Удобный Ansible

Максим Залысин | Head of DevOps | Positive Technologies

Speaker

Максим Залысин

Head of DevOps

Positive Technologies

21 год

опыта исследований
и разработок

2000+

инженеров по ИБ,
разработчиков, аналитиков
и других специалистов

250+

экспертов в нашем
исследовательском
центре безопасности

500+

серверов в инфраструктуре
DevOps

≈100

Ansible ролей

20+

проектов Ansible

создаем продукты и решения

проводим аудиты безопасности

расследуем инциденты

исследуем угрозы

Начало

Install

Install

```
— $ sudo brew install ansible

— $ sudo port install ansible

— $ sudo pacman -S ansible

— $ sudo yum install ansible

— $ sudo apt install ansible

— $ snap install ansible-ryanjyoder
```

Install

```
→$ pip install ansible

  →$ pip install --user ansible

  →$ pip install --user --upgrade ansible
```

docs.ansible.com / Installing Ansible

```
→$ pipx install --include-deps ansible

  →$ pipx upgrade ansible
```

Install

```
→$ pip install ansible

  →$ pip install --user ansible

  →$ pip install --user --upgrade ansible
```

docs.ansible.com / Installing Ansible

```
→$ pipx install --include-deps ansible

  →$ pipx upgrade ansible
```

ansible

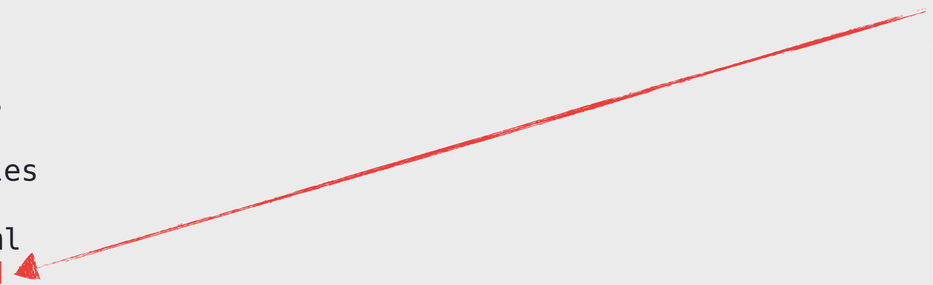
ansible-core

Основное

Ansible collections

Collection

```
collection
├── playbooks
├── plugins
│   └── modules
├── roles
├── galaxy.yml
└── README.md
```



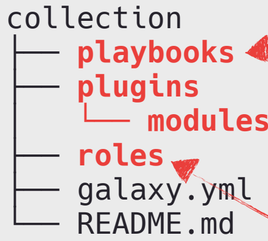
$ cat README.md

```
collection_name
---

_Описание этой коллекции в 2-3 предложения. Главное дать ответ на вопрос
"Для чего эта коллекция?"_


### Ссылки
_Список полезных ссылок на описание функций, собранных в этой коллекции. Те
ссылки что отвечают на вопрос "Что почитать про коллекцию?"_
- <http://...>


### Примеры
```yaml

```


TODO
- ...
```

Collection

```
collection
├── playbooks
├── plugins
│   └── modules
├── roles
├── galaxy.yml
└── README.md
```

```
$ cat galaxy.yml
---
version: "0.0.0"
namespace: ptsecurity
name: collection_name
description: This collection description
authors: ["Positive Technologies"]
license: unlicense
readme: README.md
```

Collection



playbooks

- Вспомогательные плейбуки, связанные общим назначением или функциональностью.
- Используется в проектах подключением через `ansible.builtin.import_playbook`.

plugins/modules

- Собственные Ansible модули на Python.

roles

- Небольшие «локальные» роли коллекции, не требующие независимого версионирования.
- Использование «локальных» ролей коллекции допустимо только в плейбуках коллекции.

Collection

```
collection
├── playbooks
├── plugins
│   └── modules
├── roles
├── galaxy.yml
└── README.md
```

Правила

- Новая коллекция только из шаблона в GitLab.
- Разделитель в имени коллекции — только нижнее подчеркивание.
- Версионирование коллекции по правилам SemVer 2.0.0 тегами в GitLab и указанием версии в `galaxy.yml`.
- Коллекции используются для:
  - Собственных Ansible модулей на Python.
  - Вспомогательных плейбуков, связанных общим назначением или функциональностью.
  - Небольших «локальных» ролей, используемых во вспомогательных плейбуках.

Ansible roles

Role

```
role
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── library
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── vars
│   └── main.yml
└── README.md
```

$ cat README.md

```
role_name
---

_Описание функции этой роли в 2-3 предложения. Главное дать ответ на вопрос
"Что делает роль?"_


### Ссылки
_Список полезных ссылок на сайт компонента, для которого сделана роль,
репозиторий с исходниками, полезные статьи и примеры. Те ссылки что отвечают
на вопрос "Что почитать про этот компонент?"_
- <http://...>


### Переменные
_Перечисление всех переменных роли с описание и значением по-умолчанию или
указанием обязательности, если такое требуется. Название переменной
обязательно начинается с названия роли._
- **`role_name_variable_x`** *(type=bool|number|string|list|dict)* -
Описание необязательной переменной без значения по умолчанию.
- **`role_name_variable_y`** *(type=bool|number|string|list|dict,
mandatory)* - Описание обязательной переменной.
- **`role_name_variable_z`** *(type=bool|number|string|list|dict,
default=...)* - Описание переменной с указанием её значения по умолчанию.
Если значение содержит пробел, обернуть в двойные кавычки.


### Требования
- ...


### Примеры
```yaml

```


TODO
- ...
```

Role

```
role
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── library
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── vars
│   └── main.yml
└── README.md
```

```
$ cat meta/main.yml
```

```yaml
---
dependencies: []
galaxy_info:
  description: This role description
  author: Positive Technologies
  license: unlicense
  min_ansible_version: "2.15"
  platforms:
    - name: GenericLinux
      version: [any]
    - name: Windows
      version: [any]
```

Role

```
role
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── library
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── vars
│   └── main.yml
└── README.md
```

```
$ cat {defaults,handlers,tasks,vars}/main.yml
---
```

Role

```
role
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── library
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── vars
│   └── main.yml
└── README.md
```

library

→ Модули Ansible на Python, расширяющие функциональность роли.

Role

```
role
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── library
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── vars
│   └── main.yml
└── README.md
```

Правила

- Новая роль только из шаблона в GitLab.
- Разделитель в имени роли — только нижнее подчеркивание.
- Версионирование роли по правилам SemVer 2.0.0 тегами в GitLab.
- Именование публичных переменных роли начинается с имени роли.
- Именование приватных переменных роли начинается с нижнего подчеркивания и имени роли.

Ansible projects

Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

```
$ cat README.md
```

```
project-name
---

_Описание этого проекта в 2-3 предложения. Главное дать ответ на вопрос "Что
за проект?"_


### Ссылки
_Список полезных ссылок на описание инфраструктуры, окружения, документацию,
hosting WEB UI, под который сделан проект. Те ссылки что отвечают на вопрос
"Что почитать про проект?"_
- <http://...>


### Подготовка к использованию
Установка внешних зависимостей
```shell
ansible-galaxy install --force --role-file requirements.yml
```

Для защиты "секретов" проекта, в том числе для доступа в пространство
HashiCorp Vault проекта, используется файл `.vault`.
Файл `.vault` объявлен в `ansible.cfg` строкой `vault_password_file` и
добавлен в `.gitignore` для защиты от попадания в Git.
```

Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

$ cat ansible.cfg

```
[defaults]
forks = 10
timeout = 30
host_key_checking = False
retry_files_enabled = False
gathering = smart
vault_password_file = .vault
library = plugins/modules
roles_path = ~/.ansible/galaxy_roles:roles:../../roles
log_path = ansible.log
stdout_callback = yaml
bin_ansible_callbacks = True
callbacks_enabled = profile_tasks
ansible_managed = Ansible managed: {file} modified on %Y-%m-%d %H:%M:%S by
{uid} on {host}

[ssh_connection]
pipelining = True
transfer_method = piped
ssh_args = -o ControlMaster=auto -o ControlPersist=15m

[callback_profile_tasks]
task_output_limit = 0
```

Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

```
$ cat .gitignore
```

```
.vault
ansible.log
```

Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

```
$ cat requirements.yml
---
collections: []

roles: []
```

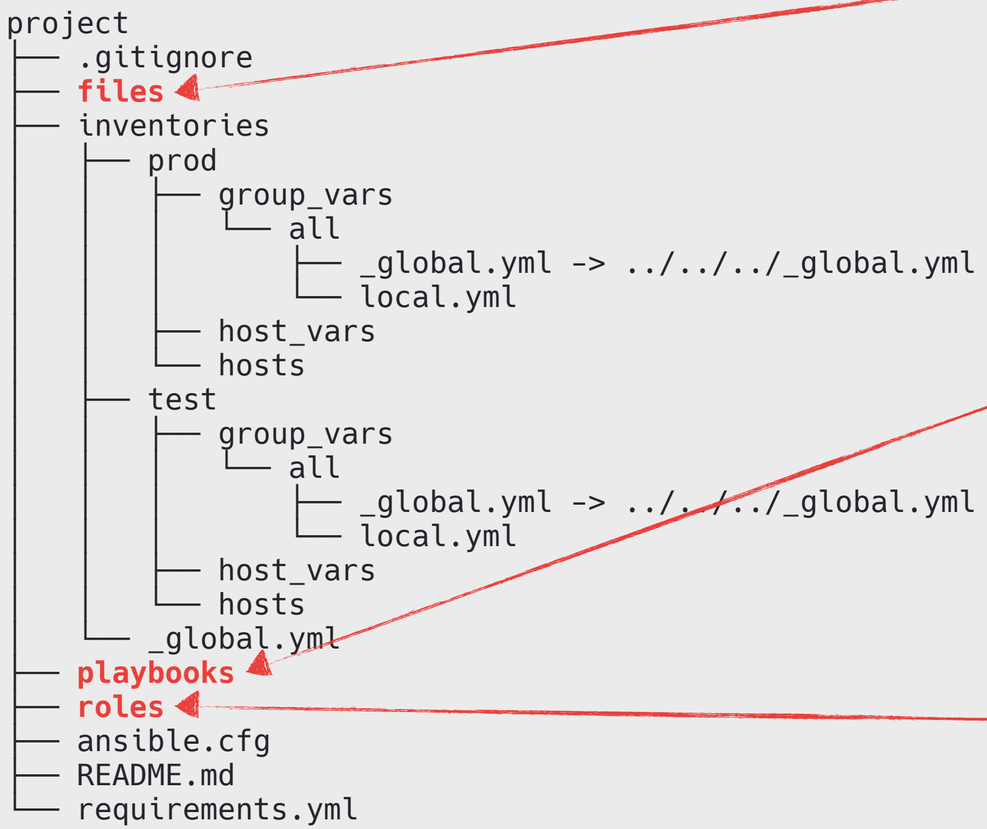
Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

```
$ cat requirements.yml
```

```yaml
---
collections:
  - name: git+ssh://git@gitlab.example.com/ansible/collections/playbooks.git
    version: v0.1.0

roles:
  - name: docker_v0.2.1
    src: git+ssh://git@gitlab.example.com/ansible/roles/docker.git
    version: v0.2.1

  - name: keepalived_v0.1.0
    src: git+ssh://git@gitlab.example.com/ansible/roles/keepalived.git
    version: v0.1.0

  - name: nginx_v0.1.1
    src: git+ssh://git@gitlab.example.com/ansible/roles/nginx.git
    version: v0.1.1

  - name: nginx_v0.2.0
    src: git+ssh://git@gitlab.example.com/ansible/roles/nginx.git
    version: v0.2.0

  - name: node_exporter_v0.1.1
    src: git+ssh://git@gitlab.example.com/ansible/roles/node_exporter.git
    version: v0.1.1

  - name: postgres_v0.2.2
    src: git+ssh://git@gitlab.example.com/ansible/roles/postgres.git
    version: v0.2.2

  - name: postgres_v0.4.0
    src: git+ssh://git@gitlab.example.com/ansible/roles/postgres.git
    version: v0.4.0

  - name: systemd_journald_v0.1.0
    src: git+ssh://git@gitlab.example.com/ansible/roles/systemd_journald.git
    version: v0.1.0
```

Project



files

- Статичные файлы проекта (ssl, terraform, ...), обращение к которым возможно только в плейбуке.
  - Контент файла в роль можно передать через `lookup('ansible.builtin.file', playbook_dir~'/../files/…')`.

playbooks

- Плейбуки проекта.
- Являются местом связывания ролей и инвентаря и точкой запуска сценария.

roles

- «Локальные» роли проекта, не требующие версионирования и использования в других проектах.
- «Локальные» роли создаются полностью аналогично общим ролям из шаблона в GitLab.

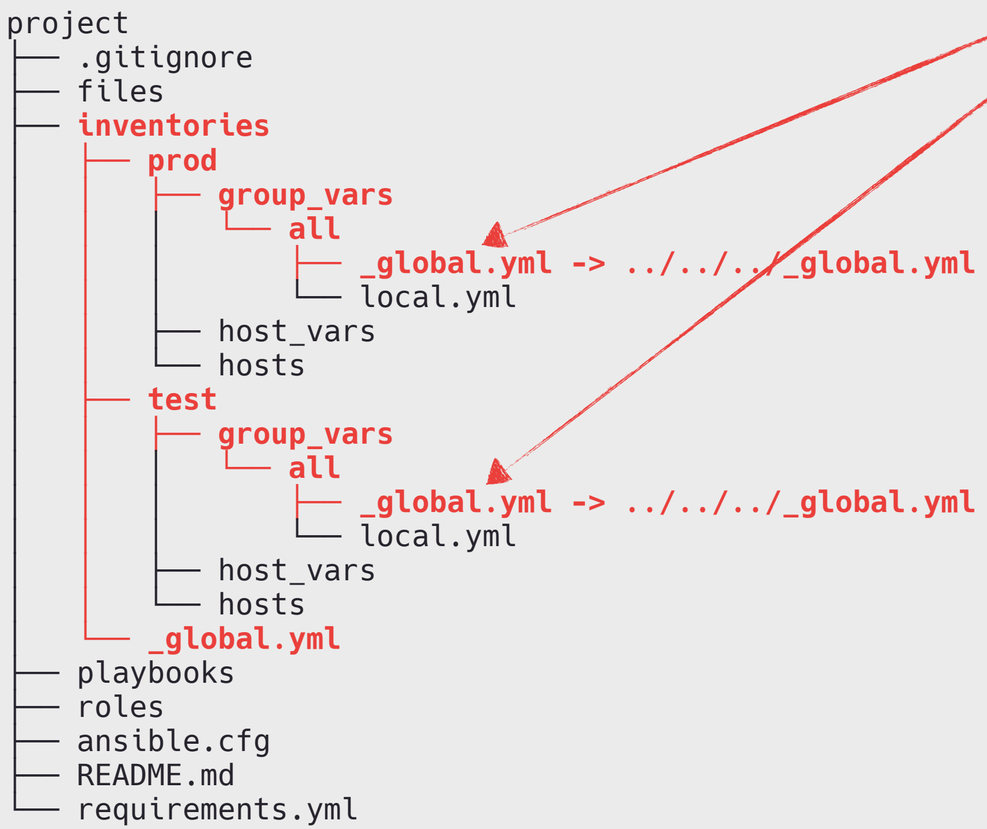
Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

inventories

- Все окружения в одном `inventories`.
- `hosts` в простом `INI`-формате.
- Указание переменных только на трех уровнях инвентаря:
  - `hosts` для переменных типо `ansible_host`.
  - `host_vars` для переменных конкретного сервера.
  - `group_vars` для переменных групп серверов в `hosts`.
    - Указание глобальных переменных для всех окружений в общем файле `group_vars/all/_global.yml`.
    - Указание локальных переменных для окружения в файле `group_vars/all/local.yml`.

Project




```
$ cat inventories/_global.yml
```

```yaml
---
# sysctl
_sysctl_entries:
  - name: net.ipv4.ip_unprivileged_port_start
    value: "0"
  - name: net.ipv6.conf.all.disable_ipv6
    value: "1"

# packages
_required_apt_packages: [ca-certificates, curl, python3, python3-pip, rsync]
_required_pip_packages: []

# systemd_journald
systemd_journald_configs:
  - filename: storage.conf
    content: |
      [Journal]
      Storage=persistent
      Compress=yes
      SystemMaxUse=10G
      RuntimeMaxUse=1G

# sshd
_sshd_configs:
  - filename: root.conf
    content: PermitRootLogin no
  - filename: password.conf
    content: PasswordAuthentication no
```

Project

```
project
├── .gitignore
├── files
├── inventories
│   ├── prod
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   ├── test
│   │   ├── group_vars
│   │   │   └── all
│   │   │       ├── _global.yml -> ../../../_global.yml
│   │   │       └── local.yml
│   │   ├── host_vars
│   │   └── hosts
│   └── _global.yml
├── playbooks
├── roles
├── ansible.cfg
├── README.md
└── requirements.yml
```

Правила

- Новый проект только из шаблона в GitLab.
- Разделитель в имени проекта — только дефис.
- В проекте возможно использование локальных ролей, создаваемых по единому шаблону.

Финал

Structure

```
ansible
├── projects
│   ├── ...
│   ├── blackbox
│   ├── cybsi
│   ├── devops
│   ├── ...
│   ├── ngfw
│   ├── ml
│   └── ...
└── roles
    ├── ...
    ├── docker
    ├── consul_service
    ├── ...
    ├── keepalived
    ├── mongo
    ├── nginx
    ├── node_exporter
    ├── ...
    ├── postgres
    ├── redis
    └── ...
```

$ cat project/.../ansible.cfg

```
[defaults]
forks = 10
timeout = 30
host_key_checking = False
retry_files_enabled = False
gathering = smart
vault_password_file = .vault
library = plugins/modules
roles_path = ~/.ansible/galaxy_roles:roles:../../roles
log_path = ansible.log
stdout_callback = yaml
bin_ansible_callbacks = True
callbacks_enabled = profile_tasks
ansible_managed = Ansible managed: {file} modified on %Y-%m-%d %H:%M:%S by
{uid} on {host}

[ssh_connection]
pipelining = True
transfer_method = piped
ssh_args = -o ControlMaster=auto -o ControlPersist=15m

[callback_profile_tasks]
task_output_limit = 0
```

Quality assurance

Статический анализ

- ansible-lint с включенным yamllint в каждом merge request роли.
- Собственный project-lint в merge request проекта, который отлавливает:
  - Дубликаты ролей в requirements.yml.
  - Различие версии роли с её именем в requirements.yml.
  - Различие названия и версии роли с её именем в requirements.yml.
  - Некорректный src-адрес роли в requirements.yml.
  - Неправильное название репозитория роли в requirements.yml.
  - Сортировку коллекций и ролей в requirements.yml.
  - Дубликаты коллекций и ролей в requirements.yml.
  - Используемая в плейбуке роль отсутствует в requirements.yml.
  - Объявленная в requirements.yml роль не используется ни в одном плейбуке.
  - Некорректное объявление роли в плейбуке.

Quality assurance

Статический анализ

- ansible-lint с включенным yamllint в каждом merge request роли.
- Собственный project-lint в merge request проекта, который отлавливает:
  - Дубликаты ролей в requirements.yml.
  - Различие версии роли с её именем в requirements.yml.
  - Различие названия и версии роли с её именем в requirements.yml.
  - Некорректный src-адрес роли в requirements.yml.
  - Неправильное название репозитория роли в requirements.yml.
  - Сортировку коллекций и ролей в requirements.yml.
  - Дубликаты коллекций и ролей в requirements.yml.
  - Используемая в плейбуке роль отсутствует в requirements.yml.
  - Объявленная в requirements.yml роль не используется ни в одном плейбуке.
  - Некорректное объявление роли в плейбуке.

Тестирование

- Molecule не используем, тестируем на реальной инфраструктуре.
- Приглядывались к github.com/weaveworks/footloose.

Спасибо!