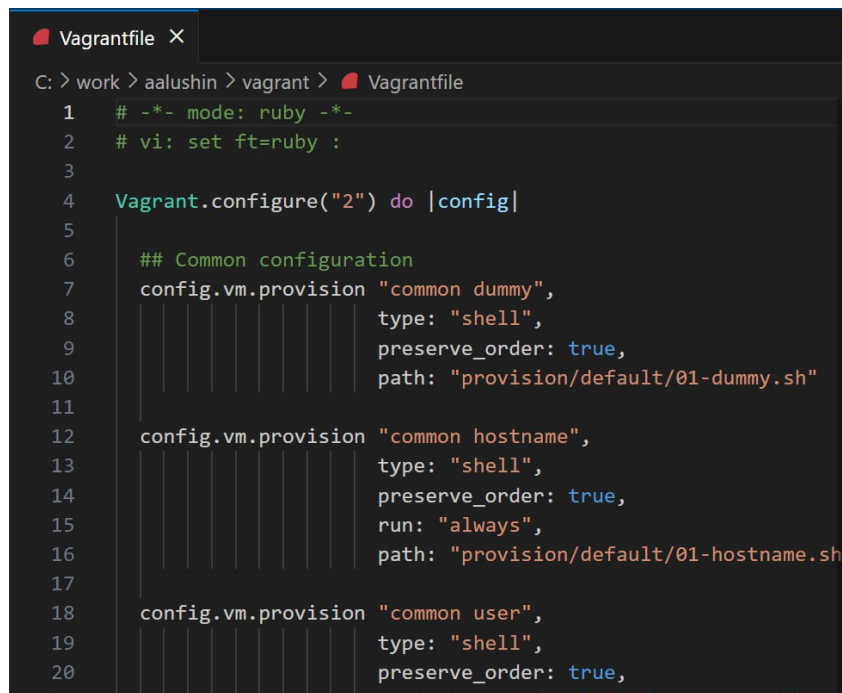
```

C: > work > aalushin > packer > http > ks.cfg
1  # System bootloader configuration
2  bootloader --append="no_timer_check console=tty0 console=ttyS"
3  # Clear the Master Boot Record
4  zerombr
5  # Partition clearing information
6  clearpart --all
7  # Reboot after installation
8  reboot
9  # Use text mode install
10 text
11 # Keyboard layouts
12 keyboard --vckeymap=us,ru --xlayouts='us,ru'
13 # System language
14 lang en_US.UTF-8
15
16 # Network information
17 network --bootproto=dhcp --device=link --activate
18
19 # System authorization information

```

Рис. 2.12: ks.cfg

- 12) Сверил содержимое файла Vagrantfile. Содержимое совпадает с задачей лабораторной работы.

A screenshot of a text editor window titled 'Vagrantfile'. The editor shows the content of a Vagrantfile with line numbers 1 through 20 on the left. The code is as follows:

```
1  # -*- mode: ruby -*-
2  # vi: set ft=ruby :
3
4  Vagrant.configure("2") do |config|
5
6      ## Common configuration
7      config.vm.provision "common dummy",
8                          type: "shell",
9                          preserve_order: true,
10                         path: "provision/default/01-dummy.sh"
11
12      config.vm.provision "common hostname",
13                          type: "shell",
14                          preserve_order: true,
15                          run: "always",
16                          path: "provision/default/01-hostname.sh"
17
18      config.vm.provision "common user",
19                          type: "shell",
20                          preserve_order: true,
```

Рис. 2.13: Vagrantfile

- 13) Сверил содержимое файла Makefile в каталоге racker. Содержимое совпадает с задачей лабораторной работы.

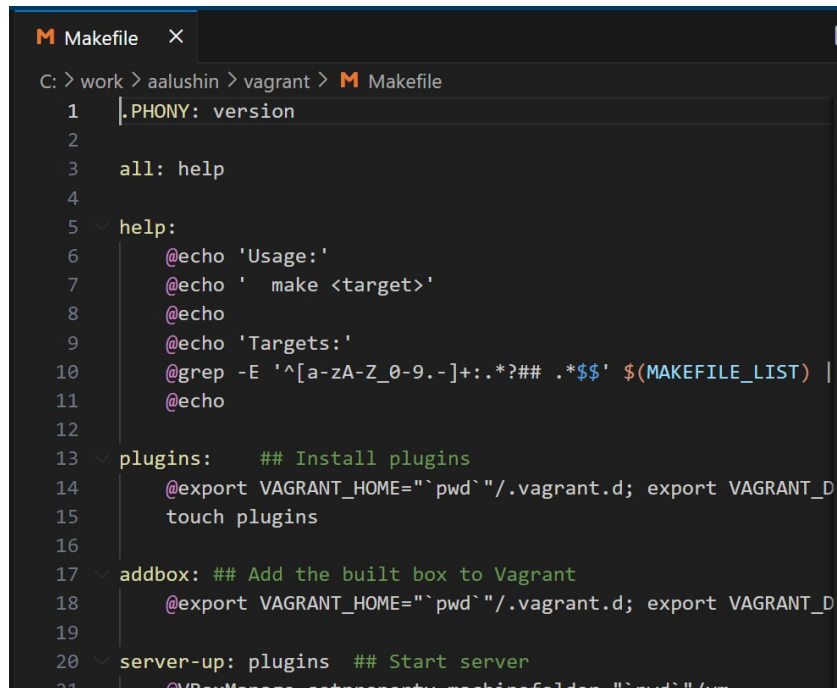
```

C: > work > aalushin > packer > M Makefile
1  .PHONY: version
2
3  all: help
4
5  init: ## Install missing plugins for packer
6      @mkdir -p "`pwd`"/.config/packer/plugins"
7      @export PACKER_CONFIG_DIR="`pwd`"/.config/packer"; export
8
9  box:    init    ## Build box for Rocky Linux
10      -@VBoxManage setproperty language C
11      @VBoxManage setproperty machinefolder "`pwd`"/vm
12      @export TMPDIR="`pwd`"; export PACKER_CONFIG_DIR="`pwd`
13      @VBoxManage setproperty machinefolder default
14
15  help:
16      @echo 'Usage:'
17      @echo '  make <target>'
18      @echo
19      @echo 'Targets:'
20      @grep -E '^[a-zA-Z_0-9.-]+:.*?## .*$$' $(MAKEFILE_LIST) |
21      @echo
22

```

Рис. 2.14: Makefile в packer

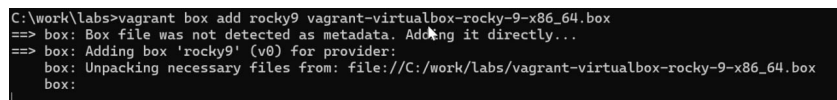
- 14) Сверил содержимое файла Makefile в каталоге vagrant. Содержимое совпадает с задачей лабораторной работы.



```
1 |.PHONY: version
2
3 all: help
4
5 help:
6     @echo 'Usage:'
7     @echo '  make <target>'
8     @echo
9     @echo 'Targets:'
10    @grep -E '^[a-zA-Z_0-9.-]+:.*?## .*$$' $(MAKEFILE_LIST) |
11    @echo
12
13 plugins:    ## Install plugins
14     @export VAGRANT_HOME=`pwd`"/.vagrant.d; export VAGRANT_D
15     touch plugins
16
17 addbox: ## Add the built box to Vagrant
18     @export VAGRANT_HOME=`pwd`"/.vagrant.d; export VAGRANT_D
19
20 server-up: plugins ## Start server
21     @VBoxManage setproperty machinefolden "`pwd`"/vm
```

Рис. 2.15: Makefile в vagrant

15) Так как я работаю в ОС Windows, то и команды буду использовать для этой системы. Я скачал файл `vagrant-virtualbox-rocky-9-x86_64.box` по ссылке на туис, поэтому пропустил шаг с его созданием. Следующим шагом было регистрация образа виртуальной машины.



```
C:\work\labs>vagrant box add rocky9 vagrant-virtualbox-rocky-9-x86_64.box
==> box: Box file was not detected as metadata. Adding it directly...
==> box: Adding box 'rocky9' (v0) for provider:
box: Unpacking necessary files from: file://C:/work/labs/vagrant-virtualbox-rocky-9-x86_64.box
box:
```

Рис. 2.16: Регистрация образа виртуальной машины

16) С помощью консольной команды я запускаю виртуальную машину Server.

```

C:\work\aalushin\vagrant>vagrant up server
Bringing machine 'server' up with 'virtualbox' provider...
==> server: You assigned a static IP ending in ".1" to this machine.
==> server: This is very often used by the router and can cause the
==> server: network to not work properly. If the network doesn't work
==> server: properly, try changing this IP.
==> server: Preparing master VM for linked clones...
server: This is a one time operation. Once the master VM is prepared,
server: it will be used as a base for linked clones, making the creation
server: of new VMs take milliseconds on a modern system.

```

Рис. 2.17: Запуск Server

17) С помощью консольной команды я запускаю виртуальную машины Client.

```

C:\work\aalushin\vagrant>vagrant up client
Bringing machine 'client' up with 'virtualbox' provider...
==> client: Cloning VM...
==> client: Matching MAC address for NAT networking...
==> client: Setting the name of the VM: client
==> client: Fixed port collision for 22 => 2222. Now on port 220
0.
==> client: Clearing any previously set network interfaces...
==> client: Preparing network interfaces based on configuration.
..
client: Adapter 1: nat
client: Adapter 2: intnet

```

Рис. 2.18: Запуск client

18) Убедился что обе машины работают хорошо и параллельно. Ничего не вылетает, ошибок нет.

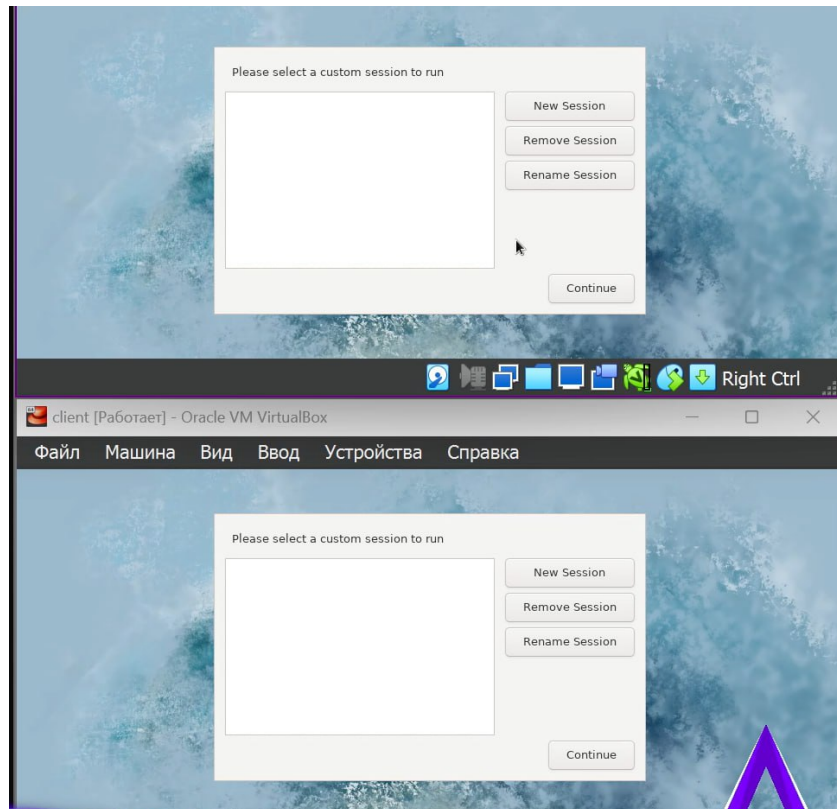


Рис. 2.19: Сверка машин

- 19) Подключился к серверу из консоли. Ввёл пароль для пользователя вагрант. Перешёл на пользователя aalushin.

```
C:\work\aalushin\vagrant>vagrant ssh server
==> server: The machine you're attempting to SSH into is configured to use
==> server: password-based authentication. Vagrant can't script entering the
==> server: password for you. If you're prompted for a password, please enter
==> server: the same password you have configured in the Vagrant file.
vagrant@127.0.0.1's password:
Last failed login: Fri Sep 6 20:52:58 UTC 2024 from 10.0.2.2 on ssh:notty
There were 3 failed login attempts since the last successful login.
Last login: Fri Sep 6 20:52:02 2024
[vagrant@server ~]$ su - aalushin
Password:
[aalushin@server.net ~]$ logout
```

Рис. 2.20: Подключение к серверу

- 20) Подключился к клиенту из консоли. Ввёл пароль для пользователя вагрант. Перешёл на пользователя aalushin.

```
C:\work\aalushin\vagrant>vagrant ssh client
==> client: The machine you're attempting to SSH into is configured to use
==> client: password-based authentication. Vagrant can't script entering the
==> client: password for you. If you're prompted for a password, please enter
==> client: the same password you have configured in the Vagrant file.
vagrant@127.0.0.1's password:
vagrant@127.0.0.1's password:
Last failed login: Fri Sep 6 20:55:06 UTC 2024 from 10.0.2.2 on ssh:notty
There was 1 failed login attempt since the last successful login
.
Last login: Fri Sep 6 20:33:30 2024
[vagrant@client ~]$ su - aalushin
Password:
[aalushin@client.net ~]$
```

Рис. 2.21: Подключение к клиенту

- 21) После того как разлогинился от двух машин, обе виртуальные машины я закрыл.

```
C:\work\aalushin\vagrant>vagrant halt client
==> client: Attempting graceful shutdown of VM...

C:\work\aalushin\vagrant>vagrant halt server
```

Рис. 2.22: Выключение машин

- 22) Убедился, что в конфигурации файла Vagrantfile есть необходимая запись.


```

## Common configuration
config.vm.provision "common dummy",
    type: "shell",
    preserve_order: true,
    path: "provision/default/01-dummy.sh"

config.vm.provision "common hostname",
    type: "shell",
    preserve_order: true,
    run: "always",
    path: "provision/default/01-hostname.sh"

config.vm.provision "common user",
    type: "shell",
    preserve_order: true,
    path: "provision/default/01-user.sh"

```

Рис. 2.23: Запись в Vagrantfile

- 23) Я зафиксировал машин Server и Client (фотографии с вводом команды для Client нет, но команда аналогичная, только вместо сервера пишем клиент)

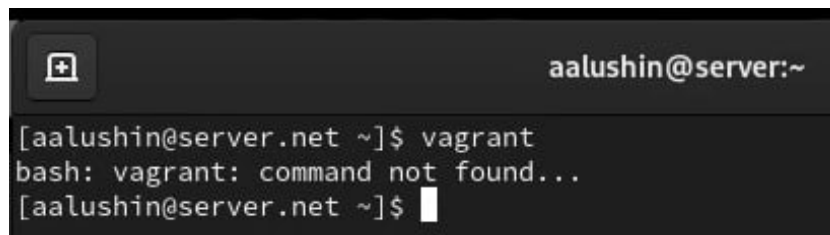
```

C:\work\aalushin\vagrant>vagrant up server --provision
Bringing machine 'server' up with 'virtualbox' provider...
==> server: You assigned a static IP ending in ".1" to this machine.
==> server: This is very often used by the router and can cause the
==> server: network to not work properly. If the network doesn't
work
==> server: properly, try changing this IP.
==> server: You assigned a static IP ending in ".1" to this machine.
==> server: This is very often used by the router and can cause the
==> server: network to not work properly. If the network doesn't
work
==> server: properly, try changing this IP.

```

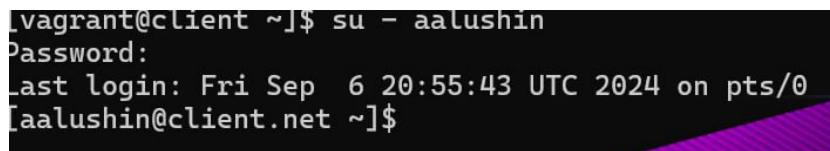
Рис. 2.24: Фиксация машин

- 24) Проверил, что обе машины работают от имени пользователя, разлогинился и выключил машины. Так же создал папку на рабочем столе, чтобы можно было переносить системы на другие компьютеры.



```
aalushin@server:~  
[aalushin@server.net ~]$ vagrant  
bash: vagrant: command not found...  
[aalushin@server.net ~]$
```

Рис. 2.25: Проверка работоспособности Сервер



```
vagrant@client ~]$ su - aalushin  
Password:  
Last login: Fri Sep 6 20:55:43 UTC 2024 on pts/0  
[aalushin@client.net ~]$
```

Рис. 2.26: Проверка работоспособности Клиент

3 Вывод

Я приобрёл практические навыки установки Rocky Linux на виртуальную машину с помощью инструмента vagrant.

4 Контрольные вопросы

1) Для чего предназначен Vagrant?

Ответ: Инструмент для создания и управления средами виртуальных машин в одном рабочем процессе.

2) Что такое box-файл? В чём назначение Vagrantfile? Box-file - сохранённый образ виртуальной машины с развёрнутой в ней операционной системой. То есть этот файл используется как основа для клонирования виртуальным машин. Vagrantfile - конфигурационный файл, написанный на языке Ruby, в котором указаны настройки запуска виртуальных машин.

3) Приведите описание и примеры вызова основных команда Vagrant.

- `vagrant help` — вызов справки по командам Vagrant;
- `vagrant box list` — список подключённых к Vagrant box-файлов;
- `vagrant box add` — подключение box-файла к Vagrant
- `vagrant destroy` — отключение box-файла от Vagrant и удаление его из виртуального окружения;
- `vagrant init` — создание «шаблонного» конфигурационного файла Vagrantfile для его последующего изменения
- `vagrant up` — запуск виртуальной машины с использованием инструкций по запуску из конфигурационного файла Vagrantfile;

- `vagrant reload` — перезагрузка виртуальной машины;
- остановка и выключение виртуальной машины;
- `vagrant provision` — настройка внутреннего окружения имеющейся виртуальной машины (например, добавление новых инструкций (скриптов) в ранее созданную виртуальную машину);
- `vagrant ssh` — подключение к виртуальной машине через ssh.

4) Дайте построчные пояснения содержания файлов `vagrant-rocky.pkr.hcl`, `ks.cfg`, `Vagrantfile`, `Makefile`.

Пример содержимого файла `Vagrantfile`: `# -- mode: ruby -- # vi: set ft=ruby :`
`Vagrant.configure(2) do |config|`
`config.vm.box = "BOX_NAME" config.vm.hostname = "HOST_NAME" config.vm.network`
`"private_network", ip: "192.168.1.1" config.vm.define "VM_NAME" config.vm.provider`
`"virtualbox" do |vb| vb.gui = false vb.memory = "1024" end end`

Первые две строки указывают на режим работы с `Vagrantfile` и использование языка Ruby. Затем идёт цикл `do`, заменяющий конструкцию `Vagrant.configure` далее по тексту на `config`. Строка `config.vm.box = "BOX_NAME"` задаёт название образа (box-файла) виртуальной машины (обычно выбирается из официального репозитория). Строка `config.vm.hostname = "HOST_NAME"` задаёт имя виртуальной машины. Конструкция `config.vm.network` задаёт тип сетевого соединения и может иметь следующие назначения:

- `config.vm.network "private_network", ip: "xxx.xxx.xxx.xxx"` — адрес из внутренней сети;
- `config.vm.network "public_network", ip: "xxx.xxx.xxx.xxx"` — публичный адрес, по которому виртуальная машина будет доступна;
- `config.vm.network "private_network", type: "dhcp"` — адрес, назначаемый по протоколу DHCP.

Строка `config.vm.define "VM_NAME"` задаёт название виртуальной машины, по которому можно обращаться к ней из Vagrant и VirtualBox. В конце идёт конструкция, определяющая параметры провайдера, а именно запуск виртуальной машины без графического интерфейса и с выделением 1 ГБ памяти.