

Автоматизация администрирования инфраструктуры: **Ansible**



Артём
Поневин



Артем Поневин

Инженер

Luxoft



Модуль «Автоматизация администрирования инфраструктуры»

Цели модуля:

- познакомиться с системами управления конфигурациями и утилитами развертывания облачной инфраструктуры;
- изучить средства управления конфигурацией и зачем они нужны;
- научиться настраивать облачный сервер с помощью Terraform и сконфигурировать его с помощью Ansible.

Структура модуля

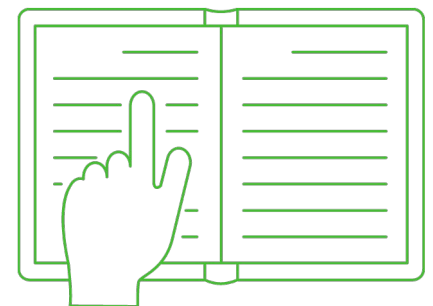
1. Ansible. Часть 1.
2. Ansible. Часть 2.
3. Terraform.
4. Подъем инфраструктуры в Yandex.Cloud.

Предисловие

На этом занятии мы узнаем:

- что такое IaC и какие преимущества он дает;
- что такое Ansible и какие задачи с его помощью можно решать.

По итогу занятия вы получите представление о возможностях Ansible, научитесь настраивать и запускать команды ad-hoc.



План занятия

1. [Предисловие](#)
2. [ИАС. Что такое Ansible?](#)
3. [Практика. Установка Ansible](#)
4. [Настройка Ansible](#)
5. [Настройка Ansible. Практика](#)
6. [Модули. Ad-hoc команды](#)
7. [Ad-hoc команды. Практика](#)
8. [Итоги](#)
9. [Домашнее задание](#)



IAC. Что такое Ansible?



IAC

Инфраструктура как код (англ. Infrastructure-as-Code; Iac) — это подход к управлению и описанию инфраструктуры через конфигурационные файлы, а не через ручное редактирование конфигураций на серверах. Процесс настройки инфраструктуры аналогичен процессу программирования ПО. Границы между написанием приложений и созданием сред для этих приложений стираются.

Не путать с SAAS, IAAS, PAAS.

Плюсы IAC

- Нет необходимости в ручной настройке;
- Скорость: настройка («поднятие») инфраструктуры занимает на порядок меньше времени;
- Воспроизводимость: поднимаемая инфраструктура всегда идентична;
- Масштабируемость: один инженер может с помощью одного и того же кода настраивать и управлять огромным количеством машин.



Configuration management systems

Системы управления конфигурациями — программы и комплексы, позволяющие централизованно управлять конфигурацией множества разнообразных разрозненных ОС и прикладного ПО, работающего в них. При работе с системами управления конфигурациями говорят об автоматизации инфраструктуры или об оркестрации серверов.

Преимущества Configuration management systems:

- использовать систему контроля версий для отслеживания любых изменений инфраструктуры;
- повторно использовать скрипты конфигурирования для нескольких серверных сред, например, для разработки, тестирования и производства;
- предоставлять сотрудникам общий доступ к скриптам конфигурирования для упрощения сотрудничества в стандартизированной среде разработки;
- упрощать процесс дублирования серверов для ускорения восстановления в случае сбоя системы.

Ведущие инструменты для управления конфигурацией



Chef	Puppet	SaltStack	Ansible
DSL	DSL(JSON)	YAML	YAML
Ruby	Ruby	Python	Python
клиент-сервер	клиент-сервер	клиент-сервер, безагентный	безагентный

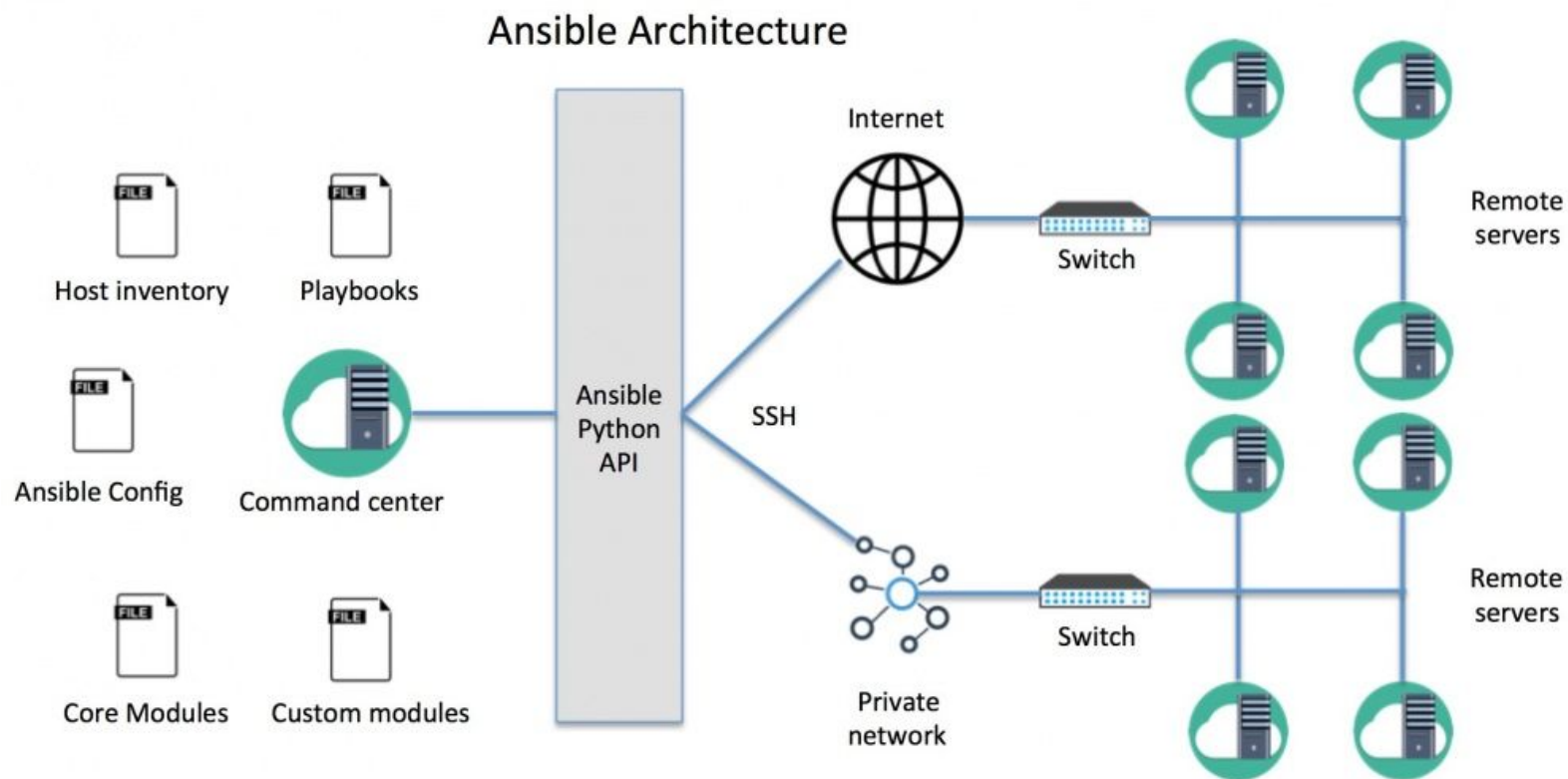
Ansible



Ansible — система управления конфигурациями. Позволяет централизованно управлять ПО, ОС и их настройками на Linux, Mac, Windows. Написан на Python, открытый исходный код, разработчик — Red Hat.

Ansible Architecture

Ansible отправляет команды на удаленные хосты через ssh.



Особенности Ansible

Безагентный — для работы (настройки) с управляемым узлом нет необходимости ставить на управляемый узел агента. Необходимых требований только два: работающий ssh-сервер, python версии 2.6 и выше.

Идемпотентность — независимо от того, сколько раз вы будете запускать playbook, результат(конфигурация управляемого узла) всегда должен приводить к одному и тому же состоянию.

Push-model — изменения конфигураций “заталкиваются” на управляемые узлы. Это может быть минусом.

Основные термины Ansible

Узел управления — устройство с установленным и настроенным Ansible. Может быть ваш ноутбук или специальный узел в сети (подсети) выделенный для задач управления.

Управляемые узлы — узлы, конфигурация которых выполняется.

Файлы инвентаризации(inventory) — файл(ы), в которых перечислены управляемые узлы: *.ini, *.yaml или динамический.

Модули(modules) отвечают за действия, которые выполняет Ansible. Иными словами, инструментарий Ansible.

Основные термины Ansible

Задачи (tasks) — отдельный элемент работы, которую нужно выполнить. Могут выполняться самостоятельно или в составе плейбука.

Плейбук (playbook) состоит из списка задач и/или других директив, указывающих на то, какие действия и где будут производиться.

Обработчики (handlers) — элемент, который служит для экономии кода и способен перезапускать службу при его вызове.

Роли (roles) — набор плейбуков и других файлов, которые предназначены для выполнения какой-либо конечной задачи. Также упрощают, сокращают код и делают его переносимым.



Практика. Установка Ansible.

Установка Ansible

Устанавливаем Ansible:

```
yum install ansible/apt install ansible
```

Правим (при необходимости) конфигурационный файл Ansible:

```
vim /etc/ansible/ansible.cfg
```

Правим файл inventory по умолчанию или создаем свой:

```
vim /etc/ansible/hosts
```

Смотрим версию и другие переменные запуска Ansible:

```
ansible --version
```



Настройка Ansible

Inventory

Инвентарный файл — это файл, в котором описываются устройства, к которым будет подключаться и управлять Ansible. Может быть в формате INI или YAML (или быть динамически конфигурируемым какой-либо вычислительной системой). Две группы по умолчанию: all и ungrouped.

Inventory

Пример:

```
[servers]
192.168.1.[1:5]

[another_servers]
my[A:C].example.com

[all_servers:children]
servers
another_servers
```

Команды:

```
ansible all -m ping --list-hosts — вывести список хостов
ansible-playbook --list-hosts — вывести список хостов для playbook
```

Ansible.cfg

— это основной конфигурационный файл. Может храниться:

- `ANSIBLE_CONFIG` (переменная окружения),
- `ansible.cfg` (в текущем каталоге),
- `~/.ansible.cfg` (в домашнем каталоге пользователя),
- `/etc/ansible/ansible.cfg` (можно брать за образец для внесения правок).

`ansible --version` — покажет какой конфигурационный файл будет использоваться

Ansible.cfg

В конфигурационном файле можно задавать множество параметров, например:

```
[defaults]
inventory = inventory.ini # расположение файла inventory
remote_user = ansible # пользователь, которым подключаемся по ssh
gathering = explicit # отключает сбор фактов
forks = 5 # количество хостов, на которых текущая задача выполняется
одновременно

[privilege_escalation]
become = True # требуется повышение прав
become_user = root # пользователь, под которым будут выполняться
задачи
become_method = sudo # способ повышения прав
```


Параметры

Большинство настроек также может задаваться или переопределяться во время выполнения команд через параметры.

Примеры:

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

 – вручную указывает файл инвентори

```
ansible all -m ping -e "ansible_user=vagrant
```

```
ansible_ssh_pass=vagrant"
```


 – вручную задает удаленного пользователя и пароль

```
ansible all -u vagrant -m ping
```

 – вручную задает удаленного пользователя

```
ansible web* -m ping
```

 – задает список хостов к выполнению через регулярные выражения



Настройка Ansible. Практика.

Настройка подключения к VM Vagrant

1. В VM создается второй сетевой интерфейс, который включается в мост с сетевым интерфейсом в локальную сеть.
2. В VM в конфигурационном файле `sshd` выставляется параметр `PasswordAuthentication yes`. Сервис перезапускается.
3. С помощью команды `ssh-copy-id vagrant@YOURIP` копируется ключ на созданные VM(для беспарольного входа).
4. Создается директория под проект. Внутри директории создаются необходимые конфигурационные файлы.
5. Проверка подключения: `ansible all -m ping`.



Модули. Ad-hoc команды.

Ansible modules

Модули — это небольшая программа, входящая в поставку Ansible, принимающая на вход значения и выполняющая работу на целевых хостах. Фактически вся работа происходит с использованием модулей. Можно самостоятельно писать модули и расширять возможности Ansible.

Примеры:

```
ansible-doc -l  
ansible-doc shell  
ansible all -m ping
```

Ansible ad-hoc

Ad-Hoc команды — это самый быстрый способ начать использовать Ansible. Для запуска Ansible в режиме ad-Hoc не нужно писать плейбуки, достаточно помнить минимальный синтаксис.

Примеры:

```
ansible all -m ping  
ansible all -m command -a "cat /etc/hosts"
```

Часто используемые модули

	Название модуля	Какое действие выполняет
1	ping	Позволяет проверить доступность хостов для работы через Ansible
2	service	Позволяет управлять работой служб(демонов) на хостах
3	command	Позволяет запустить команду без использования окружения
4	copy	Позволяет копировать файлы
5	lineinfile	Позволяет заменять(добавлять) строки в текстовых файлах
6	debug	Позволяет выводить отладочные сообщения
7	git	Позволяет работать с git-репозиториями



Ad-hoc команды. Практика



Итоги

Итоги

Сегодня мы:

- рассмотрели подход к организации инфраструктуры IAC;
- познакомились с системами управления конфигурациями и их возможностями;
- научились настраивать Ansible для работы с управляемыми хостами;
- научились использовать модули в ad-hoc командах.





Домашнее задание

Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

Дополнительные материалы

- [Официальная документация](#)
- Книги и видеокурсы от Sander van Vugt
- Курсы и книги от Red Hat(EX294)

**Задавайте вопросы и
пишите отзыв о лекции!**

Артем Поневин