Principes des Systèmes d'exploitation

Tags and Files

Guillerme Duvillié

1. Tags
2. Files
3. Work

Tags

Why?

- Groups and machine name allow to target machine in a playbook,
- Tags allow to target plays that will be executed.
- Makes easier to test playbooks,
- Avoid to run unnecessary tasks,
- Keep playbooks consistent.

How ?

- Single YAML keyword: **`tags:`**,
- List of tags.

```yaml
- name: Install samba
  become: true
  hosts: fileservers
  tasks:
    - name: Install Samba
      tags: samba,installation
      package:
        name: "{{ samba_pkg }}"
        state: latest

    - name: Start Samba
      tags:
        - samba
        - configuration
      ansible.builtin.service:
        name: "{{ samba_service }}"
        state: started
```

And then?

- Can select the tasks to execute based on tags:

```
$ ansible-playbook --tags configuration myplaybook.yml
```

- List existing tags:

```
$ ansible-playbook --list-tags myplaybook.yml
```

Where ? (1)

- At task level to tag a single task,
- At play level, to apply to all tasks of a play:

```yaml
- name: Install samba
  become: true
  hosts: fileservers
  tags: samba
  tasks:
    - name: Install Samba
      tags: installation
      package:
        name: "{{ samba_pkg }}"
        state: latest

    - name: Start Samba
      tags: configuration
      ansible.builtin.service:
        name: "{{ samba_service }}"
        state: started
```

Where ? (2)

- At block level to apply to multiple task of a play but not all:

```yaml
- name: Install functional Apache
  become: true
  hosts: webservers
  tasks:
    - name: Ubuntu apache
      tags: ubuntu,apache,apache2
      block:
      - name: Install Apache
        package:
          name:
            - apache2
            - libapache2-mod-php
          state: latest
      - name: Start Apache
        ansible.builtin.service:
          name: apache2
          state: started
    - name: Fedora Apache
      tags: fedora,apache,httpd
      block:
      - name: Install Apache
        package:
          name:
            - httpd
            - php
          state: latest
      - name: Start Apache
        ansible.builtin.service:
          name: httpd
          state: started
```

Special tags

- ***always***:
  - task is always executed,
  - unless options ***--skip-tags*** `always` is used,
  - fact gathering is a task tagged by default with ***always***.
- ***never***:
  - task is never executed,
  - unless options ***--tags*** `never` is used.

Getting further

- `--tags all`: run all tasks, ignore tags (default behavior)
- `--tags [tag1, tag2]`: run only tasks with either the tag tag1 or the tag tag2
- `--skip-tags [tag3, tag4]`: run all tasks except those with either the tag tag3 or the tag tag4
- `--tags tagged`: run only tasks with at least one tag
- `--tags untagged`: run only tasks with no tags
- `--list-tasks`: when used with `--tags` or `--skip-tags`, generate a preview of tasks to execute.
- `TAGS_RUN` and `TAGS_SKIP` define, in ansible configuration, the tags to be run (or skipped) by default.

Files

Send file to remote servers

- Use module `ansible.builtin.copy:`,
- Allows to recursively sends file from workstation to remote machine,
- Allows to set:
  - destination path,
  - file permissions,
  - file owner,
  - …
- You should always specify at least:
  - `src:`,
  - `dest:`,
  - `group:`,
  - `owner:`,
  - `mode:`.

Fetch files from remote servers

- Possible using `ansible.builtin.copy:`,
- But prefer `ansible.builtin.fetch:`,
- Files are organized by hostname in file tree.

Other interesting modules

- `ansible.builtin.unarchive:`: share and decompress archives files,
- `community.general.archive:`: creates archives,
- `ansible.builtin.file:`: manage files and file properties,
- `ansible.builtin.replace:`: modify files using regular expressions,
- `ansible.builtin.tempfile:`: creates temporary files and directories.

Work

Glances installation

Using https://www.tecmint.com/glances-monitor-remote-linux-in-web-server-mode/, deploy a web accessible `glances` service on all machines.