



# TENSOR

## Введение в администрирование Linux: Ansible Семинар #1 “Ansible: Кто такой? Где применяется?”

Пантелеев Александр



# Введение в администрирование Linux: Ansible

## План занятия

- Общее описание
- Установка и конфигурация
- Проверка работы
- YAML
- Структура сценария
- Практика

*Вопросы просьба задавать в конце подтемы.*

# Ansible: кто такой, где применяется?

## Что такое Ansible?



Ansible – это единая система управления конфигурациями и развертыванием приложений.

Основные особенности:

- Декларативный подход и приведение системы к желаемому состоянию.
- [Push-based](#) управление конфигурациями (Управляющий сервер рассылает необходимые настройки)
- Принцип идемпотентности (многократное повторение действия эквивалентно однократному)
- Написан на Python
- Модульность

# Ansible: Установка, первичная конфигурация

yum:

```
# yum install centos-release-ansible-29
```

```
# yum install ansible-2.9.14
```

pip:

```
$ python3 -m venv ./ansivenv && source ./ansivenv/bin/activate
```

```
$ python3 -m pip install --upgrade pip && python3 -m pip install ansible==2.9.14
```

## Установка из исходного кода:

[Ссылка с инструкцией по установке](#)

# Ansible: Установка, первичная конфигурация

## Файл hosts

Файл hosts позволяет описать inventory (реестр хостов), с которым в будущем будет работать Ansible. Имена хостов описываются в виде DNS-имен либо IP адресов хостов.

Ansible по умолчанию хранит реестр хостов в файле `/etc/ansible/hosts`.

[Ссылка на документацию](#)

```
(venv) [ansible@ansible-control ansible-course]$ cat /etc/ansible/hosts
[nodes]
node1  ansible_host=ansible-node1.localdomain
```

```
1 lb1 ansible_host=lb1.proxy.localnet ansible_port=222
2 lb2 ansible_host=lb2.proxy.localnet ansible_port=222
3 web1 ansible_host=192.168.0.3
4 web2 ansible_host=192.168.0.4
5 web3 ansible_host=192.168.0.5
6 web4 ansible_host=192.168.0.6
7 db-a ansible_host=192.168.0.50 ansible_user=db-admin
8 db-b ansible_host=192.168.0.51 ansible_user=db-admin
9
10 [loadbalancers]
11 lb1
12 lb2 lb_level="round-robin"
13
14 [webservers]
15 web[1:4]
16
17 [webservers:vars]
18 http_port=80
19 https_port=443
20
21 [[databases]]
22 db-[a:b]
23
24 [production]
25 web[1:2]
26 db-1
```

# Ansible: Файл hosts

Имя параметра	Значение по умолчанию	Описание
ansible_host	Имя хоста	Имя хоста или ip адрес
ansible_port	22	Порт для подключения по протоколу SSH
ansible_user	Имя пользователя, запускающего сценарий	Пользователь для подключения по протоколу SSH
ansible_password	(нет)	Пароль для подключения по протоколу SSH
ansible_python_interpreter	/usr/bin/python	Путь к интерпретатору Python на хосте

# Ansible: Проверка работы

Проверяем установку Ansible через следующую команду:

```
$ ansible --version
```

Ansible позволяет использовать инструмент командной строки для запуска произвольных команд на удаленных серверах.

Проверим работу Ansible, запустив задачу:

```
$ ansible nodes -m ping
```

# Ansible: Установка, первичная конфигурация

## Файл ansible.cfg

Основной конфигурационный файл – ansible.cfg.

Ansible будет искать файл ansible.cfg в следующих директориях в указанном порядке:

1. Файл, указанный в переменной окружения ANSIBLE\_CONFIG
2. ./ansible.cfg (Текущая директория)
3. ~/.ansible.cfg (Домашняя директория пользователя)
4. /etc/ansible/ansible.cfg

Посмотреть список доступных опций можно командой:

```
$ ansible-config list
```

```
1 # config file for ansible -- https://ansible.com/
2 # =====
3
4 # nearly all parameters can be overridden in ansible-playbook
5 # or with command line flags. ansible will read ANSIBLE_CONFIG,
6 # ansible.cfg in the current working directory, .ansible.cfg in
7 # the home directory or /etc/ansible/ansible.cfg, whichever it
8 # finds first
9
10 [defaults]
11
12 # some basic default values...
13
14 #inventory      = /etc/ansible/hosts
15 #library        = /usr/share/my_modules/
16 #module_utils   = /usr/share/my_module_utils/
17 #remote_tmp     = ~/.ansible/tmp
18 #local_tmp      = ~/.ansible/tmp
19 #plugin_filters_cfg = /etc/ansible/plugin_filters.yml
20 #forks          = 5
21 #poll_interval  = 15
22 #sudo_user      = root
23 #ask_sudo_pass  = True
24 #ask_pass       = True
25 #transport      = smart
26 #remote_port    = 22
```

[Ссылка на документацию](#)

[Ссылка на конфиг со значениями по умолчанию](#)



# Ansible: Принцип работы

Задача — проверить соединение с конечным хостом.

Выполняя задачу, Ansible проделает следующие действия:

1. Сгенерирует скрипт на Python
2. Скопирует скрипт на хосты
3. Запустит скрипт на хостах
4. Дождется, пока скрипт завершится на всех хостах

# Ansible-playbook: начало

При работе с Ansible большая часть времени уделяется написанию сценариев. Сценарием называется файл, описывающий порядок управления конфигурациями.

Все сценарии Ansible пишутся на YAML. YAML — это язык для хранения информации в формате понятном человеку. Перед написанием сценария кратко пробежимся по основным понятиям YAML, наиболее важных для написания сценариев.

# Структура YAML-файла

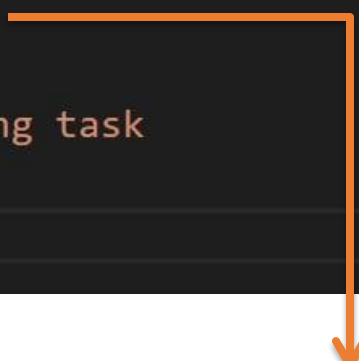
- Файлы YAML начинаются с трех дефисов, обозначающих начало документа
- Комментарии начинаются со знака #
- 1 перенос — 2 пробела
- Основные типы:
  - Строки
  - Числа
  - Булевы выражения
  - Дата
  - Null
  - Списки
  - Словари

```
1 ---
2 # Комментарий на языке YAML
3 string_var: String content
4 string_var2: "String content"
5 multiline_string_var: >
6   americ
7   an football
8 int_var: 42
9 float_var: 42.0
10 exponential_var: 0.42e+2
11 hex_var: 0x2A
12 string_number_var: "42"
13 bool_true_var: true
14 bool_false_var: false
15 date_time_var: 2015-10-21T07:28:00.42+02:00
16 simple_date_var: 2015-10-21
17 undefined_var: null
18 my_list:
19   - var1
20   - var2
21   - var3
22 my_list2: [ var1, var2, var3 ]
23 my_dict:
24   key1: value1
25   key2: value2
26   key3: value3
27 my_dict2: { key1: value1, key2:value2, key3:value3 }
28
```

# Пишем простой сценарий

- ping.yaml — сценарий (Playbook), содержит в себе список операций (Play).
- Операция объединяет множество хостов из инвентори (файл hosts) и список задач, исполняемых на этих хостах.
- Задачи содержат в себе модули. Модуль принимает параметры из задачи и передает их в скрипт на Python.
- Список параметров и примеры можно посмотреть в документации по модулю на сайте [docs.ansible.com](https://docs.ansible.com) или в консоли, вызвав команду `$ ansible-doc *имя модуля*`
- Команда для запуска сценария:  
`$ ansible-playbook ping.yaml`

```
ping_test > ! ping.yaml
1  ---
2  - name: ping remote server
3    hosts: nodes
4    tasks:
5      - name: ping task
6        ping:
7
```



```
(venv) [ansible@ansible-control ansible-course]$ cat /etc/ansible/hosts
[nodes]
node1  ansible_host=ansible-node1.localdomain
```

# Пишем простой сценарий

- В vars указываются переменные, используемые в сценарии
- Для использования файлов с переменными используется vars\_files
- Шаблоны позволяют подставлять значения из переменных.

[Документация по модулю copy](#)

[Документация по модулю template](#)

```
1 ---
2 - name: copy files to remote host
3   hosts: nodes
4   vars:
5     custom_var: ansible
6     user_home_dir: /home/ansible
7   vars_files:
8     - vars/main.yaml
9   tasks:
10    - name: copy file from files directory
11      copy:
12        src: sample_file
13        dest: "{{ user_home_dir }}/sample_file.txt"
14        owner: ansible
15        group: ansible
16        mode: 0644
17
18    - name: copy file from template
19      template:
20        src: templates/file_from_template.txt.j2
21        dest: "{{ user_home_dir }}/file_from_template.txt"
22        owner: ansible
23        group: ansible
24        mode: 0644
25
```

copy\_and\_templates > vars > ! main.yaml

1 ext\_custom\_var: 42

copy\_and\_templates > templates > file\_from\_template.txt.j2

```
1 custom variable is {{ custom_var }}
2 variable from external file is {{ ext_custom_var }}
3
```



# Основные файлы в директории со сценарием

- main.yaml (файл со сценарием)
- vars (директория с переменными, используемыми в сценарии)
- files (директория с файлами, используемыми в сценарии)
- templates (директория с шаблонами jinja, используемыми в сценарии)

```
▼ copy_and_templates
  ▼ files
    ≡ sample_file.txt
  ▼ templates
    ≡ file_from_template.txt.j2
  ▼ vars
    ! main.yaml
    ! main.yaml
```

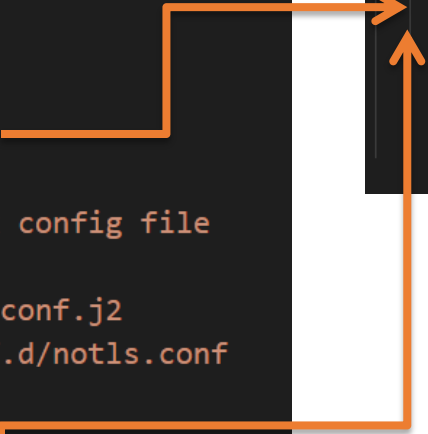
# Обработчики

- Обработчики выполняются только после завершения задач и только один раз, даже если было получено несколько.
- Обработчики выполняются в порядке следования в разделе handlers, а не в порядке следования уведомлений.

```
- name: copy main nginx config file
  copy:
    src: files/nginx.conf
    dest: /etc/nginx/nginx.conf
    owner: root
    group: root
    mode: 0644
  notify: restart nginx

- name: copy site's nginx config file
  template:
    src: templates/nginx.conf.j2
    dest: /etc/nginx/conf.d/notls.conf
    owner: root
    group: root
    mode: 0644
  notify: restart nginx
```

```
handlers:
  - name: restart nginx
    service:
      name: nginx
      state: restarted
```



[Сценарий nginx-no-tls](#)

[Ссылка на документацию](#)

# Управление повышением привилегий

Ansible позволяет выполнять задачи от имени привилегированного пользователя.

Опцию можно задать на следующих уровнях:

- Сценарий
- Блоки задач
- Отдельные задачи

```
339
340 [privilege_escalation]
341 #become=True
342 #become_method=sudo
343 #become_user=root
344 #become_ask_pass=False
345
```

```
1 ---
2 - name: Play with two tasks, one uses privilege escalation
3   hosts: nodes
4   become: false
5   gather_facts: false
6   tasks:
7     - name: This task needs privileges
8       file:
9         path: /root/file.txt
10        state: touch
11        mode: 0644
12        become: true
```

# Краткий итог

(Сценарий) Playbook содержит одну или несколько операций (Play). Операции связывают неупорядоченное множество хостов и упорядоченный список задач. Задачи запускают работу модуля. Каждая задача соответствует только одному модулю.



# Практика

В качестве практики поднимем на удаленном хосте Docker-контейнер с браузерной версией игры Doom или Mario на выбор.

Предварительно поставим коллекцию модулей для работы с докером

```
$ ansible-galaxy collection install community.docker
```

[Сценарий разворота контейнера с игрой](#)



# Домашнее задание

## **Задача 1:**

Проделать практику с разворотом контейнера.

## **Задача 2:**

Настроить Nginx на для работы по HTTPS

[Ссылка на Github-репозиторий](#)



Ознакомились с принципами  
работы Ansible.  
Научились писать сценарии  
на базовом уровне.

Вопросы?

