Методическое пособие по выполнению домашнего задания по курсу «Администратор Linux. Professional»

Домашнее задание VPN

Цель домашнего задания

Создать домашнюю сетевую лабораторию. Научится настраивать VPN-сервер в Linux-based системах.

Описание домашнего задания

1. Настроить VPN между двумя ВМ в tun/tap режимах, замерить скорость в туннелях, сделать вывод об отличающихся показателях
2. Поднять RAS на базе OpenVPN с клиентскими сертификатами, подключиться с локальной машины на ВМ
3. (*) Самостоятельно изучить и настроить ocserv, подключиться с хоста к ВМ

Формат сдачи: Vagrantfile + ansible

Функциональные и нефункциональные требования

- ПК на Unix с 8 ГБ ОЗУ или виртуальная машина с включенной Nested Virtualization.

Предварительно установленное и настроенное следующее ПО:

- Hashicorp Vagrant (https://www.vagrantup.com/downloads)
- Oracle VirtualBox (https://www.virtualbox.org/wiki/Linux_Downloads).
- Ansible (версия 2.8 и выше) - https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html
- Любой редактор кода, например Visual Studio Code, Atom и т.д.

Инструкция по выполнению домашнего задания

Все дальнейшие действия были проверены при использовании Vagrant 2.2.19, VirtualBox v6.1.26 r145957. В качестве ОС на хостах установлена Ubuntu 22.04. Серьёзные отступления от этой конфигурации могут потребовать адаптации с вашей стороны.

1. TUN/TAP режимы VPN

Для выполнения первого пункта необходимо написать Vagrantfile, который будет поднимать 2 виртуальные машины server и client. Типовой пример Vagrantfile для данной задачи:

```
Vagrant.configure("2") do |config|
      config.vm.box = "ubuntu/jammy64"
      config.vm.define "server" do |server|
            server.vm.hostname = "server.loc"
            server.vm.network "private_network", ip: "192.168.56.10"
      end

      config.vm.define "client" do |client|
            client.vm.hostname = "client.loc"
            client.vm.network "private_network", ip: "192.168.56.20"
      end
end
```

После запуска машин из Vagrantfile необходимо выполнить следующие действия на server и client машинах:

```
      # Устанавливаем нужные пакеты и отключаем SELinux
      apt update
      apt install openvpn iperf3 selinux-utils
      setenforce 0
```

Настройка хоста 1:

```
      # Создаем файл-ключ
      openvpn --genkey secret /etc/openvpn/static.key
      # Создаем конфигурационный файл OpenVPN
      vim /etc/openvpn/server.conf

      # Содержимое файла server.conf
      dev tap
      ifconfig 10.10.10.1 255.255.255.0
      topology subnet
      secret /etc/openvpn/static.key
      comp-lzo
      status /var/log/openvpn-status.log
      log /var/log/openvpn.log
      verb 3


      # Создаем service unit для запуска OpenVPN
      vim /etc/systemd/system/openvpn@.service


      # Содержимое файла-юнита
      [Unit]
```

```
Description=OpenVPN Tunneling Application On %I
After=network.target
[Service]
Type=notify
PrivateTmp=true
ExecStart=/usr/sbin/openvpn --cd /etc/openvpn/ --config
%i.conf
[Install]
WantedBy=multi-user.target
```

```
# Запускаем сервис
systemctl start openvpn@server
systemctl enable openvpn@server
```

Настройка хоста 2:

```
# Создаем конфигурационный файл OpenVPN
vim /etc/openvpn/server.conf

# Содержимое конфигурационного файла
dev tap
remote 192.168.56.10
ifconfig 10.10.10.2 255.255.255.0
topology subnet
route 192.168.56.0 255.255.255.0
secret /etc/openvpn/static.key
comp-lzo
status /var/log/openvpn-status.log
log /var/log/openvpn.log
verb 3
```

На хост 2 в директорию /etc/openvpn необходимо скопировать файл-ключ static.key, который был создан на хосте 1.

```
# Создаем service unit для запуска OpenVPN
vim /etc/systemd/system/openvpn@.service

# Содержимое файла-юнита
[Unit]
Description=OpenVPN Tunneling Application On %I
After=network.target
[Service]
```

```
Type=notify
PrivateTmp=true
ExecStart=/usr/sbin/openvpn --cd /etc/openvpn/ --config
%i.conf
[Install]
WantedBy=multi-user.target
```

```
# Запускаем сервис
systemctl start openvpn@server
systemctl enable openvpn@server
```

Далее необходимо замерить скорость в туннеле:

1) На хосте 1 запускаем iperf3 в режиме сервера: iperf3 -s &
2) На хосте 2 запускаем iperf3 в режиме клиента и замеряем скорость в туннеле: iperf3 -c 10.10.10.1 -t 40 -i 5

Повторяем пункты 1-2 для режима работы tun.
Конфигурационные файлы сервера и клиента изменятся только в директиве dev. Делаем выводы о режимах, их достоинствах и недостатках.

2. RAS на базе OpenVPN

Для выполнения данного задания можно воспользоваться Vagrantfile из 1 задания, только убрать одну ВМ. После запуска отключаем SELinux (setenforce 0) или создаём правило для него.

Настройка сервера:

```
# Устанавливаем необходимые пакеты
apt update
apt install openvpn easy-rsa
```

```
# Переходим в директорию /etc/openvpn и инициализируем PKI
cd /etc/openvpn
/usr/share/easy-rsa/easyrsa init-pki
# Генерируем необходимые ключи и сертификаты для сервера
echo 'rasvpn' | /usr/share/easy-rsa/easyrsa gen-req server
nopass
```

```
     echo 'yes' | /usr/share/easy-rsa/easyrsa sign-req server
server
     /usr/share/easy-rsa/easyrsa gen-dh
     openvpn --genkey secret ca.key

     # Генерируем необходимые ключи и сертификаты для клиента
     echo 'client' | /usr/share/easy-rsa/easyrsa gen-req client
nopass
     echo 'yes' | /usr/share/easy-rsa/easyrsa sign-req client
client

     # Создаем конфигурационный файл сервера
     vim /etc/openvpn/server.conf

     # Зададим параметр iroute для клиента
     echo 'iroute 10.10.10.0 255.255.255.0' >
/etc/openvpn/client/client

     # Содержимое файла server.conf
     port 1207
     proto udp
     dev tun
     ca /etc/openvpn/pki/ca.crt
     cert /etc/openvpn/pki/issued/server.crt
     key /etc/openvpn/pki/private/server.key
     dh /etc/openvpn/pki/dh.pem
     server 10.10.10.0 255.255.255.0
     ifconfig-pool-persist ipp.txt
     client-to-client
     client-config-dir /etc/openvpn/client
     keepalive 10 120
     comp-lzo
     persist-key
     persist-tun
     status /var/log/openvpn-status.log
     log /var/log/openvpn.log
     verb 3


     # Запускаем сервис (при необходимости создать файл юнита как в
      задании 1)
     systemctl start openvpn@server
     systemctl enable openvpn@server


На хост-машине:
```

1) Необходимо создать файл client.conf со следующим содержимым:

```
dev tun
proto udp
remote 192.168.56.10 1207
client
resolv-retry infinite
remote-cert-tls server
ca ./ca.crt
cert ./client.crt
key ./client.key
route 192.168.56.0 255.255.255.0
persist-key
persist-tun
comp-lzo
verb 3
```

2) Скопировать в одну директорию с client.conf файлы с сервера:

```
/etc/openvpn/pki/ca.crt
/etc/openvpn/pki/issued/client.crt
/etc/openvpn/pki/private/client.key
```

Далее можно проверить подключение с помощью: openvpn --config client.conf

При успешном подключении проверяем пинг по внутреннему IP адресу сервера в туннеле: ping -c 4 10.10.10.1

Также проверяем командой ip r (netstat -rn) на хостовой машине что сеть туннеля импортирована в таблицу маршрутизации.

3. (*) OpenConnect-сервер

Материал по данному заданию необходимо самостоятельно изучить и поднять OpenConnect сервер и подключиться к нему с хост-машины.

Критерии оценивания

Статус «Принято» ставится при выполнении следующих условий:

1. Ссылка на репозиторий GitHub.
2. Vagrantfile, который будет разворачивать виртуальные машины
3. Настройка виртуальных машин происходит с помощью Ansible.
4. Документация по каждому заданию:

Создайте файл README.md и снабдите его следующей информацией:
- название выполняемого задания;
- текст задания;
- схема сети;
- особенности проектирования и реализации решения,
- заметки, если считаете, что имеет смысл их зафиксировать в репозитории.