МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
Кафедра компьютерной инженерии и моделирования

ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №6
«Знакомство с Ansible»

Практическая работа
по дисциплине «Современные технологии программирования»
студента 1 курса группы ПИ-б-о-233
Иващенко Дениса Олеговича

направления подготовки 09.03.04 «Программная инженерия»

Симферополь, 2024

Цель:Ознакомиться на практике с инструментом для удаленного управления конфигурациями Ansible.

Ход работы:

Ansible. Ручной запуск модулей. Playbook

Подготовка виртуальных машин

1)Нам понадобится две чистые виртуальные машины. Если у вас дефицит дискового пространства, то ВМ из предыдущих работ можно удалить, здесь они нам не понадобятся.

На одной из виртуальных машин будет размещаться ansible, поэтому я её так и назову - "Ansible", а вторая будет просто "коробкой" для других виртуальных машин, которыми мы будем управлять при помощи ansible, поэтому я назову её "Internet" (имена можете выбрать по своему вкусу).

2)Создайте две виртуальные машины с Ubuntu Server . В процессе установки используйте стандартные настройки. Из дополнительного софта понадобится только ssh-сервер.

Для машины с именем "Internet" создайте диск большого объёма 50+ГБ.

3)В настройках сети VirtualBox установите "Сетевой мост" для обеих машин.

Машина с Ansible

1)Для удобства дальнейшей работы, подключитесь к машине по ssh.

2)Добавьте в список dns-серверов гугловский (8.8.8.8) и(или) яндексовый (77.88.8.8).

3)Обновите индексы пакетов: sudo apt-get update.

4)Проверьте, что в системе установлен python 3: python3 --version. Если нет, установите.

5)Проверьте, что в системе установлен pip: python3 -m pip -V. Если нет, установите.

6)Установите последнюю доступную версию ansible: python3 -m pip install --user ansible.

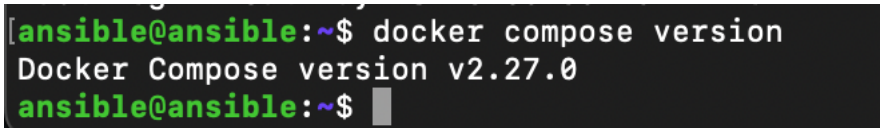
Флаг --user установит пакет ansible как локальный, т.е. ansible будет доступен только текущему пользователю.

7)После установки выскочит предупреждение, что запустить ansible по имени не получится, т.к. каталог в который он установился (~/.local/bin) не добавлен в PATH.
Нам не придётся делать это вручную, т.к. путь ~/.local/bin будет добавлен в PATH автоматически после перезагрузки или перелогина. Вместо перезагрузки можно выполнить команду source ~/.profile, что вызовет принудительное обновление PATH.

8)Теперь убедитесь, что ansible установился: ansible --version.
В результате вы увидите довольно подробный вывод о самой версии ansible, о путях к конфигам, версии python и т.д.

Машина с Docker Compose

1)Для удобства дальнейшей работы, подключитесь к машине по ssh например при помощи KiTTY.
2)Добавьте в список dns-серверов гугловский (8.8.8.8) и(или) яндексовый (77.88.8.8).
3)Обновите индексы пакетов: sudo apt-get update.
4)Устанавливать docker и docker compose будем из официальных репозиториев Docker



Создаём сервера

1)Выполните команду ip a и найдите **сетевой интерфейс**, который подключён к вашему роутеру.

```
ansible@ansible:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group d
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
oup default qlen 1000
    link/ether 08:00:27:17:8e:5a brd ff:ff:ff:ff:ff:ff
    inet 192.168.89.78/24 metric 100 brd 192.168.89.255 scope global dynami
0s3
       valid_lft 468sec preferred_lft 468sec
    inet6 fe80::a00:27ff:fe17:8e5a/64 scope link
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue stat
N group default
    link/ether 02:42:c7:0d:a4:a7 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```

2)Выполните команду ip r и найдите default. Этот ip - **gateway** (в нашем случае это ip роутера).

```
ansible@ansible:~$ ip r
default via 192.168.89.1 dev enp0s3 proto dhcp src 192.168.89.78 metric 100
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
172.31.254.1 via 192.168.89.1 dev enp0s3 proto dhcp src 192.168.89.78 metric 100

185.112.140.5 via 192.168.89.1 dev enp0s3 proto dhcp src 192.168.89.78 metric 10
0
185.112.140.7 via 192.168.89.1 dev enp0s3 proto dhcp src 192.168.89.78 metric 10
0
192.168.89.0/24 dev enp0s3 proto kernel scope link src 192.168.89.78 metric 100
192.168.89.1 dev enp0s3 proto dhcp scope link src 192.168.89.78 metric 100
ansible@ansible:~$
```

3)Теперь нужно посмотреть занятые ip адреса в нашей локальной сети. Это нужно, чтобы выбрать диапазон свободных для наших серверов.
Установите утилиту arp при помощи команды: sudo apt install net-tools, а затем выполните arp -e. IP-адреса из списка заняты.

```
arp ansible@ansible:~$ arp -e
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.89.54            ether   28:f0:76:5f:fe:e0   C                     enp0s3
_gateway                 ether   cc:2d:e0:f3:d0:3c   C                     enp0s3
192.168.89.22            ether   ac:c9:06:07:ba:ee   C                     enp0s3
```

4)Чтобы уточнить список воспользуемся утилитой nmap. Установите её (sudo apt-get install nmap) и попросите просканировать все ip в сети (sudo nmap -Pn 192.168.1.0/24). К сожалению, при помощи nmap тоже не всегда можно получить полный список занятых адресов, т.к. некоторые устройства могут временно отключаться от сети для экономии батарее и т.д.
Точный способ узнать занятые IP - посмотреть их в web-интерфейсе роутера

```
ansible@ansible:~$ sudo nmap -Pn 192.168.1.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2024-05-18 15:44 UTC
```

Теперь воспользуемся docker compose чтобы создать сервера. Для начала пропишем в нём только один сервер.
1)На сервере с docker compose создайте файл compose.yaml и откройте его в текстовом редакторе.
2)Поместите в него следующий текст

```
ansible@ansible:~$ touch compose.yaml
ansible@ansible:~$ cat <<END>> compose.yaml
> services:
  test: # Так я назвал сервис
    image: rastasheep/ubuntu-sshd # Тот самый базовый образ
    dns:
      - 77.88.8.8 # Яндексовый
      - 8.8.8.8   # Гугловый
    networks:
      outside:
        ipv4_address: 192.168.1.100 # Свободный ip из нашей подсети

# Здесь создаём docker-сеть
networks:
  outside: # Так я назвал сеть
    driver: ipvlan
    driver_opts:
      parent: enp0s3 # Сетевой интерфейс
    ipam:
      config:
        - subnet: 192.168.1.0/24 # Адрес подсети и маска
          gateway: 192.168.1.1
> END
ansible@ansible:~$
```

3)Наберите в терминале:

docker compose up -d

```
[ansible@ansible:~$ docker compose up -d
[+] Running 14/14
 ✔ test Pulled                                                    23.3s
   ✔ a48c500ed24e Pull complete                                    9.3s
   ✔ 1e1de00ff7e1 Pull complete                                    9.3s
   ✔ 0330ca45a200 Pull complete                                    9.4s
   ✔ 471db38bcfbf Pull complete                                    9.4s
   ✔ 0b4aba487617 Pull complete                                    9.4s
   ✔ b42109ad2a3d Pull complete                                   15.8s
   ✔ dde737735b18 Pull complete                                   20.7s
   ✔ d836c14266f7 Pull complete                                   20.7s
   ✔ 5ed86b5d4a15 Pull complete                                   20.8s
   ✔ 5273c120f396 Pull complete                                   20.8s
   ✔ b0299e0551df Pull complete                                   20.8s
   ✔ 0ae38e059780 Pull complete                                   20.9s
   ✔ ca79c723275f Pull complete                                   20.9s
[+] Running 2/2
 ✔ Network ansible_outside   Created                               0.0s
 ✔ Container ansible-test-1  Started                               7.3s
ansible@ansible:~$
```

4)Проверьте, что к этой машине можно подключится как и к любой другой, например при помощи KiTTY (ip, port и пользователь вам известны).

```
denisivaschenko@MacBook-Air-Denis ~ % ssh ansible@192.168.89.78
ansible@192.168.89.78's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-107-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

  System information as of Сб 18 мая 2024 15:49:09 UTC

  System load:  0.3818359375      Processes:               123
  Usage of /:   51.1% of 11.21GB  Users logged in:         1
  Memory usage: 19%               IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                IPv4 address for enp0s3:  192.168.89.78

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.
```

5)Наберите в терминале:

docker compose down

```
ansible@ansible:~$ docker compose down
[+] Running 2/1
 ✔ Container ansible-test-1  Removed                    0.4s
 ✔ Network ansible_outside   Removed                    0.0s
ansible@ansible:~$
```

Модифицируем файл таким образом, чтобы создать 3 машины: одну под - балансировщик нагрузки, вторую под базу данных и третью под приложение (пока одну, затем их будет больше).

1)Откройте файл compose.yaml и поместите следующий текст
2)Запустите машины командой: docker compose up -d и на некоторое время мы перейдём на машину с ansible.

```
ansible@ansible:~$ docker compose up -d
[+] Running 4/4
 ✔ Network ansible_outside            Created                0.0s
 ✔ Container ansible-db-1             Started                1.8s
 ✔ Container ansible-worker-1         Star...                2.0s
 ✔ Container ansible-load_balancer-1  Started                2.0s
ansible@ansible:~$
```

Соединяем Ansible с управляемыми машинами

1)Выполните команду:
ansible-inventory --list

```
ansible@ansible:~$ ansible-inventory --list
```

2)Выполните команду:
ansible --version | grep config

```
ansible@ansible:~$ ansible --version | grep config
  config file = None
```

3)Создайте каталог "app" и в нём "ansible", затем перейдите туда. Это не какие-то специальные названия, просто чтобы была понятная структура.

```
ansible@ansible:~$ mkdir app
ansible@ansible:~$ cd app
ansible@ansible:~/app$ mkdir ansible
ansible@ansible:~/app$ cd ansible
ansible@ansible:~/app/ansible$
```

4)Создайте файл "ansible.cfg" содержащий:

[defaults]

inventory = ./hosts

```
ansible@ansible:~/app/ansible$ touch ansible.cfg
ansible@ansible:~/app/ansible$ cat <<END>> ansible.cfg
> [defaults]
inventory = ./hosts
> END
ansible@ansible:~/app/ansible$
```

5)Снова выполните команду:

ansible --version | grep config

```
ansible@ansible:~/app/ansible$ ansible --version | grep config
  config file = /home/ansible/app/ansible/ansible.cfg
```

7)Создайте этот файл и напишите в нём следующее (только ip-адреса укажите свои):

load_balancer ansible_host=192.168.1.100 ansible_user=root ansible_password=root

db ansible_host=192.168.1.200 ansible_user=root ansible_password=root

worker ansible_host=192.168.1.101 ansible_user=root ansible_password=root

8)Убедитесь, что ansible подхватил информацию о хостах: ansible-inventory --list.

```
ansible@ansible:~/app/ansible$ ansible-inventory --list
```

9)Попробуем подключится к ним.

Ansible может выполнять команды в двух режимах: одиночная команда (ad hoc) или сценарий (play). Общий вид запуска одиночной команды выглядит так: ansible [pattern] -m [module] -a "[module options]"

```
[ansible@ansible:~/app/ansible$ ansible [pattern] -m [module] -a "[module options
]"
```

10)В итоге вы должны были получить ошибку, т.к. для авторизации по паролю нужен дополнительный пакет: sshpass. Установите его при помощи apt и повторите ping.

```
[ansible@ansible:~/app/ansible$ sudo apt install sshpass
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer
  python3-resolvelib
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  sshpass
0 upgraded, 1 newly installed, 0 to remove and 18 not upgraded.
Need to get 11.7 kB of archives.
After this operation, 35.8 kB of additional disk space will be used
Get:1 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 sshp
9-1 [11.7 kB]
Fetched 11.7 kB in 0s (82.4 kB/s)
Selecting previously unselected package sshpass.
(Reading database ... 114244 files and directories currently instal
Preparing to unpack .../sshpass_1.09-1_amd64.deb ...
Unpacking sshpass (1.09-1) ...
Setting up sshpass (1.09-1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
```

12)В этот раз подключение должно быть успешно, и все три сервера должны ответить "pong".

```
ansible@ansible:~$ ansible all -m ping
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
ansible@ansible:~$
```

Разделение на группы

1)Откройте файл "hosts" и замените его на (со своими ip):

load_balancer ansible_host=192.168.1.100 ansible_user=root ansible_password=root

db ansible_host=192.168.1.200 ansible_user=root ansible_password=root

[workers]

worker ansible_host=192.168.1.101 ansible_user=root ansible_password=root

2)Перейдите на виртуальную машину с docker compos, остановите запущенные машины и добавьте в "compose.yaml" ещё 3 машины.

3)Запустите машины и вернитесь на сервер с ansible.

4)Т.к. у нас появилось несколько новых машин, нам нужно добавить их в инвентарь.

Снова откройте файл "hosts" и замените его на (со своими ip):

```
ansible@ansible:~/app/ansible$ cat <<END>> hosts
> load_balancer ansible_host=192.168.1.100 ansible_user=root ansible_password=
ot
db ansible_host=192.168.1.200 ansible_user=root ansible_password=root

[prod]
worker1 ansible_host=192.168.1.101 ansible_user=root ansible_password=root
worker2 ansible_host=192.168.1.102 ansible_user=root ansible_password=root
worker3 ansible_host=192.168.1.103 ansible_user=root ansible_password=root

[staging]
worker4 ansible_host=192.168.1.104 ansible_user=root ansible_password=root

[workers:children]
prod
staging
> END
ansible@ansible:~/app/ansible$
```

5)Проверьте, что машины доступны при помощи модуля ansible ping.

Прошпигуйте все воркеры и только те, которые находятся в группе "prod".

```
[ansible@ansible:~/app/ansible$ ansible all -m ping
```

Начальное конфигурирование управляемых машин

1)Выполните команду:

cp hosts hosts_root

2)Проверьте, что копирование файла прошло успешно. Для этого пропингуйте все сервера командой:

ansible -i hosts_root all -m ping

```
[ansible@ansible:~/app/ansible$ cp hosts hosts_root
[ansible@ansible:~/app/ansible$ ansible -i hosts_root all -m ping
```

3)Обновим индексы apt-пакетов на сервере, для этого будем использовать модуль ansible apt:

ansible staging -m apt -a "update_cache=yes"

```
[ansible@ansible:~/app/ansible$ ansible staging -m apt -a "update_cache=yes"
```

4)Установим пакет sudo (да, это отдельный пакет, а не встроенная команда):

ansible staging -m apt -a "name=sudo state=latest"

5)По умолчанию новые пользователи обладают минимальными правами, чтобы дать возможность пользователю повышать свои привилегии при помощи sudo он или группа в которой он состоит должны быть записаны в файле "/etc/sudoers" с указанием доступных привилегий.

Создадим новую группу с названием "ansible" при помощи модуля group:
ansible staging -m group -a'name=ansible state=present'

```
[ansible@ansible:~/app/ansible$ ansible staging -m group -a'name=ansible state=pr]
esent'
```

6)Разрешим пользователям из этой группы выполнять любые команды, при этом без необходимости вводить пароль (нужно для ansible).
Чтобы случайно не поломать файл "/etc/sudoers" внося в него изменения, мы создадим новый файл в каталоге "/etc/sudoers.d/" (файл "/etc/sudoers" подтягивает все файлы из этого каталога):
ansible staging -m copy -a"content='%ansible ALL=(ALL:ALL) NOPASSWD:ALL' dest=/etc/sudoers.d/ansible validate='/usr/sbin/visudo -cf %s'"

```
[ansible@ansible:~/app/ansible$ ansible staging -m copy -a"content='%ansible ALL=
(ALL:ALL) NOPASSWD:ALL' dest=/etc/sudoers.d/ansible validate='/usr/sbin/visudo -
cf %s'"
```

7)Теперь настало время создать пользователя под которым в дальнейшем будет работать ansible:
ansible staging -m user -a"name=ansible shell=/bin/bash groups=ansible append=yes password={{ '123' | password_hash('sha512') }} update_password=on_create»

8)Настроим на машинах из группы "staging" доступ под пользователем "ansible" по ключу.
Для этого первым делом нам нужна пара приватный/публичный ключ которую мы будем использовать для доступа. В принципе на нашем сервере уже есть такая пара (она была создана при установке ssh-сервера), но мы создадим ещё одну.

```
ansible@ansible:~/app/ansible$ mkdir keys
ansible@ansible:~/app/ansible$ cd keys
ansible@ansible:~/app/ansible/keys$ ssh-keygen -t rsa -b 4096 -f ansible_key
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ansible_key
Your public key has been saved in ansible_key.pub
The key fingerprint is:
SHA256:JDFBZhYA1b+m7UP0UOjZKy/OaV0mWGs0E5RJQVPpZ40 ansible@ansible
The key's randomart image is:
+---[RSA 4096]----+
|   .oo+Xo =*=..   |
|      +.o. =..    |
|       .o.+ o   o |
|        o* * . E .|
|        .SB = o   |
|         * * o    |
|        = = +     |
|       ..=.o      |
|        +=o       |
+----[SHA256]-----+
ansible@ansible:~/app/ansible/keys$
```

```
ansible@ansible:~/app/ansible/keys$ ansible staging -m authorized_key -a"user=an
sible key=\"{{ lookup('file', '/home/' + lookup('env', 'USER') + '/app/ansible/k
eys/ansible_key.pub') }}\""
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
[WARNING]: Could not match supplied host pattern, ignoring: staging
```

9)Отключим пользователю "root" пароль, чтобы ни у кого не было возможности залогинится под ним:

ansible staging -m user -a"name=root password='*' update_password=always»

```
ansible@ansible:~/app/ansible/keys$ ansible staging -m user -a"name=root passwor
d='*' update_password=always"
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
[WARNING]: Could not match supplied host pattern, ignoring: staging
ansible@ansible:~/app/ansible/keys$
```

10)Подождите несколько минут и запустите пинг **всех** серверов (через ansible). Вы должны увидеть, что сервер из группы "staging" не ответит.

11)Внесём изменения в файл "hosts", чтобы наладить связь с сервером.

Откройте "hosts" в текстовом редакторе и для "worker4" (т.к. он один у нас в группе "staging")

замените ansible_user=root на ansible_user=ansible и ansible_password=root на ansib

le_ssh_private_key_file=keys/ansible_key.

Здесь путь к приватному ключу указан относительно расположения файла "hosts".

12)Убедитесь, что теперь все сервера отвечают на ping.

```
ansible@ansible:~/app/ansible/keys$ ansible all -m ping
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
```

Пишем сценарий

1)Создайте в каталоге "~/app/ansible" файл "ping.yaml";

2)Добавьте в файл следующие строки:

- name: Ping all servers

  hosts: all

```
ansible@ansible:~/app/ansible$ touch ping.yaml
ansible@ansible:~/app/ansible$ cat <<END>> ping.yaml
> - name: Ping all servers
[  hosts: all
[> END
ansible@ansible:~/app/ansible$
```

3)Теперь добавим непосредственно команды:

```
 tasks:
 - name: Ping all servers
   ping:
```

4)Запустите palybook при помощи команды:

ansible-playbook ping.yaml

```
[ansible@ansible:~/app/ansible$ ansible-playbook ping.yaml
```

Теперь напишем palybook который будет делать ровно тоже, что мы делали руками в предыдущем разделе:

1)Создайте в каталоге "~/app/ansible" файл "setup.yaml";

2)Добавьте в файл следующие строки:

```
- name: Change access from root to ansible user and apt-package update servers
  hosts: all
  become: yes
```

3)Теперь добавим непосредственно команды. Сравните ad hoc команды из предыдущего раздела с их вариантами в palybook:

```
        sed -i 's|archive.ubuntu.com|mirror.yandex.ru|g' /etc/apt/sources.list
        sed -i 's|security.ubuntu.com|mirror.yandex.ru|g' /etc/apt/sources.list
    - name: Install sudo
      apt:
        name: sudo
        state: latest
        update_cache: yes   # Перед установокой зделает update
    - name: Create group ansible
      group:
        name: ansible
        state: present
    - name: Add group to sudoers
      copy:
        content: '%ansible ALL=(ALL:ALL) NOPASSWD:ALL'
        dest: /etc/sudoers.d/ansible
        validate: '/usr/sbin/visudo -cf %s'
    - name: Create user ansible and add to ansible group
      user:
        name: ansible
        shell: /bin/bash
        groups: ansible
        update_password: alwayswordome/' + lookup('env', 'USER') + '/app/ansible
> END
```

4)Запустите palybook и дождитесь его завершения. В логах вы должны увидеть, что на всех серверах кроме "worker4" на каждом этапе происходили изменения (changed), а на "worker4" изменений не было (ok).

```
ansible@ansible:~/app/ansible$ ansible-playbook setup.yaml
```

5)Подождите несколько минут и запустите "ping.yaml". Теперь, все сервера, кроме "worker4" перестали отвечать, т.к. в hosts указаны данные для пользователя root.

6)Поправьте файл hosts и убедитесь, что теперь все сервера на связи.

Установка необходимого софта

1)Создайте в каталоге "~/app/ansible" файл «install_mysql.yaml";

2)Добавьте в файл следующие строки:

```
ansible@ansible:~/app/ansible$ cat <<END>> install_mysql.yaml
> - name: Install_mysql
  hosts: db
  become: yes

  tasks:
    - name: Install pip
      apt:
        name: python3-pip
        state: latest
    - name: Install PyMySQL
      shell:
        cmd: python3 -m pip install PyMySQL
>     - name: Install mysql server
      apt:
        name: mysql-server
        state: latest
    - name: Stop mysql server
      sysvinit:
        name: mysql
        state: stopped
>
>     - name: Copy credentials file to home directory
      copy:
```

3)Добавьте в файл следующие строки:

```
- name: Install mysql server
  apt:
    name: mysql-server
    state: latest
- name: Stop mysql server
  sysvinit:
    name: mysql
    state: stopped
```

4)Добавьте в файл следующие строки:

```
- name: Copy credentials file to home directory
  copy:
    src: /etc/mysql/debian.cnf
    remote_src: true
    dest: /home/ansible/.my.cnf
```

```
  - name: Create ansible credentials file
    copy:
      src: /etc/mysql/debian.cnf
      remote_src: true
      dest: /home/ansible/.my_ansible.cnf
  - name: Remane user and password items
    shell: |
      sed -i "s/user/login_user/g" /home/ansible/.my_ansible.cnf
      sed -i "s/password/login_password/g" /home/ansible/.my_ansible.cnf
```

5)Добавьте в файл следующие строки:

```
  - name: Allow bind to all hosts
    shell:
      cmd: sed -i -r "s/bind-address\s{1,}= 127.0.0.1/bind-address = 0.0.0.0/" /etc/
mysql/mysql.conf.d/mysqld.cnf
  - name: Started and enabled mysql server
    sysvinit:
      name: mysql
      state: started
      enabled: yes
```

6)Добавьте в файл следующие строки:

```
  - name: Create database user
    mysql_user:
      config_file: /home/ansible/.my_ansible.cnf # Логин и пароль для входа в БД
      name: db_user_for_app       # Логин нового пользователя
      password: 123               # Пароль нового пользователя
      host: '%'                   # Разрешаем заходить с любых хостов
      priv: '*.*:ALL'             # Даём пользователю все привилегии
      state: present              # Создать
```

7)Запустите плейбук и дождитесь завершения.

```
[ansible@ansible:~/app/ansible$ ansible-playbook install_mysql.yaml
```

Установка софта на рабочие сервера

1)Создайте в каталоге "~/app/ansible" файл "install_worker_soft.yaml";

2)Добавьте в файл следующие строки

```
[ansible@ansible:~/app/ansible$ touch install_worker_soft.yaml
[ansible@ansible:~/app/ansible$ cat <<END>> install_worker_soft.yaml
> - name: Install worker soft
  hosts: workers
  become: yes

  tasks:
    - name: Install python3 pip venv git
      apt:
        name:
          - python3.8
          - python3-pip
          - python3.8-venv
          - git
```

3)Запустите playbook и дождитесь установки.

```
[ansible@ansible:~/app/ansible$ ansible-playbook install_worker_soft.yaml
```

Установка софта на сервер балансировщика нагрузки

1)Создайте в каталоге "~/app/ansible" файл "install_haproxy.yaml";

2)Добавьте в файл следующие строки

```
[ansible@ansible:~/app/ansible$ touch install_haproxy.yaml
[ansible@ansible:~/app/ansible$ cat <<END>> install_haproxy.yaml
> - name: Install worker soft
  hosts: load_balancer
  become: yes

  tasks:
    - name: Install haproxy
      apt:
        name: haproxy
        state: latest
        update_cache: yes
    - name: Stop haproxy
      sysvinit:
        name: haproxy
        state: stopped
    - name: Backup original config
      copy:
        src: /etc/haproxy/haproxy.cfg
        remote_src: yes  # Говорим, что источник тоже на удалённой машине
        dest: /etc/haproxy/haproxy.cfg_bup
```

Проверка работоспособности приложения

1) Откройте браузер на основной операционной системе, или на другом устройстве подключённом к той же сети, и введите ip-адрес сервера "load_balancer"

2)Обновите несколько раз страницу и попробуйте воспользоваться функционалом приложения.

3)Зайдите на страницу "/haproxy_stats" (порт 81) и введите учётные данные.

4)Изучите статистику распределения запросов к рабочим серверам столбец **Sessions** -> **Total** в нижней таблице. Количество запросов ко всем серверам должно быть примерно одинаковым, т.к. установлена политика балансировки roundrobin (карусель).

Масштабирование приложения

1)Перейдите на виртуальную машину с docker compose и остановите запущенные контейнеры;

```
[ansible@ansible:~/app/ansible$ docker compose down
[+] Running 3/2
 ✔ Container ansible-load_balancer-1  Removed                    1.0s
 ✔ Container ansible-db-1             Removed                    1.0s
 ! Network ansible_outside            Resourc...                 0.0s
ansible@ansible:~/app/ansible$
```

2)Добавьте в файл compose.yaml дополнительные сервера, таки образом, чтобы *рабочих* серверов было 8;

3)Запустите их при помощи docker compose up -d. Все машины будут чистыми с настройками по умолчанию;

```
ansible@ansible:~$ docker-compose up -d
```

4)Добавьте новые сарваера в файл hosts (учтите, что теперь там снова пользователь root), запустите начальное конфигурирование и установку всего необходимого софта;

5)В конфиге балансировщика, в раздел "backend" добавьте новые рабочие сервера;

```
> backend webservers
    balance roundrobin
```

6)Протестируйте работоспособность приложения, можно просто обновить вкладку раз 20-30;

7)Зайдите на страницу "/haproxy_stats" и проверьте, что все сервера работают и загружены равномерно.

Вывод: сегодня на практическом занятии по дисциплине «Современные технологии программирования» я ознакомился на практике с инструментом для удаленного управления конфигурациями Ansible.

Контрольные вопросы

1)Каким образом команде absible-playbook можно указать файл с серверами (hosts)? Перечислите несколько способов.

- Через параметр -i или --inventory:

```
ansible-playbook -i hosts.txt playbook.yml
```

- Указав путь к файлу в инвентарном файле (обычно inventory или `hosts`) в playbook:

```
- hosts: all
  inventory: hosts.txt
  tasks:
    ...
```

- Используя переменную окружения ANSIBLE_INVENTORY:

```
export ANSIBLE_INVENTORY=hosts.txt
ansible-playbook playbook.yml
```

2)Какое программное обеспечение нужно установить на linux севера, которыми планируется управлять при помощи ansible, при условии, что доступ по ssh уже есть? В вопросе рассматривается общий случай, без учёта специфических требований отдельных модулей.

Для управления серверами через Ansible на Linux серверах обычно требуется установить только Python. Большинство модулей Ansible работают с базовым Python, который уже установлен на большинстве Linux дистрибутивов. Специфические модули могут требовать дополнительные зависимости, но в общем случае Python является единственной необходимой установкой.

3)Что в логах выполнения playbook-а показывает значение "changed" который находится в разделе "PLAY RECAP"?

Значение "changed" в разделе "PLAY RECAP" указывает, что в ходе выполнения playbook'а было выполнено изменение на хостах. Это значит, что Ansible

обнаружил расхождение между желаемым состоянием, описанным в playbook'e, и фактическим состоянием на серверах, и внёс необходимые изменения.

4)Напишите ad hoc команду ansible, которая на хостах состоящих в группе "stark_industries" создаст пользователя "thor", который состоит **только** в группах "avengers" и "asgard" с паролем "Strongest Avenger".
Вот ad hoc команда Ansible, которая создаст пользователя "thor" в группах "avengers" и "asgard" с паролем "Strongest Avenger" на хостах из группы "stark_industries":

```
ansible stark_industries -m user -a "name=thor groups=avengers,asgard password=Strongest\ Avenger"
```

5)Напишите playbook выполняющий сценарий с названием "snap", который на серверах из группы "universe" из каталога "/lifeforms" удалит половину файлов. Известно, что файлов чётное количество и при этом файлов заканчивающихся на 1 столько же, сколько и файлов заканчивающихся на 2.

удаляет половину файлов из каталога "/lifeforms" на серверах из группы "universe":

```
- name: Snap
  hosts: universe
  tasks:
    - name: Delete half of the files
      file:
        path: /lifeforms/{{ item }}
        state: absent
      loop: "{{ files|shuffle|slice(0, (files|length)//2)|list }}"
```

```
  vars:
    files: "{{ lookup('fileglob', '/lifeforms/*') }}"
```

Этот playbook использует модуль file для удаления файлов, выбирая половину из них случайным образом с помощью фильтров shuffle и slice.