Лабораторная работа № 1. Подготовка лабораторного стенда

1.1. Цель работы

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

1.2. Предварительные сведения

Vagrant — представляет собой инструмент для создания и управления средами виртуальных машин в одном рабочем процессе.

Этот инструмент по сути позволяет автоматизировать процесс установки на виртуальную машину как основного дистрибутива операционной системы, так и настройки необходимого в дальнейшем программного обеспечения.

С проектом Vagrant и документацией по этому инструментальному средству можно ознакомиться на сайте `https://www.vagrantup.com`.

Основные понятия Vagrant:

- провайдер (provider) — система виртуализации, с которой работает Vagrant (например, VirtualBox, VMWare и т.п.);
- box-файл (или Vagrant Box) — сохранённый образ виртуальной машины с развёрнутой в ней операционной системой; по сути box-файл используется как основа для клонирования виртуальных машин с теми или иными настройками;
- Vagrantfile — конфигурационный файл, написанный на языке Ruby, в котором указаны настройки запуска виртуальной машины.

1.2.1. JSON-файл настроек виртуальной машины

JSON-файл — специальный файл с описанием метаданных по установке дистрибутива на виртуальную машину.

JSON-файл является необязательным компонентом для создания box-файлов Vagrant, но полезен, так как позволяет управлять версиями и типами провайдеров (виртуального окружения) и образов операционных систем из одного файла.

Небольшой пример структуры JSON-файла для работы с Vagrant:

```
{
  "name": "BOX_NAME",
  "description": "Описание содержимого box-файла.",
  "versions": [
    {
      "version": "VERSION_NUMBER",
      "providers": [
        {
          "name": "virtualbox",
          "iso_url": "USO_URL.iso",
          "checksum_type": "sha256",
          "checksum": "CHECKSUM"
        }
      ]
    }
  ]
}
```

Здесь можно указать название box-файла, дать краткое описание, указать версию и тип виртуального окружения, путь к образу, на основе которого будет сформирован box-файл, тип чек-суммы и собственно чек-сумму образа.

1.2.2. Основные команды Vagrant

С Vagrant можно работать, используя следующие основные команды:
- `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;
- `vagrant halt` — 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.

1.2.3. Пример конфигурации Vagrantfile

Приведём пример содержимого файла `Vagrantfile` для понимания принципов его синтаксиса:

```
# -*- 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 на 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.

1.3. Задание

1. Сформируйте box-файл с дистрибутивом CentOS для VirtualBox (см. раздел 1.4.2 или 1.4.3).
2. Запустите виртуальные машины сервера и клиента и убедитесь в их работоспособности.
3. Внесите изменения в настройки загрузки образов виртуальных машин `server` и `client`, добавив пользователя с правами администратора и изменив названия хостов (см. раздел 1.4.4).
4. Скопируйте необходимые для работы с Vagrant файлы и box-файлы виртуальных машин на внешний носитель. Используя эти файлы, вы можете попробовать развернуть виртуальные машины на другом компьютере.

1.4. Последовательность выполнения работы

Подготовить box-файл Vagrant и впоследствии использовать его можно как в ОС Linux (в дисплейном классе или на собственном компьютере), так и в ОС Windows (только на собственном компьютере). Если вы планируете работать на собственном компьютере, то убедитесь, что:

– у вас достаточно свободного места на диске в разделе, где будет разворачиваться образ виртуальной машины (рекомендуется зарезервировать порядка 15GB);
– в вашей операционной системе установлены последние версии Vagrant (`https://www.vagrantup.com`) и VirtualBox (`https://www.virtualbox.org/`);
– для ОС Windows понадобится дополнительно установить Packer (`https://www.packer.io/`) и FAR (`https://www.farmanager.com`) для удобства работы в терминале.

Далее приведена последовательность действий по подготовке box-файл Vagrant.

1. Перед началом работы с Vagrant создайте каталог для проекта.
   В ОС Linux рекомендуется работать в `/var/tmp`:
   ```
   mkdir -p /var/tmp/user_name/vagrant
   ```
   где `user_name` — идентифицирующее вас имя пользователя, обычно первые буквы инициалов и фамилия.
   В ОС Windows, например, `C:\work\user_name\vagrant`, где `user_name` — идентифицирующее вас имя пользователя, обычно первые буквы инициалов и фамилия.
2. В созданном рабочем каталоге разместите образ операционной системы CentOS (в этом практикуме будем использовать `CentOS-8.2.2004-x86_64-minimal.iso` — минимальный дистрибутив CentOS, который можно взять с сайта `https://www.centos.org`). При работе в дисплейном классе университета дистрибутив можно взять из общего каталога `/afs/dk.sci.pfu.edu.ru/common/files/iso/`.
3. В этом же каталоге разместите подготовленные заранее для работы с Vagrant файлы:
   – `vagrant-centos.json` — специальный файл с описанием метаданных по установке дистрибутива на виртуальную машину (содержание используемого в данном практикуме файла `.json` приведено в разделе 1.4.1.1); в частности, в разделе переменных этот файл содержит указание на версию дистрибутива, его хэш-функцию,

имя и пароль пользователя по умолчанию; в разделе `builders` указаны специальные синтаксические конструкции для автоматизации работы VirtualBox; в разделе `provisioners` прописаны действия (по сути shell-скрипт) по установке дополнительных пакетов дистрибутива;
- `ks.cfg` — определяет настройки для установки дистрибутива, которые пользователь обычно вводит вручную, в частности настройки языка интерфейса, языковые настройки клавиатуры, тайм-зону, сетевые настройки и т.п.; файл должен быть расположен в подкаталоге `http` (содержание используемого в данном практикуме файла `./http/ks.cfg` приведено в разделе 1.4.1.2);
- `Vagrantfile` — файл с конфигурацией запуска виртуальных машин — сервера и клиента (содержание используемого в данном практикуме на данном этапе файла `Vagrantfile` приведено в разделе 1.4.1.3);
- `Makefile` — набор инструкций для программы `make` по работе с Vagrant (содержание используемого в данном практикуме файла `Makefile` приведено в разделе 1.4.1.4)

Основное назначение `Makefile` в этом практикуме — применение команд Vagrant в ОС Linux в определённом каталоге — только в каталоге с проектом (в частности в `/var/tmp/user_name/vagrant`). Для пользователей, работающих с Vagrant в ОС Windows, `Makefile` не понадобится.

4. В этом же каталоге создайте каталог `provision` с подкаталогами `default`, `server` и `client`, в которых будут размещаться скрипты, изменяющие настройки внутреннего окружения базового (общего) образа виртуальной машины, сервера или клиента соответственно.
5. В каталогах `default`, `server` и `client` разместите заранее подготовленный скрипт-заглушку `01-dummy.sh` следующего содержания:

```
#!/bin/bash

echo "Provisioning script $0"
```

6. В каталоге `default` разместите заранее подготовленный скрипт `01-user.sh` по изменению названия виртуальной машины следующего содержания:

```
#!/bin/bash

echo "Provisioning script $0"

username=user
userpassword=123456

encpassword=`openssl passwd -1 ${userpassword}`

id -u $username
if [[ $? ]]
then
    adduser -G wheel -p ${encpassword} ${username}
    homedir=`getent passwd ${username} | cut -d: -f6`
    echo "export PS1='[\u@\H \W]\\$ '" >> ${homedir}/.bashrc
fi
```

В этом скрипте в качестве значения переменной `username` вместо `user` укажите имя пользователя, совпадающее с вашим логином, т.е. для Ивана Петровича Сидорова логин должен иметь вид `ipsidorov`.

7. В каталоге `default` разместите заранее подготовленный скрипт `01-hostname.sh` по изменению названия виртуальной машины следующего содержания:

```
#!/bin/bash

username=user

hostnamectl set-hostname "${HOSTNAME%%.*}".${username}.net
```

В этом скрипте в качестве значения переменной `username` вместо `user` укажите имя пользователя, совпадающее с вашим логином, т.е. для Ивана Петровича Сидорова логин должен иметь вид `ipsidorov`.

1.4.1. Конфигурационные файлы

1.4.1.1. Содержание файла vagrant-centos.json

```
{
    "variables": {
        "iso_url": "CentOS-8.2.2004-x86_64-minimal.iso",
        "iso_checksum": "47ab14778c823acae2ee6d365d76a9aed3f95bb8d0ad
        ↪  d23a06536b58bb5293c0",
        "iso_checksum_type": "sha256",
        "redhat_release": "8",
        "redhat_platform": "x86_64",
        "artifact_description": "CentOS 8.2 (build 2004)",
        "artifact_version": "8.2.2004",
        "ssh_username": "vagrant",
        "ssh_password": "vagrant",
        "disk_size": "40960"
    },

    "builders": [
        {
            "name": "centos-{{user `redhat_release`}}",
            "type": "virtualbox-iso",
            "vm_name": "packer-centos-vm",

            "boot_wait": "10s",
            "disk_size": "{{user `disk_size`}}",
            "guest_os_type": "RedHat_64",
            "http_directory": "http",

            "iso_url": "{{user `iso_url`}}",
            "iso_checksum": "{{user `iso_checksum`}}",
            "iso_checksum_type": "{{user `iso_checksum_type`}}",
            "guest_additions_path": "VBoxGuestAdditions.iso",

            "boot_command": [
                "<esc>",
                "<wait><esc><esc>",
                "linux
                ↪  inst.ks=http://{{.HTTPIP}}:{{.HTTPPort}}/ks.cfg
                ↪  biosdevname=0 net.ifnames=0",
                "<enter>"
            ],
```

```
            "shutdown_command": "sudo -S /sbin/halt -h -p",
            "shutdown_timeout" : "5m",

            "ssh_wait_timeout": "60m",
            "ssh_timeout": "60m",
            "ssh_username": "{{user `ssh_username`}}",
            "ssh_password": "{{user `ssh_password`}}",
            "ssh_port": 22,
            "ssh_pty": true,

            "output_directory": "builds",

            "vboxmanage": [
                [  "modifyvm",  "{{.Name}}",  "--memory",  "1024" ],
                [  "modifyvm",  "{{.Name}}",  "--cpus",  "1" ]
            ],
            "hard_drive_interface": "sata",
            "virtualbox_version_file": ".vbox_version",

            "export_opts":
            [
                "--manifest",
                "--vsys", "0",
                "--description", "{{user `artifact_description`}}",
                "--version", "{{user `artifact_version`}}"
            ]

        }
    ],

    "post-processors": [
        {
            "output": "vagrant-centos-{{user
             ↪  `redhat_release`}}-{{user `redhat_platform`}}.box",
            "compression_level": "6",
            "type": "vagrant"
        }
    ],

    "provisioners": [{
        "type": "shell",
        "inline": [
            "sleep 30",

            "sudo dnf -y install epel-release",

            "sudo dnf -y groupinstall 'Development Tools'",

            "sudo dnf -y install kernel-devel",
            "sudo dnf -y install dkms",
            "sudo mkdir /tmp/vboxguest",
```

```
            "sudo mount -t iso9660 -o loop
             ↪  /home/vagrant/VBoxGuestAdditions.iso /tmp/vboxguest",
            "cd /tmp/vboxguest",
            "sudo ./VBoxLinuxAdditions.run",
            "cd /tmp",
            "sudo umount /tmp/vboxguest",
            "sudo rmdir /tmp/vboxguest",
            "rm /home/vagrant/VBoxGuestAdditions.iso",

            "sudo dnf -y groupinstall 'Server with GUI'",

            "sudo dnf install -y mc htop tmux",

            "sudo systemctl set-default graphical.target",
            "echo Image Provisioned!"
        ]
    }]
}
```

1.4.1.2. Содержание файла ks.cfg

```
# System bootloader configuration
bootloader --append="no_timer_check console=tty0
 ↪  console=ttyS0,115200n8 net.ifnames=0 biosdevname=0 elevator=noop"
 ↪  --location=mbr --timeout=1
# Clear the Master Boot Record
zerombr
# Partition clearing information
clearpart --all
# Reboot after installation
reboot
# Use text mode install
text
# Keyboard layouts
keyboard --vckeymap=us,ru --xlayouts='us,ru'
# System language
lang en_US.UTF-8

# Network information
network  --bootproto=dhcp --device=link --activate

# System authorization information
authselect select sssd with-sudo with-mkhomedir --force
authselect apply-changes
# Root password
rootpw vagrant
user --name=vagrant --password=vagrant
firstboot --disable
# Do not configure the X Window System
# skipx
# System services
services --enabled="NetworkManager,sshd,chronyd"
# System timezone
```

```
timezone UTC --isUtc
user --name=vagrant --password=vagrant
# Disk partitioning information
part / --fstype="xfs" --size=10239

%post
# configure swap to a file
fallocate -l 2G /swapfile
chmod 600 /swapfile
mkswap /swapfile
echo "/swapfile none swap defaults 0 0" >> /etc/fstab

# sudo
echo "%vagrant ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers.d/vagrant
chmod 0440 /etc/sudoers.d/vagrant

# Fix for https://github.com/CentOS/sig-cloud-instance-build/issues/38
cat > /etc/sysconfig/network-scripts/ifcfg-eth0 << EOF
DEVICE="eth0"
BOOTPROTO="dhcp"
ONBOOT="yes"
TYPE="Ethernet"
PERSISTENT_DHCLIENT="yes"
EOF

# Decrease connection time by preventing reverse DNS lookups
# (see https://lists.centos.org/pipermail/centos-devel/2016-July/0149
 ↪  81.html
#  and man sshd for more information)
OPTIONS="-u0"
EOF

# Fix for issue #76, regular users can gain admin privileges via su
ex -s /etc/pam.d/su <<'EOF'
# allow vagrant to use su, but prevent others from becoming root or
 ↪  vagrant
/^account\s\+sufficient\s\+pam_succeed_if.so uid = 0 use_uid quiet$/
:append
account         [success=1 default=ignore] \\
                                pam_succeed_if.so user = vagrant
 ↪  use_uid quiet
account         required        pam_succeed_if.so user notin
 ↪  root:vagrant

:update
:quit
EOF

# systemd should generate a new machine id during the first boot, to
# avoid having multiple Vagrant instances with the same id in the
 ↪  local
# network. /etc/machine-id should be empty, but it must exist to
 ↪  prevent
```

```
# boot errors (e.g.  systemd-journald failing to start).
:>/etc/machine-id

#echo 'vag' > /etc/yum/vars/infra

# Blacklist the floppy module to avoid probing timeouts
echo blacklist floppy > /etc/modprobe.d/nofloppy.conf
chcon -u system_u -r object_r -t modules_conf_t
 ↪  /etc/modprobe.d/nofloppy.conf

# Customize the initramfs
pushd /etc/dracut.conf.d
# There's no floppy controller, but probing for it generates timeouts
echo 'omit_drivers+=" floppy "' > nofloppy.conf
popd
# Fix the SELinux context of the new files
restorecon -f - <<EOF
/etc/sudoers.d/vagrant
#/etc/dracut.conf.d/vmware-fusion-drivers.conf
#/etc/dracut.conf.d/hyperv-drivers.conf
/etc/dracut.conf.d/nofloppy.conf
EOF

# Rerun dracut for the installed kernel (not the running kernel):
KERNEL_VERSION=$(rpm -q kernel --qf '%{version}-%{release}.%{arch}\n')
dracut -f /boot/initramfs-${KERNEL_VERSION}.img ${KERNEL_VERSION}

# Seal for deployment
rm -rf /etc/ssh/ssh_host_*
hostnamectl set-hostname localhost.localdomain
rm -rf /etc/udev/rules.d/70-*
%end

%packages --instLangs=en
bash-completion
bzip2
chrony
cifs-utils
man-pages
nfs-utils
rsync
yum-utils
kernel-devel
-dracut-config-rescue
-iwl100-firmware
-iwl1000-firmware
-iwl105-firmware
-iwl135-firmware
-iwl2000-firmware
-iwl2030-firmware
-iwl3160-firmware
-iwl3945-firmware
-iwl4965-firmware
```

```
-iwl5000-firmware
-iwl5150-firmware
-iwl6000-firmware
-iwl6000g2a-firmware
-iwl6050-firmware
-iwl7260-firmware
-microcode_ctl
-plymouth

%end

%addon com_redhat_kdump --disable --reserve-mb='128'

%end
```

1.4.1.3. Содержание файла Vagrantfile

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  # 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"

  # Server configuration
  config.vm.define "server", autostart: false do |server|
    server.vm.box = "centos8"
    server.vm.hostname = 'server'

    server.ssh.insert_key = false
    server.ssh.username = 'vagrant'
    server.ssh.password = 'vagrant'

    server.vm.network :private_network, ip: "192.168.1.1",
     ↪  virtualbox__intnet: true

    server.vm.provision "server dummy",
      type: "shell",
```

```
      preserve_order: true,
      path: "provision/server/01-dummy.sh"

    server.vm.provider :virtualbox do |v|
      v.linked_clone = true
      v.customize ["modifyvm", :id, "--natdnshostresolver1", "on"]
      # Customize the amount of memory on the VM
      v.memory = 1024
      v.cpus = 1
      v.name = "server"
      # Display the VirtualBox GUI when booting the machine
      v.gui = true
      # Set the video memory to 12Mb
      v.customize ["modifyvm", :id, "--vram", "12"]
    end
  end

  # Client configuration
  config.vm.define "client", autostart: false do |client|
    client.vm.box = "centos8"
    client.vm.hostname = 'client'

    client.ssh.insert_key = false
    client.ssh.username = 'vagrant'
    client.ssh.password = 'vagrant'

    client.vm.network :private_network, type: "dhcp",
     ↪  virtualbox__intnet: true

    client.vm.provision "client dummy",
      type: "shell",
      preserve_order: true,
      path: "provision/client/01-dummy.sh"

    client.vm.provider :virtualbox do |v|
      v.linked_clone = true
      v.customize ["modifyvm", :id, "--natdnshostresolver1", "on"]
      # Customize the amount of memory on the VM
      v.memory = 1024
      v.cpus = 1
      v.name = "client"
      # Display the VirtualBox GUI when booting the machine
      v.gui = true
      # Set the video memory to 12Mb
      v.customize ["modifyvm", :id, "--vram", "12"]
    end
  end
end
```

1.4.1.4. Содержание файла Makefile

```
.PHONY: version
```

```
help:
        @echo 'Usage:'
        @echo '  make <target>'
        @echo
        @echo 'Targets:'
        @grep -E '^[a-zA-Z_0-9.-]+:.*?## .*$$' $(MAKEFILE_LIST) |
         ↪  sort | awk 'BEGIN {FS = ":.*?## "}; {printf "
         ↪  \033[36m%-30s\033[0m %s\n", $$1, $$2}'
        @echo

all: box add2vagrant

box:    ## Build box from CentOS
        @export TMPDIR=`pwd`; packer build vagrant-centos.json

add2vagrant:        ## Add the built box to Vagrant
        @export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; vagrant box add
         ↪  centos8 vagrant-centos-8-x86_64.box

up:     ## Up boxies
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant up
        @VBoxManage setproperty machinefolder default

server:       ## Up server
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant up server
        @VBoxManage setproperty machinefolder default

client:       ## Up client
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant up client
        @VBoxManage setproperty machinefolder default

server-provision:       ## Up and provision server
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant up server --provision
        @VBoxManage setproperty machinefolder default

client-provision:       ## Up and provision client
```

```
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant up client --provision
        @VBoxManage setproperty machinefolder default

server-destroy:        ## Destroy server
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant destroy server
        @VBoxManage setproperty machinefolder default

client-destroy:        ## Destroy client
        @VBoxManage setproperty machinefolder `pwd`/vm
        -@export VAGRANT_HOME=`pwd`/.vagrant.d; export
         ↪  VAGRANT_DOTFILE_PATH=`pwd`/.vagrant; export
         ↪  VBOX_USER_HOME=`pwd`/.vbox; export
         ↪  VBOX_INSTALL_PATH=`pwd`/vm; vagrant destroy client
        @VBoxManage setproperty machinefolder default
```

1.4.2. Развёртывание лабораторного стенда на ОС Linux

1. Перейдите в каталог с проектом:

   ```
   cd /var/tmp/user_name/vagrant
   ```

   где `user_name` — идентифицирующее вас имя пользователя, обычно первые буквы инициалов и фамилия.
2. В терминале наберите

   ```
   make help
   ```

   Вы увидите перечень указанных в `Makefile` целей и краткое описание их действий.
3. Для формирования box-файла с дистрибутивом CentOS для VirtualBox в терминале наберите:

   ```
   make box
   ```

   Начнётся процесс скачивания, распаковки и установки драйверов VirtualBox и дистрибутива ОС на виртуальную машину. Во время автоматического развёртывания дистрибутива можно просматривать выводимую на экран информацию с разных окон, перемещаясь по ним с помощью клавиш Alt-Tab.
   После завершения процесса автоматического развёртывания образа виртуальной машины в каталоге `/var/tmp/user_name/vagrant` временно появится каталог `builds` с промежуточными файлами `.vdi`, `.vmdk` и `.ovf`, которые затем автоматически будут преобразованы в box-файл сформированного образа: `vagrant-centos-8-x86_64.box`.
4. Сохраните файлы `vagrant-centos-8-x86_64.box`, `vagrant-centos.json`, `./http/ks.cfg`, `Vagrantfile` и `Makefile` на внешний носитель или другой каталог.
5. Для регистрации образа виртуальной машины в Vagrant в терминале в каталоге `/var/tmp/user_name/vagrant` наберите

   ```
   make add2vagrant
   ```

Это позволит на основе конфигурации, прописанной в файле `Vagrantfile`, сформировать box-файлы образов двух виртуальных машин — сервера и клиента с возможностью их параллельной или индивидуальной работы.

6. Запустите виртуальную машину Server, введя
   ```
   make server
   ```
7. Запустите виртуальную машину Client, введя
   ```
   make client
   ```
8. Убедитесь, что запуск обеих виртуальных машин прошёл успешно, залогиньтесь под пользователем `vagrant` с паролем `vagrant`. Затем выключите обе виртуальные машины.

1.4.3. Развёртывание лабораторного стенда на ОС Windows

В данном разделе приведена последовательность действий при развёртывании образа виртуальной машины в ОС Windows на домашнем компьютере в VirtualBox с использованием Vagrant. После установки необходимого программного обеспечения не забудьте перегрузить систему.

Далее выполните следующие действия:

1. Используя FAR, перейдите в созданный вами рабочий каталог с проектом. В этом же каталоге должен быть размещён файл `packer.exe`. В командной строке введите
   ```
   packer.exe build vagrant-centos.json
   ```
   для начала автоматической установки образа операционной системы CentOS в VirtualBox и последующего формирования box-файла с дистрибутивом CentOS для VirtualBox. По окончании процесса в рабочем каталоге сформируется box-файл с названием `vagrant-centos-8-x86_64.box`.
2. Для регистрации образа виртуальной машины в `vagrant` в командной строке введите
   ```
   vagrant box add centos8 vagrant-centos-8-x86_64.box
   ```
3. Для запуска виртуальной машины Server введите в консоли
   ```
   vagrant up server
   ```
4. Для запуска виртуальной машины Client введите в консоли
   ```
   vagrant up client
   ```
5. Убедитесь, что запуск обеих виртуальных машин прошёл успешно. Корректно выключите виртуальные машины.

1.4.4. Внесение изменений в настройки внутреннего окружения виртуальной машины

1. Для отработки созданных скриптов во время загрузки виртуальных машин убедитесь, что в конфигурационном файле `Vagrantfile` до строк с конфигурацией сервера имеется следующая запись:
   ```
   # Common configuration
   config.vm.provision "common user",
     type: "shell",
     preserve_order: true,
     path: "provision/default/01-user.sh"

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

2. Зафиксируйте внесённые изменения для внутренних настроек виртуальных машин, введя в терминале:
   ```
   make server-provision
   ```
   Затем
   ```
   make client-provision
   ```
   Для работающих под ОС Windows вместо инструкций `Makefile` следует последовательно ввести в командной строке:
   ```
   vagrant up server --provision
   vagrant up client --provision
   ```
3. Залогиньтесь на сервере и клиенте под созданным пользователем. Убедитесь, что в терминале приглашение отображается в виде `user@server.user.net` на сервере и в виде `user@client.user.net` на клиенте, где вместо `user` указан ваш логин.
4. Выключите виртуальные машины.
5. После выключения виртуальных машин скопируйте необходимые для работы с Vagrant файлы и box-файлы виртуальных машин на внешний носитель или в другой каталог вашей ОС. Используя эти файлы, вы можете развернуть виртуальные машины на другом компьютере.

1.5. Содержание отчёта

1. Титульный лист с указанием номера лабораторной работы и ФИО студента.
2. Формулировка задания работы.
3. Описание результатов выполнения задания:
   - скриншоты (снимки экрана), фиксирующие выполнение работы;
   - подробное описание настроек служб в соответствии с заданием;
   - полные тексты конфигурационных файлов настраиваемых в работе служб;
   - результаты проверки корректности настроек служб в соответствии с заданием (подтверждённые скриншотами).
4. Выводы, согласованные с заданием работы.
5. Ответы на контрольные вопросы.

1.6. Контрольные вопросы

1. Для чего предназначен Vagrant?
2. Что такое box-файл? В чём назначение `Vagrantfile`?
3. Приведите описание и примеры вызова основных команд Vagrant.
4. Дайте построчные пояснения содержания файлов `vagrant-centos.json`, `ks.cfg`, `Vagrantfile`, `Makefile`.

При ответах на вопросы рекомендуется ознакомиться с источниками [1—5].

Список литературы

1. GNU Bash Manual. — 2019. — URL: `https://www.gnu.org/software/bash/manual/`.
2. GNU Make Manual. — 2016. — URL: `http://www.gnu.org/software/make/manual/`.
3. *Powers S.* Vagrant Simplified [Просто о Vagrant] / Перевод: А. Панин // Библиотека сайта rus-linux.net. — 2015. — URL: `http://rus-linux.net/MyLDP/vm/vagrant-simplified.html`.

4. Vagrant Documentation. — URL: https://www.vagrantup.com/docs/index.html.

5. *Купер М.* Искусство программирования на языке сценариев командной оболочки. — 2004. — URL: https://www.opennet.ru/docs/RUS/bash_scripting_guide/.