

TTS 10.0 COOKBOOK

(NSD ARCHITECTURE DAY02)

版本编号 10.0

2018-08

达内 IT 培训集团

NSD ARCHITECTURE DAY02

1. 练习 1 : playbook 练习

• 问题

本案例要求：

- 安装 Apache 并修改监听端口为 8080
- 修改 ServerName 配置，执行 apachectl -t 命令不报错
- 设置默认主页 hello world
- 启动服务并设开机自启

• 步骤

实现此案例需要按照如下步骤进行。

步骤一：playbook 的 ping 脚本检测

```
[root@ansible ansible]# vim ping.yml
---
- hosts: all
  remote_user: root
  tasks:
    - ping:

```

```
[root@ansible ansible]# ansible-playbook ping.yml //输出结果

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [web1]
ok: [web2]
ok: [cache]
ok: [db1]
ok: [db2]

TASK [ping] *****
ok: [db1]
ok: [web2]
ok: [cache]
ok: [web1]
ok: [db2]

PLAY RECAP *****
cache      : ok=2    changed=0    unreachable=0    failed=0
db1        : ok=2    changed=0    unreachable=0    failed=0
db2        : ok=2    changed=0    unreachable=0    failed=0
web1       : ok=2    changed=0    unreachable=0    failed=0
web2       : ok=2    changed=0    unreachable=0    failed=0
```

注意：如果检测的时候出错，会在当前的目录生成一个新的文件（以.retry 结尾），可以去这个文件里面看是哪个主机的错

步骤二：用 playbook 安装 Apache, 修改端口, 配置 ServerName, 修改主页, 设置开机自启

```
[root@ansible ansible]# vim http.yml

---
- hosts: cache
  remote_user: root
  tasks:
    - name: install one specific version of Apache
      yum:
        name: httpd           //安装 Apache
        state: installed
    - lineinfile:
        path: /etc/httpd/conf/httpd.conf
        regexp: '^Listen '
        line: 'Listen 8080'    //修改端口为 8080
    - replace:
        path: /etc/httpd/conf/httpd.conf
        regexp: '^#(ServerName).*' //配置 ServerName
        replace: '\1 localhost'
    - service:
        name: httpd
        enabled: yes           //开机自启
        state: restarted
    - copy:
        src: /root/index.html  //修改主页, 可以自己写个页面
        dest: /var/www/html/index.html

[root@ansible ansible]# curl 192.168.1.56:8080
hello world
[root@ansible ansible]# ssh cache
Last login: Fri Sep  7 09:32:05 2018 from 192.168.1.51
[root@cache ~]# apachectl -t
Syntax OK
```

2. 案例 2：变量练习

• 问题

本案例要求熟悉 playbook 进阶：

- 练习使用 user 模块添加用户
- 练习使用变量简化 task, 让 play 通用性更强
- 练习使用过滤器

• 步骤

实现此案例需要按照如下步骤进行。

步骤一：使用 user 模块添加用户, 并修改密码

```
[root@ansible ansible]# vim user.yml
---
- hosts: cache
```

```
remote_user: root
vars:
  username: xiaoming
tasks:
  - name: create user "{{username}}"
    user: group=wheel uid=1000 name={{username}}
  - shell: echo 123456 | passwd --stdin xiaoming
  - shell: chage -d 0 {{username}}

[root@ansible ansible]# ansible-playbook user.yml //执行结果

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [create user " xiaoming "] *****
changed: [cache]

TASK [command] *****
changed: [cache]

TASK [command] *****
changed: [cache]

PLAY RECAP *****
cache                : ok=4    changed=3    unreachable=0    failed=0
```

步骤二：变量过滤器，创建一个用户，设置密码

```
[root@ansible ansible]# vim user1.yml
---
- hosts: cache
  remote_user: root
  tasks:
    - user:
        name: lisi
        group: root
        password: "{{'123456' | password_hash('sha512')}}"
    - shell: chage -d 0 lisi

[root@ansible ansible]# ansible-playbook user1.yml

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [user] *****
changed: [cache]

TASK [command] *****
changed: [cache]

PLAY RECAP *****
cache                : ok=3    changed=2    unreachable=0    failed=0
```

步骤三：定义一个变量创建用户

```
[root@ansible ansible]# vim user2.yml
---
```

```
- hosts: cache
  remote_user: root
  vars:
    user: zhangs
  tasks:
    - user:
        name: "{{user}}"
        group: root
        password: "{{'123456' | password_hash('sha512')}}"
    - shell: chage -d 0 "{{user}}"
[root@ansible ansible]# ansible-playbook user2.yml
PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [user] *****
changed: [cache]

TASK [command] *****
changed: [cache]

PLAY RECAP *****
cache                : ok=3    changed=2    unreachable=0    failed=0
```

3. 案例 3 : handlers 练习

• 问题

本案例要求：

- 安装 Apache 软件
- 配置文件，重新载入配置文件让服务生效
- 使用 handlers 来实现

• 步骤

实现此案例需要按照如下步骤进行。

步骤一：error

playbook 从上往下顺序执行，若报错，后面的命令不会在执行，若想解决有两种方法：

1) 当返回值为假时，显示 true：- shell: setenforce 0 || true

```
[root@ansible ansible]# vim user5.yml
---
- hosts: cache
  remote_user: root
  vars:
    user: bb
  tasks:
    - shell: setenforce 0 || true
    - user:
        name: "{{user}}"
        group: root
        password: "{{'123456' | password_hash('sha512')}}"
    - shell: chage -d 0 "{{user}}"
```

```
[root@ansible ansible]# ansible-playbook user5.yml

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [command] *****
changed: [cache]

TASK [user] *****
changed: [cache]

TASK [command] *****
changed: [cache]

PLAY RECAP *****
cache                : ok=4    changed=3    unreachable=0    failed=0
```

2、忽略: `ignoring_errors: True`(推荐使用这个, 会有报错信息, 告诉你错误忽略, 继续执行下面的命令)

```
[root@ansible ansible]# vim user6.yml
---
- hosts: cache
  remote_user: root
  vars:
    user: bb
  tasks:
    - shell: setenforce 0
      ignore_errors: True
    - user:
        name: "{{user}}"
        group: root
        password: "{{'123456' | password_hash('sha512')}}"
    - shell: chage -d 0 "{{user}}"

[root@ansible ansible]# ansible-playbook user6.yml

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [command] *****
fatal: [cache]: FAILED! => {"changed": true, "cmd": "setenforce 0", "delta":
"0:00:00.004198", "end": "2018-09-07 11:08:14.936959", "msg": "non-zero return code",
"rc": 1, "start": "2018-09-07 11:08:14.932761", "stderr": "setenforce: SELinux is
disabled", "stderr_lines": ["setenforce: SELinux is disabled"], "stdout": "",
"stdout_lines": []}
...ignoring

TASK [user] *****
changed: [cache]

TASK [command] *****
changed: [cache]

PLAY RECAP *****
cache                : ok=4    changed=3    unreachable=0    failed=0
```

步骤二: handlers

关注的资源发生变化时采取的操作

1) 使用 handlers 来配置文件，重新载入配置文件让服务生效

```
[root@ansible ansible]# vim adhttp.yml
---
- hosts: cache
  remote_user: root
  tasks:
    - copy:
        src: /root/httpd.conf
        dest: /etc/httpd/conf/httpd.conf
        owner: root
        group: root
        mode: 0644
      notify:
        - restart httpd
  handlers:
    - name: restart httpd
      service: name=httpd state=restarted

[root@ansible ansible]# ansible-playbook adhttp.yml

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [copy] *****
ok: [cache]

PLAY RECAP *****
cache                : ok=2    changed=0    unreachable=0    failed=0

[root@ansible ansible]# ssh cache apachectl -t
Syntax OK
[root@ansible ansible]# curl 192.168.1.56:8080
hello world
```

2) 使用脚本调用变量更改服务

```
[root@ansible ansible]# vim adhttp2.yml
---
- hosts: cache
  remote_user: root
  vars:
    server: httpd
  tasks:
    - copy:
        src: /root/httpd.conf
        dest: /etc/httpd/conf/httpd.conf
        owner: root
        group: root
        mode: 0644
      notify:
        - restart "{{server}}"
  handlers:
    - name: restart "{{server}}"
      service: name=httpd state=restarted
[root@ansible ansible]# ansible-playbook adhttp2.yml

PLAY [cache] *****
*****
```

```
*****

TASK                                                                [Gathering Facts]
*****
*****
ok: [cache]

TASK                                                                [copy]
*****
*****
ok: [cache]

PLAY                                                                RECAP
*****
*****
cache                      : ok=2    changed=0    unreachable=0    failed=0

[root@ansible ansible]#
```

4. 案例 4 : 编写 playbook

• 问题

本案例要求：

- 把所有监听端口是 8080 的 Apache 服务全部停止

• 步骤

实现此案例需要按照如下步骤进行。

步骤一：把监听端口是 8080 的 Apache 服务全部停止

```
[root@ansible ansible]# vim ad.yml
---
- hosts: cache
  remote_user: root
  tasks:
    - shell: netstat -atunlp | awk '{print $4}' | awk '-F:' '{print $2}'
      register: result
    - service:
        name: httpd
        state: stopped
[root@ansible ansible]# ansible-playbook ad.yml

PLAY                                                                [cache]
*****
*****

TASK                                                                [Gathering Facts]
*****
*****
ok: [cache]

TASK                                                                [command]
*****
*****
```



```

changed: [cache]

TASK                                                                    [service]
*****
*****
changed: [cache]

PLAY                                                                    RECAP
*****
*****
cache                          : ok=3    changed=2    unreachable=0    failed=0

```

步骤二：when 条件判断

1) 当系统负载超过 0.7 时，则关掉 httpd

```

[root@ansible ansible]# vim when.yml
---
- hosts: cache
  remote_user: root
  tasks:
    - shell: uptime | awk '{printf("%.2f"),$(NF-2)}'
      register: result
    - service:
        name: httpd
        state: stopped
        when: result.stdout|float > 0.7

[root@ansible ansible]# ansible-playbook when.yml

PLAY                                                                    [cache]
*****

TASK                                                                    [Gathering Facts]
*****
ok: [cache]

TASK                                                                    [command]
*****
changed: [cache]

TASK                                                                    [service]
*****
changed: [cache]

PLAY                                                                    RECAP
*****
*****
cache                          : ok=3    changed=2    unreachable=0    failed=0

```

步骤三：with_items 标准循环

1) 为不同用户定义不同组

```

[root@ansible ansible]# vim add.yml
---
- hosts: web2
  remote_user: root
  tasks:

```

```
- user:
  name: "{{item.name}}"
  group: "{{item.group}}"
  password: "{{'123456'|password_hash('sha512')}}"
with_items:
  - {name: "aa", group: "users"}
  - {name: "bb", group: "mail" }
  - {name: "cc", group: "wheel"}
  - {name: "dd", group: "root" }
[root@ansible ansible]# ansible-playbook add.yml

PLAY                                                                 [web2]
*****
*****

TASK                                                                 [Gathering Facts]
*****
*****
ok: [web2]

TASK                                                                 [user]
*****
*****
changed: [web2] => (item={u'group': u'users', u'name': u'aa'})
changed: [web2] => (item={u'group': u'mail', u'name': u'bb'})
changed: [web2] => (item={u'group': u'wheel', u'name': u'cc'})
changed: [web2] => (item={u'group': u'root', u'name': u'dd'})

PLAY                                                                 RECAP
*****
*****
web2                                : ok=2    changed=1    unreachable=0    failed=0
```

2) 嵌套循环，循环添加多用户

```
[root@ansible ansible]# vim add1.yml
---
- hosts: web2
  remote_user: root
  vars:
    un: [a, b, c]
    id: [1, 2, 3]
  tasks:
    - name: add users
      shell: echo {{item}}
      with_nested:
        - "{{un}}"
        - "{{id}}"

[root@ansible ansible]# ansible-playbook add1.yml

PLAY                                                                 [web2]
*****
*****

TASK                                                                 [Gathering Facts]
*****
*****
ok: [web2]

TASK                                                                 [add users]
*****
*****
changed: [web2] => (item=[u'a', 1])
changed: [web2] => (item=[u'a', 2])
```

```
changed: [web2] => (item=[u'a', 3])
changed: [web2] => (item=[u'b', 1])
changed: [web2] => (item=[u'b', 2])
changed: [web2] => (item=[u'b', 3])
changed: [web2] => (item=[u'c', 1])
changed: [web2] => (item=[u'c', 2])
changed: [web2] => (item=[u'c', 3])
```

PLAY

RECAP

```
*****
*****
web2                                : ok=2    changed=1    unreachable=0    failed=0
```

步骤四：tags 给指定的任务定义一个调用标识

1) tags 样例

```
[root@ansible ansible]# vim adhttp.yml
---
- hosts: cache
  remote_user: root
  tasks:
    - copy:
        src: /root/httpd.conf
        dest: /etc/httpd/conf/httpd.conf
        owner: root
        group: root
        mode: 0644
        tags: config_httpd
      notify:
        - restart httpd
  handlers:
    - name: restart httpd
      service: name=httpd state=restarted
```

2) 调用方式

```
[root@ansible ansible]# ansible-playbook adhttp.yml --tags=config_httpd

PLAY [cache] *****

TASK [Gathering Facts] *****
ok: [cache]

TASK [copy] *****
ok: [cache]

PLAY RECAP *****
cache                                : ok=2    changed=0    unreachable=0    failed=0
```

3) include and roles

在编写 playbook 的时候随着项目越来越大,playbook 越来越复杂。可以把一些 play、task 或 handler 放到其他文件中,通过包含进来是一个不错的选择

roles 像是加强版的 include,它可以引入一个项目的文件和目录

一般所需的目录层级有

vars : 变量层

tasks : 任务层
handlers : 触发条件
files : 文件
template : 模板
default : 默认, 优先级最低

```
...
tasks:
  - include: tasks/setup.yml
  - include: tasks/users.yml user=plj
//users.yml 中可以通过{{ user }}来使用这些变量
handlers:
  - include: handlers/handlers.yml
```

步骤五：debug 检测

```
[root@ansible ansible]# ansible-playbook --syntax-check http.yml //检测语法
```

```
playbook: http.yml
```

```
[root@ansible ansible]# ansible-playbook -C http.yml //测试运行
```

```
[root@ansible ansible]# ansible-playbook http.yml --list-tasks
```

```
//显示要执行的工作
```

```
playbook: http.yml
```

```
play #1 (cache): cache TAGS: []
tasks:
  install one specific version of Apache TAGS: []
  lineinfile TAGS: []
  replace TAGS: []
  service TAGS: []
  copy TAGS: []
```

```
[root@ansible ansible]# vim debug.yml
```

```
---
```

```
- hosts: cache
  remote_user: root
  tasks:
    - shell: uptime |awk '{printf("%f\n",$(NF-2))}'
      register: result
    - shell: touch /tmp/isreboot
      when: result.stdout|float > 0.5
    - name: Show debug info
      debug: var=result
```

```
[root@ansible ansible]# ansible-playbook debug.yml //运行
```

```
PLAY [cache]
```

```
*****
*****
```

```
TASK [Gathering Facts]
```

```
*****
*****
```

```

ok: [cache]

TASK                                                                    [command]
*****
*****
changed: [cache]

TASK                                                                    [command]
*****
*****
skipping: [cache]

TASK          [Show          debug          info]
*****
*****
ok: [cache] => {
  "result": {
    "changed": true,
    "cmd": "uptime |awk '{printf(\"%f\\n\\n\",$(NF-2))}'\"",
    "delta": "0:00:00.005905",
    "end": "2018-09-07 12:57:51.371013",
    "failed": false,
    "rc": 0,
    "start": "2018-09-07 12:57:51.365108",
    "stderr": "",
    "stderr_lines": [],
    "stdout": "0.000000",
    "stdout_lines": [
      "0.000000"
    ]
  }
}

PLAY                                                                    RECAP
*****
*****
cache          : ok=3    changed=1    unreachable=0    failed=0

```