3.4.2

Установка Ansible

Создаем файл **Vagrantfile** с таким содержимым.

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

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
 config.vm.define "controlnode" do |controlnode|
   controlnode.vm.box = "ubuntu/focal64"
   controlnode.vm.hostname = "controlnode"
   controlnode.vm.network "private_network", ip: "192.168.50.4"
   controlnode.vm.synced_folder "./ansible","/home/vagrant/ansible"
   controlnode.vm.provision "shell", inline: <<-SHELL
     sed -i 's/ChallengeResponseAuthentication no/ChallengeResponseAuthentication
yes/#g' /etc/ssh/sshd_config
     service ssh restart
   SHELL
 end
 # Every Vagrant development environment requires a box. You can search for
 # boxes at https://vagrantcloud.com/search.
 config.vm.define "server" do |server|
  server.vm.box = "ubuntu/focal64"
  server.vm.hostname = "server"
  server.vm.network "private_network", ip: "192.168.50.5"
  server.vm.provision "shell", inline: <<-SHELL
    sed -i 's/ChallengeResponseAuthentication no/ChallengeResponseAuthentication
yes/#g' /etc/ssh/sshd_config
    service ssh restart
  SHELL
 end
end
```

Подсмотреть ещё можно из нашего репо `ansible_base\config_files\2.lemp-ansible\writing_playbook\Vagrantfile`

Тут же создаём папку **ansible**

```
mkdir ansible
```

Запускаем создание машин

```
vagrant up
```

После запуска проверяем их статус

```
vagrant status

Current machine states:

controlnode               running (virtualbox)
server                    running (virtualbox)
```

Подключаемся к контрол ноде.

```
ssh vagrant@192.168.50.4
```

Вводим пароль **vagrant**

И мы подключились. Можем подключаться как раньше, только указывать наименование ноды.

```
vagrant ssh controlnode
```

Добавим архив **ansible**

```
sudo apt-add-repository ppa:ansible/ansible

...
http://ansible.com/
 More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
Press [ENTER] to continue or Ctrl-c to cancel adding it.
```

Соглашаемся, жмём **Enter**.

```
Get:45 http://archive.ubuntu.com/ubuntu focal-backports/multiverse amd64 c-n-f
Metadata [116 B]
Fetched 19.8 MB in 21s (950 kB/s)
Reading package lists... Done
```

Готово! Теперь можем установить ansible.

```
sudo apt install ansible -y
```

Проверяем установку.

```
ansible --version

ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/vagrant/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, Jun  2 2021, 10:49:15) [GCC 9.4.0]
```

Едем дальше!

3.5.2

Пишем плейбук: hosts

Получаем список хостов для работы ansible.

```
ansible all --list-hosts

[WARNING]: provided hosts list is empty, only localhost is available. Note that the
implicit localhost does not match 'all'
  hosts (0):
```

Пусто, потому что команда по умолчанию берёт данные из этого файла **/etc/ansible/hosts**. Там всё закомментировано. Можете проверить.

```
cat /etc/ansible/hosts
```

Мы этот файл менять не будем, а создадим новый файл **hosts.ini** в папке **ansible**. С IP-адресом server-a.

```
192.168.50.5
```

Команде задаём ключ **-i** (INVENTORY) и путь, по которому нужно взять новый файл с хостами.

```
ansible all --list-hosts -i ./ansible/hosts.ini

  hosts (1):
    192.168.50.5
```

Попробуем теперь выполнить простую команду **ping** на хостах.

```
cd ansible/

ansible all -i hosts.ini -m ping

192.168.50.5 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Warning: Permanently added
'192.168.50.5' (ECDSA) to the list of known hosts.\r\nvagrant@192.168.50.5:
Permission denied (publickey,keyboard-interactive).",
    "unreachable": true
}
```

Просит ввода пароля. Чтобы решить данную ошибку, добавляем **--ask-pass** который спросит пароль.

```
ansible all -i hosts.ini -m ping --ask-pass

SSH password:
192.168.50.5 | FAILED! => {
    "msg": "to use the 'ssh' connection type with passwords, you must install the
sshpass program"
}
```

Нельзя использовать пароли без установки **sshpass**. Установим.

```
sudo apt install sshpass
```

Пробуем снова.

```
ansible all -i hosts.ini -m ping --ask-pass

SSH password:
192.168.50.5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Присвоим в нашем файле **hosts.ini** специфичные параметры для server, а именно user\password. Чтобы мы могли больше не вводить пароль при выполнении плейбука.

Редактируем **hosts.ini**

```
192.168.50.5 ansible_user=vagrant ansible_password=vagrant
```

Убираем ключ **--ask-pass** и пробуем.

```
ansible all -i hosts.ini -m ping
```

```
192.168.50.5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Теперь настроим плейбук чтобы установить LEMP.

3.5.4 Пишем плейбук: nginx

Создадим файл **playbook.yml** на вашей рабочей машине в папке **ansible**, где создавали hosts.ini.

Внесём это содержание:

```
##################
# LEMP PLAYBOOK #
##################
---
- hosts: "all" # применяем команды ко всем хостам из inventory
  become: true # становимся привилегированным пользователем
  tasks: # список тасков
  # NGINX
  - name: "Install nginx via apt" # имя, обязательно для любой таски из плейбука, должно содержать описание того что эта таска делает
    ansible.builtin.apt: # имя модуля, этот модуль устанавливает софт на хосте с помощью пакета apt
      name: "nginx" # имя пакета для установки
      state: "latest" # установить последнюю версию пакета, в отличие от present, если пакет уже установлен в системе, попробует его обновить
      update_cache: true # вызвать apt update перед установкой, чтобы убедиться в наличии самой свежей базы данных о пакетах

  - name: "Delete /var/www/html folder"
    ansible.builtin.file: # этот модуль работает с файлами и папками на удаленной машине
      path: "/var/www/html" # путь к файлу/папке на удаленной машине
      state: "absent" # перевести папку в состояние absent - то есть удалить ее

  - name: "Copy our landing to /var/www/html folder"
    ansible.builtin.copy: # этот модуль копирует файлы/директории с контрольной ноды на хосты
      src: "files/html" # путь к файлу/директории для копирования на контрольной ноде
      dest: "/var/www/" # путь для вставки на удаленной машине, вставка папок ВСЕГДА будет осуществляться внутрь папки, ансибл никогда не будет подменять папки или переименовывать их
      owner: "vagrant" # пользователь владелец файла/директории на удаленной машине
      group: "vagrant" # группа владелец файла/директории на удаленной машине
      mode: "0644" # права доступа к файлу/директории
```

Сохраняем. В этой же папке **anisble** создаем папку **files/html** и бросаем сюда файлы из тестового лендинга. Что вы уже загружали из просторов интернета в ручной настройки LEMP или наш одностраничный слёрмовский сайт, по пути - `ansible_base\config_files\2.lemp-ansible\writing_playbook\ansible\files\html`

Запускаем плейбук на controlnode.

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

Если плейбук отыграл успешно, то заходим сюда и любуемся - http://192.168.50.5/

Пойдёмте прикручивать mysql.

3.5.6

Пишем плейбук: mysql

Теперь добавляем таски для установки **mysql** в playbook.yml.

```
  # MYSQL
  - name: "Install mysql via apt"
    ansible.builtin.apt:
      name: "mysql-server"
      state: "latest"
      update_cache: true

  - name: "Install pymysql via apt"
    ansible.builtin.apt:
      name: "python3-pymysql" # необходим для работы модулей mysql
      state: "latest"
      update_cache: true

  - name: "Set up root user"
    community.mysql.mysql_user: # модуль для управления пользователями
      name: "root" # имя пользователя для редактирования, обратите внимание на
единообразие параметров модулей
      password: "password" # пароль пользователя для редактирования
      login_user: "root" # имя пользователя, под которым вносим изменения
      login_password: "password" # пароль пользователя под которым редактируем
      check_implicit_admin: true # сначала попробует зайти без пароля
      login_unix_socket: "/var/run/mysqld/mysqld.sock" # если mysql работает через
сокет, вместо коннекта через порт, то нужен этот путь

  - name: "Remove anonymous users"
    community.mysql.mysql_user:
      name: "" # анонимные пользователи не имеют имени
      state: "absent" # как и с модулем file - удалить
      login_user: "root"
      login_password: "password"

  - name: "Remove test database"
    community.mysql.mysql_db: # модуль для управления базами данных
      name: "test" # тут это имя базы
      state: "absent" # как и с модулем file - удалить
      login_user: "root"
      login_password: "password"
```

Для работы модуля **community.mysql** нужно будет его установить.

```
ansible-galaxy collection install community.mysql

Process install dependency map
Starting collection install process
Installing 'community.mysql:2.1.1' to
'/home/vagrant/.ansible/collections/ansible_collections/community/mysql'
```

При установке модуля может возникнуть данная ошибка:

```
ansible-galaxy collection install community.mysql

Process install dependency map
ERROR! Unknown error when attempting to call Galaxy at
'https://galaxy.ansible.com/api/': <urlopen error [Errno -3] Temporary failure in
name resolution>
```

Решение этой проблемы, обновление каталога сертификатов.

```
sudo update-ca-certificates --fresh
export SSL_CERT_DIR=/etc/ssl/certs
```

Запускаем плейбук.

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

Осталось прикрутить последний элемент.

3.5.8

Пишем плейбук: php

Добавляем таски для php в файл playbook.yml. Проверить корректность плейбука можете тут - `ansible_base-test\ansible_base\config_files\2.lemp-ansible\writing_playbook\ansible\playbook.yml`

```
  # PHP
  - name: "Installing php fpm and php mysql"
    ansible.builtin.apt:
      name: "{{ item }}" # здесь мы будем перечислять пакеты для установки, поэтому
вместо имени здесь плейсхолдер
      state: "latest"
      update_cache: true
    with_items: # а вот тут уже и перечисляем, ansible сам их подставит вместо {{
name }}
      - "php-fpm"
      - "php-mysql"

  - name: "Copy php files to /var/www"
    ansible.builtin.copy:
      src: "files/test-php/php_test"
      dest: "/var/www/"
      owner: "vagrant"
      group: "vagrant"
      mode: "0644"

  - name: "Copy nginx config for php testing"
    ansible.builtin.copy:
      src: "files/test-php/nginx.conf"
      dest: "/etc/nginx/sites-available/php_test.conf"
      owner: "vagrant"
      group: "vagrant"
      mode: "0644"

  - name: "Link folder"
    ansible.builtin.file:
      src: "/etc/nginx/sites-available/php_test.conf"
      dest: "/etc/nginx/sites-enabled/php_test"
      state: "link" # здесь создаем симлинк

  - name: "Reload nginx"
    ansible.builtin.service:
```

```
      name: "nginx"
      state: "reloaded"
```

В этой же папке **ansible** создаем папку **files\test-php**, куда сохраняем файл **nginx.conf**. Так же создаем папку **files\test-php\php_test**, куда сохраняем файлы **index.php\test-connection.php.** Эти все файлы идентичны, которые создавали в прошлой теме. Вы можете скопировать их из нашего репо, по пути - `ansible_base\config_files\2.lemp-ansible\writing_playbook\ansible\files\test-php`

Запустим плейбук на controlnode с определенного таска, чтоб не проигрывались все таски.

```
ansible-playbook playbook.yml -i hosts.ini --start-at-task "Installing php fpm and php mysql"
```

После того как успешно отыграет плейбук, редактируем hosts своего ПК. Теперь у настраиваемой машины есть собственный IP-адрес.

```
127.0.0.1 php-test.com
```

на


```
192.168.50.5 php-test.com
```

Пробуем зайти http://php-test.com и http://php-test.com/test-connection.php

Ура! Теперь мы умеем настраивать LEMP через Ansible, не бегая на настраиваемый сервер.

3.6.2

Рефакторим с помощью ролей

Попробуйте отрефакторить полученный итоговый код с помощью ролей.

Отрефакторенная структура плейбука:

```
##################
# LEMP PLAYBOOK #
##################
---
- hosts: "all"
  become: true
  roles:
  - nginx # все таски файла main.yml по пути %директория ansible%/roles/nginx/tasks/
  - mysql # все таски файла main.yml по пути %директория ansible%/roles/mysql/tasks/
  - php # все таски файла main.yml по пути %директория ansible%/roles/php/tasks/
  tasks:
  - name: "Reload nginx" # этот таск не относится к ролям напрямую, поэтому вынесен после выполнения всех ролей
    ansible.builtin.service:
      name: "nginx"
      state: "reloaded"
```

Роли находятся в соответствующих подпапках.

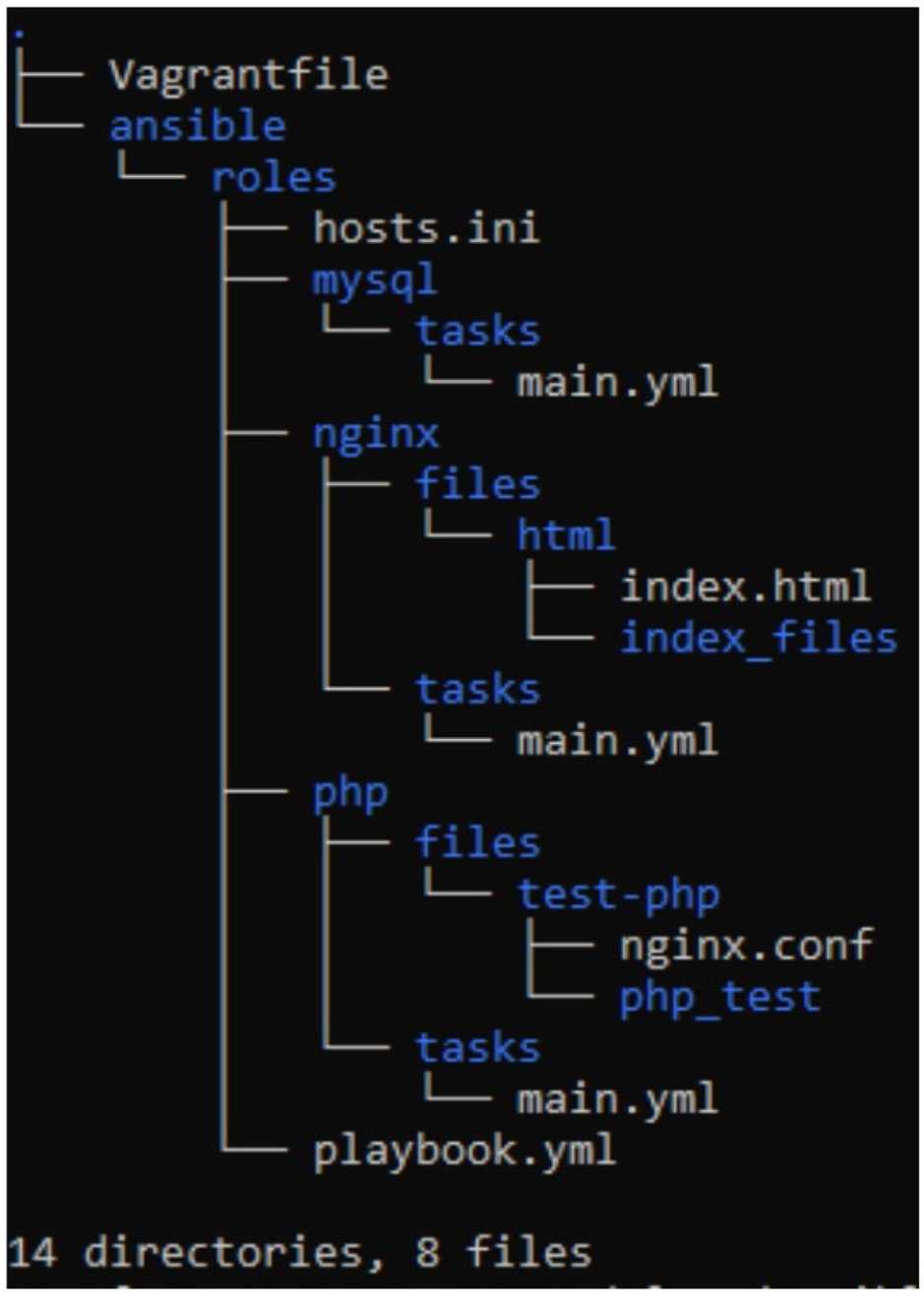
Можно использовать команду `init` для инициализации базовой структуры новой роли, что экономит время на создание различных каталогов и файлов main.yml, для которых требуется роль

```
$ ansible-galaxy init role_name
```

В скачанном вами общедоступном репо есть готовый отрефакторенный код - `ansible_base\config_files\2.lemp-ansible\refactor_with_roles` (https://gitlab.slurm.io/edu/ansible_course/-/tree/master/2.lemp-ansible/refactor_with_roles)

Пробуем зайти - http://php-test.com и http://php-test.com/test-connection.php

Готовый код, после рефакторинга для сравнения со своим - https://github.com/Slurmio/ansible_base/tree/main/config_files/2.lemp-ansible/refactor_with_roles/ansible/roles



3.7.2

В следующих лекциях мы подробнее остановимся на Ansible galaxy. Чтобы больше погрузиться в контекст, познакомьтесь с популярными ролями в каталоге:

https://galaxy.ansible.com/

3.10

Новый сайт

Загрузите веб-сайт с новой конфигурацией **nginx**, который отвечает на домен http://mytestsite.com

Для сайта можете использовать HTML код:

```
<!doctype html>

<html lang="en">
<head>
  <meta charset="utf-8">

  <title>The HTML5 Herald</title>
  <meta name="description" content="MY site">
  <meta name="author" content="SitePoint">

</head>

<body>
   <h1>Choosing an Apple</h1>
<section>
    <h2>Introduction</h2>
    <p>This document provides a guide to help with the important task of choosing the
correct Apple.</p>
</section>

<section>
    <h2>Criteria</h2>
    <p>There are many different criteria to be considered when choosing an Apple –
size, color, firmness, sweetness, tartness...</p>
</section>
</body>
</html>
```

Сколько тасок Ansible вам понадобилось? Удобно ли использовать такие подходы в работе?

3.10.2 Роли для CentOS

Напишите роли для установки LEMP стека на CentOS.

Напоминаем, что вместо **apt** используется **yum**, соответственно, модуль Ansible будет называться **ansible.builtin.yum**. Также CentOS7 по умолчанию ставится php5.4 вместо 7, не акцентируйте внимание на этом.

Полезны ли такие адаптации в реальных рабочих процессах?