**Глава 16. DevOps Развертывание серверной среды**

***DevOps*** – организационная структура проекта, которая помогает администраторам и разраотчикам ускоритьь и автоматизировать циклы разработки продуктов.

***Инструменты оркетсровки*** – позволяют точно составить сценарий поведения инфраструктуры для достижения желаемых состояний за счет автоматизации.

Управление осуществляется по группам, используя playbook модули (Ansible), которые представляют собой предопрделенные последовательности команд, выполняемых на хосте. Сценарии playbook управляют ресурсами через роли.

***Обработчики*** – действия, инициируемые событиями.

***Роли*** – связанные ресрурсы одного проекта.

Инструменты аркетсровки:

* Puppet
* Chef
* Ansible (подходит больше всего для системных администраторов)
* Salt

**1) Установка и настройка:**

**1.1 Установка**

Нужно убедиться, что версии Python на сервере и хостах обновлены:

* apt install python
* python –version

Для установки Ansible на сервер потребуется репозиторий:

* apt install software-properties-common
* add-apt-repository ppa:ansible/ansible (EPEL, Extras для CentOS6;7)

Установка Ansible:

* apt update
* apt install ansible

После чего все готово, на хостах нужен лишь Python, Ansible на них не ставится

**1.2 Настройка беспоролного доступа**

Ansible предпочитает работу через SSH. Поэтому можно выполнять аутентификацию из :

* Командной строки (ввод пароля)
* С использованием ключей (ПРЕДПОЧТИТЛЕЬНО)
  + ssh-keygen
  + ssh-copy-id -i .ssh/id\_rsa.pub [ubuntu@10.0.3.142](mailto:ubuntu@10.0.3.142) -копируем на хост

**1.3 Организация Ansible-хостов**

Ansible получает информацию о том, какими хостами управлять из файла /etc/ansible/hosts, который может содержать как просто список IP- адресов или доменов, так и список, разделенный на группы хостов. Правила, описанные в этом файле могут быть разнообразно написаны (скоращенно и т.д. тут нужно знать)

**1.4 Тестирование подключения:**

Чтобы удостовериться, что все правильно настроено, можно заставить Ansible связаться с хостами, указанными в файле hosts:

* ansible all -m ping (all-для всего в hosts, -m загружает модуль ping).
* Ansible webservers -m ping (для группы webservers)

Если все успешно прошло, можно попробовать запустить простые команды удаленно:

* ansible all -a “cp /etc/group /home/user” (копируем /etc/group в домашний каталог каждого хоста)
* Проверим, подключившись по ssh: ssh [user@10.0.3.103](mailto:user@10.0.3.103) “ls /home/user”

**2) Аутентификация**

Выполнять команды sudo через Ansible не получится, если не изменить файл */etc/ansible/ansible.conf :*

* *ansaible webservers -m copy -a “src=/home/user/group dest=/var/www/html/” (→ FAILD)*

После настрйоки файла конфигураций, следует запустить операцию копиррвания с аргументом –ask-become-pass:

* ansaible –ask-become-pass webservers *-m copy -a “src=/home/user/group dest=/var/www/html/” (Потребует пароль).*

*Для проверки можно зайти на удаленный сервер и проверить выполенение.*

**3) Сценарии Ansible playbook**

Playbook – скрипт для выполнения действий, которые выполняет Ansible.

Также playbook можно использовать как ссылку, указывающую на ресурсы, распределенные по специальному дереву каталогов.

Пример: файл site.yml

Чтобы убедиться, что скрипт работает, нужно создать каталог project с файлом index.html в домашнем каталоге. После запуска playbook, копия файла должна появиться в корневом каталоге веб-документа.

Запуск: ansaible-playbook site.yml

Посмотреть синтаксис и ф-ции playbook: ansible-doc apt

**3.1 Создание многоуровневых ролевых сценариев playbook:**

Когда инфраструктура будет развиваться, хранить все инструкции в одном файле будет неудобно. Поэтому пребегают к распределению задач в отдельные каталоги и файлы.

Ansible организует свои модельные элементы в роли и предоставляет собственный инструмент командной строки для управления ролями и создания необходимой структуры файловой системы для запуска новых ролей. (ansible-galaxy).

**3.2. Создание ролей Ansible.**

Прееходим в корневой каталог пользователя, создаем каталог role, переходим в него, инициализируем каталог:

* mkdir roles
* cd roles
* ansaible-galaxy init web-app (web-app – имя для роли)

Так создастся новый наполненный катлог web-app, можно рекурсивно посмотреть его содержимое:

* cd web-app
* ls -R

Параметры, добавленные в файлы под каталогом ролей веб-приложения, могут вызваться либо из playbook верхнего уровня, либо непосредтвенно из командной строки.

Например, можно определить корневое положение веб-документа в файле: role/web-app/defaults/main.yml как webroot\_location:/var/www/myroot/

! Каждое приложение и сервис могут быть представлены своей собственной ролью в Ansible. Для хорошего взаимодействия нужно будет расставить приоритеты для перкрывабщихся переменных. (Значения в vars/ переопределяют значения в defauts/, значения, заданные с помощью –extra-vars= (-e) превосходят все остальные)

То, что может вхдить в роль/веб-приложение/каталог:

* vars/ - содержит информацию о расположении ФС для ключей шифрования
* templates/ - шаблоны для установки (к примеру, в виде файлов конфигурации apache в формате Python)
* files/ другие файлы(HTML, который копировали в предыдущм примере), которые исп для данных хоста.

**3.3. Управление паролями:**

Возможно, придется включать пароли хостов в архитектурур Ansible, которые не стоит никогда хранит в виде простых текстовых документов.

Ansible предоставляет инструмент Vault, хранящий конфиденциальные данные в зашифрованных файлах, каоторые playbook может безопасно вызвать.

Открыть редактор, в которй можно ввести парольVault^

* export EDITOR=nano (vim-по- умолчанию меняем на nano)
* ansible-vault create mypasswordfile

Если хосты используют только один пароль , то можно добавить аргумент

--ask-vault-pass к команде ansible-playbook:

* ansible-playbook site.yml --ask-vault-pass