NSD ARCHITECTURE DAY02

1 练习1：playbook练习

1.1 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

1.2 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]
```

Top

```
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
```

Top

```
25.         dest: /var/www/html/index.html
26.
27.  [root@ansible ansible]# curl 192.168.1.56:8080
28.  hello world
29.  [root@ansible ansible]# ssh cache
30.  Last login: Fri Sep  7 09:32:05 2018 from 192.168.1.51
31.  [root@cache ~]# apachectl -t
32.  Syntax OK
```

2 案例2：变量练习

2.1 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

2.2 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

```
01.  [root@ansible ansible]# vim user.yml
02.  ---
03.  - hosts: cache
04.    remote_user: root
05.    vars:
06.      username: xiaoming
07.    tasks:
08.      - name: create user "{{username}}"
09.        user: group=wheel uid=1000 name={{username}}
10.      - shell: echo 123456 | passwd --stdin xiaoming
11.      - shell: chage -d 0 {{username}}
12.  [root@ansible ansible]# ansible-playbook user.yml   //执行结果
13.
14.  PLAY [cache] ***************************************************************
15.
16.  TASK [Gathering Facts] *****************************************************
17.  ok: [cache]
18.
19.  TASK [create user "xiaoming"] ***********************************************
```

Top

```
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
```

Top

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

3.1 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

Top

3.2 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
```

Top

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.(
...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

Top

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

Top

```
[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

4.1 问题

本案例要求：

- 把所有监听端口是8080的Apache服务全部停止

4.2 步骤

Top

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
```

Top

```
      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

```

Top

```
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])
```

Top

```
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
```

Top

```

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

```
04.
05.  TASK [Gathering Facts] ***************************************************
06.  ok: [cache]
07.
08.  TASK [copy] ***************************************************************
09.  ok: [cache]
10.
11.  PLAY RECAP *********************************************************************
12.  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

```
01.  ...
02.  tasks:
03.    - include: tasks/setup.yml
04.    - include: tasks/users.yml user=plj
05.  //users.yml 中可以通过{{ user }}来使用这些变量
06.  handlers:
07.    - include: handlers/handlers.yml
```

步骤五：debug检测

```
01.  [root@ansible ansible]# ansible-playbook --syntax-check http.yml //检测语法
02.
03.  playbook: http.yml
04.  [root@ansible ansible]# ansible-playbook -C http.yml //测试运行
05.
06.  [root@ansible ansible]# ansible-playbook http.yml --list-tasks
```

Top

```
07.  //显示要执行的工作
08.
09.  playbook: http.yml
10.
11.    play #1(cache): cache    TAGS: []
12.      tasks:
13.        install one specific version of Apache    TAGS: []
14.        lineinfile    TAGS: []
15.        replace    TAGS: []
16.        service    TAGS: []
17.        copy    TAGS: []
18.
19.
20.  [root@ansible ansible]# vim debug.yml
21.  ---
22.  - hosts: cache
23.    remote_user: root
24.    tasks:
25.      - shell: uptime | awk '{printf("%f\n",$(NF-2))}'
26.        register: result
27.      - shell: touch /tmp/isreboot
28.        when: result.stdout|float > 0.5
29.      - name: Show debug info
30.        debug: var=result
31.
32.  [root@ansible ansible]# ansible-playbook debug.yml        //运行
33.
34.  PLAY [cache] ***********************************************************
35.
36.  TASK [Gathering Facts] ***************************************************
37.  ok: [cache]
38.
39.  TASK [command] ***********************************************************
40.  changed: [cache]
41.
42.  TASK [command] ***********************************************************
43.  skipping: [cache]
44.
45.  TASK [Show debug info] ***************************************************
46.  ok: [cache] => {
47.      "result": {
```

Top

```
        "changed": true,
        "cmd": "uptime | awk '{ printf( \"%f \\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
```

Top