

Ansible Basic

An Ansible Training Course

PRESENTED BY:

Bittnet DevOps Team

ANSIBLE

We Make Custom Trainings



FASTER
SMARTER
SAFER

7. Working with Templates



Templates Basics

- Templates give the ability to provide a skeletal file that can be dynamically completed using variables.
- The most common template use case is configuration file management.
- Templates are generally used by providing a template file on the ansible control node, and then using the template module within your playbook to deploy the file to a target server or group.
- Templates are processed using the Jinja2 template language.

Template Module

- The template module is used to deploy template files
- There are two required parameters:
 - **src** : the template to use (on the ansible control host)
 - **dest** : where the resulting file should be located (on the target host)
- All templating takes place **on the control host!**

Template Module

- A useful optional parameter is **validate** which requires a successful validation command to run against the result file prior to deployment
- It is also possible to set basic file properties using the template module

Template Module

```
- name: Update sshd configuration safely, avoid locking yourself out
  template:
    src: etc/ssh/sshd_config.j2
    dest: /etc/ssh/sshd_config
    owner: root
    group: root
    mode: '0600'
    validate: /usr/sbin/sshd -t -f %s
    backup: yes
```

Template Module

- These are some of the other **parameters** which we can use to change some default behavior of template module:
 - **force** – If the destination file already exists, then this parameter decides whether it should be replaced or not. By default, the value is 'yes'.
 - **mode** – If you want to set the permissions for the destination file explicitly, then you can use this parameter.
 - **backup** – If you want a backup file to be created in the destination directory, you should set the value of the backup parameter to 'yes'. By default, the value is 'no'.
 - **group** – Name of the group that should own the file/directory.

Template Module

- Additional variables that can be used in templates:
 - **ansible_managed**: contains a string which can be used to describe the template name, host, modification time of the template file and the owner uid.
 - **template_host**: contains the node name of the template's machine.
 - **template_uid**: is the numeric user id of the owner.
 - **template_path**: is the path of the template.
 - **template_fullpath**: is the absolute path of the template.
 - **template_destpath**: is the path of the template on the remote system
 - **template_run_date**: is the date that the template was rendered.

Template File

- Template file are essentially little more than text files
- Template files are designed with a file extension of `.j2`
- Template files have access to the same variables that the play that calls them does

Template File Example

```
student@ansible-00-hivemaster:~$ vi template.j2
Nickname: {{ nickname }}
Email Address: {{ email }}
Description: {{ description }}
Role: {{ role }}
Organization: {{ org }}
```

Note: From the next lab.

Filters

- Ansible uses Jinja2 filters for transforming data inside a template expression.
- Take into account that templating happens **on the Ansible controller**, **not** on the task's target host.
- Ansible adds some additional filters to the Jinja2 default ones, and you can also write your own custom filters

Filter Examples



Formatting Data

- Reading and writing JSON/YAML:

```
{{ variable | to_json }}
```

```
{{ variable | to_nice_json(indent=2) }}
```

```
{{ variable | to_yaml }}
```

```
{{ variable | to_nice_yaml(indent=4) }}
```

```
{{ variable | from_json }}
```

```
{{ variable | from_yaml }}
```

Mandatory and Default

- By default, ansible will fail when an undefined variable is used
 - This can be turned off from ansible.cfg (`error_on_undefined_vars = False`)
 - With the setting turned off, we can still force some variables to be defined:

```
{{ variable | mandatory }}
```

- Alternatively, we can set defaults for undefined vars:

```
{{ variable | default('abc') }}
```

Dictionary to Items

- Convert a dictionary into a list of key/value items, suitable for looping:

```
{{ dict | dict2items }}
```

```
{{ dict | dict2items(key_name='file_name', value_name='file_path') }}
```

New in Ansible 2.8!

```
files:  
  users: /etc/passwd  
  groups: /etc/group
```

```
- file_name: users  
  file_path: /etc/passwd  
- file_name: groups  
  file_path: /etc/group
```


Subelements

- Product of an object with subelement values of that object (similar to the *subelements* lookup):

```
users:
- name: alice
  authorized:
    - /tmp/alice/onekey.pub
    - /tmp/alice/twokey.pub
  groups:
    - wheel
    - docker
- name: bob
  authorized:
    - /tmp/bob/id_rsa.pub
  groups:
    - docker
```

```
{{ users | subelements('authorized') }}
```

```
-
- name: alice
  groups:
    - wheel
    - docker
  authorized:
    - /tmp/alice/onekey.pub
    - /tmp/alice/twokey.pub
    - /tmp/alice/onekey.pub
-
- name: alice
  groups:
    - wheel
    - docker
  authorized:
    - /tmp/alice/onekey.pub
    - /tmp/alice/twokey.pub
    - /tmp/alice/twokey.pub
-
- name: bob
  authorized:
    - /tmp/bob/id_rsa.pub
  groups:
    - docker
    - /tmp/bob/id_rsa.pub
```

Network Utilities

- Random Mac Address Filter

- This filter can be used to generate a random MAC address from a string prefix.

```
{{ 'aa:bb:cc' | random_mac }}  
# => 'aa:bb:cc:13:a4:b3'
```

- IP Address Filters

- Check if a string is a valid IP address

```
{{ var | ipaddr }}
```

- Extract IP addresses from a CIDR block

```
{{ '10.11.12.13/24' | ipaddr(2) }}  
# => '10.11.12.2/24'
```

Other Filters

- Display the underlying Python type of a variable

```
{{ myvar | type_debug }}
```

- Ternary operator (a?b:c)

```
{{ a | ternary(b,c) }}
```

Checking a Template. The 'template' Lookup

```
> cat template_test.yml
- name: Test a template
  hosts: localhost
  vars:
    mylist:
      - 1
      - 2
      - a
      - b
      - c
  gather_facts: no

  tasks:
    - name: Print templated output
      debug:
        msg: "{{ lookup('template', 'template.j2').split('\n') }}"
```

Loops in Templates

```
{% for i in mylist %}
{{i}}
{% endfor %}
```

```
> ansible-playbook template_test.yml
```

```
TASK [Print templated output]
*****
ok: [localhost] => {
  "msg": [
    "1",
    "2",
    "a",
    "b",
    "c",
    ""
  ]
}
```

Trimming Whitespace

```
{% for i in mylist %}  
    {{i}}  
{%- endfor %}
```

```
> ansible-playbook template_test.yml
```

```
TASK [Print templated output]
```

```
*****
```

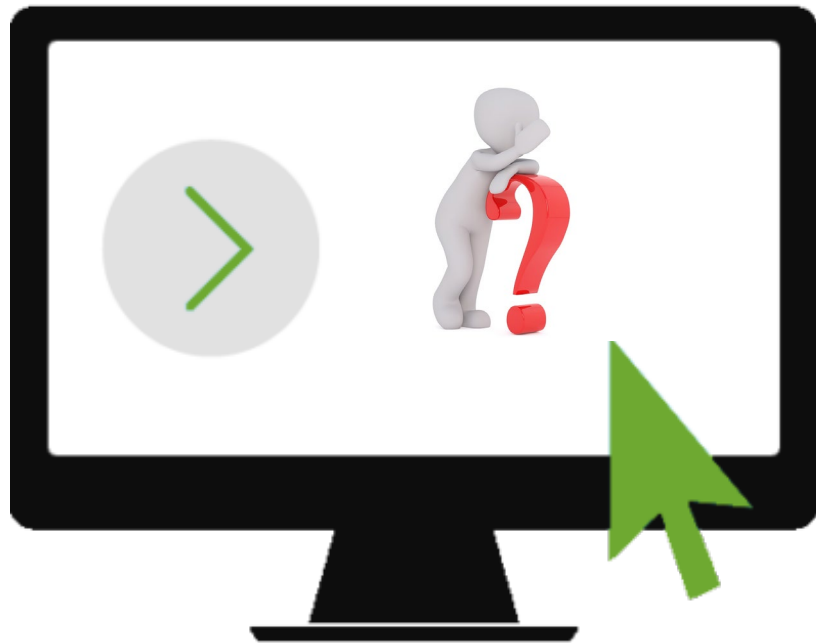
```
ok: [localhost] => {
```

```
  "msg": [
```

```
    " 1 2 a b c",
```

```
  ]
```

```
}
```



Lab 07: Templates





Solutions for training the world.