[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "page_top_case)

# NSD Devops DAY03

1. [案例1：准备ansible环境](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "case1)
2. [案例2：使用playbook](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "case2)
3. [案例3：执行ad-hoc命令](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "case3)
4. [案例4：playbook编程](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "case4)
5. [案例5：ansible模块开发](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY03/CASE/01/index.html" \l "case5)

## 1 案例1：准备ansible环境

### 1.1 问题

1. 创建ansible工作目录
2. 创建配置文件及主机列表文件
3. 测试在远程主机执行命令

### 1.2 步骤

实现此案例需要按照如下步骤进行。

步骤一： 安装ansible

1. [root@localhost ~]# pip3 install ansible

步骤二： 创建ansible工作目录

1. [root@localhost ~]# mkdir /root/myansi/

步骤三：创建配置文件

1. [root@localhost ~]# cd /root/myansi/
2. [root@localhost myansi]# cat ansible.cfg
3. [defaults]
4. inventory    =hosts
5. remote\_user=root

步骤四：创建声明被管理主机

1. [root@localhost myansi]# vim hosts
2. [dbservers]
3. node1.tedu.cn
4. [webservers]
5. node2.tedu.cn
6. node3.tedu.cn

步骤四：配置名称解析

1. [root@localhost myansi]# vim /etc/hosts
2. 192.168.4.1 node1.tedu.cn node1
3. 192.168.4.2 node2.tedu.cn node2
4. 192.168.4.3 node3.tedu.cn node3

步骤五：导入所有服务器的主机公钥

1. [root@localhost myansi]# ssh-keyscan 192.168.4.{1..3} node{1..3} node{1..3}.tedu.cn >> ~/.ssh/known\_hosts

步骤六：测试ansible到各服务器的连接

1. [root@localhost myansi]# ansible all -m ping –k
2. SSH password：
3. node1.tedu.cn    |    SUCCESS    =>    {
4. "changed":    false,
5. "ping":    "pong"
6. }
7. node3.tedu.cn    |    SUCCESS    =>    {
8. "changed":    false,
9. "ping":    "pong"
10. }
11. node2.tedu.cn    |    SUCCESS    =>    {
12. "changed":    false,
13. "ping":    "pong"
14. }

步骤六：在远程主机执行命令

1. [root@localhost myansi]# ansible node1.tedu.cn -m yum -a 'name=httpd state=present' –k
2. [root@localhost myansi]# ansible all -a 'id zhangsan' -k

## 2 案例2：使用playbook

### 2.1 问题

1. Playbook有两个play
2. 一个play用于在webservers安装并启动httpd服务
3. 另一个play用于在dbservers安装并启动mariadb服务

### 2.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：实现免密登陆：

1. [root@localhost myansi]# vim auth\_key.yml
2. ---
3. - name: configure authorized key
4. hosts: all        #运行执行任务（task）的目标主机
5. tasks:            #任务列表
6. - name: root key
7. authorized\_key:        #为root用户账号添加或删除 SSH authorized keys
8. user: root            #用户
9. state: present        #状态
10. key: "{{ lookup('file', '/root/.ssh/id\_rsa.pub') }}"

验证脚本执行情况：

1. [root@localhost myansi]# ansible-playbook --syntax-check auth\_key.yml        #检查语法
2. #调用密码执行所有主机的免密登录
3. [root@localhost myansi]# ansible-playbook auth\_key.yml –k
4. [root@localhost myansi]# ansible all -m ping    #查看所有主机连接情况

步骤二：配置yum

1. [root@localhost myansi]# mkdir files
2. [root@localhost myansi]# cp /etc/yum.repos.d/server.repo files/
3. [root@localhost myansi]# vim auth\_key.yml        #在文件末尾追加
4. - name: copy yum config file
5. copy:
6. src: files/server.repo # 本机目录
7. dest: /etc/yum.repos.d/ # 远程目录
8. [root@localhost myansi]# ansible-playbook auth\_key.yml    #执行脚本

步骤三：配置服务

1. [root@localhost myansi]# vim lamp.yml
2. ---
3. #在web服务器上配置httpd
4. - name: configure web service
5. hosts: webservers        #hosts文件中node2、node3主机
6. #两个任务，yum安装httpd、php、php-mysql，启服务httpd
7. tasks:
8. - name: install web app
9. yum:
10. name: "{{item}}"
11. state: present
12. with\_items:
13. - httpd
14. - php
15. - php-mysql
16. - name: config web service
17. service:
18. name: httpd
19. state: started
20. enabled: true
21. #在数据库服务器上配置mariadb
22. - name: configure db service
23. hosts: dbservers        #hosts文件中node1主机
24. #两个任务，yum安装mariadb-server，启服务mariadb
25. tasks:
26. - name: install db app
27. yum:
28. name: mariadb-server
29. state: latest
30. - name: config db serivce
31. service:
32. name: mariadb
33. state: started
34. enabled: yes

步骤四：测试脚本执行情况

1. [root@localhost myansi]# ansible-playbook lamp.yml
2. [root@localhost myansi]# ssh node2
3. Last login: Fri......
4. [root@node2 ~]# rpm –q php
5. php-5.4.16-42.el7.x86\_64
6. [root@node2 ~]# rpm –q httpd
7. httpd-2.4.6-67.el7.centos.x86\_64
8. [root@node2 ~]# systemctl status httpd
9. ....
10. Active: active(running)......
11. .....
12. [root@node2 ~]# 登出
13. [root@localhost myansi]# ssh node1
14. Last login: Fri Aug 31 11:52:44 2018 from 192.168.4.254
15. [root@node1 ~]#systemctl status mariadb
16. ....
17. Active: active(running)......
18. .....

## 3 案例3：执行ad-hoc命令

### 3.1 问题

1. 编写ansible脚本
2. 用于在远程主机执行任意命令

### 3.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写脚本

1. [root@localhost ~]# vim ansible\_adhoc.py
2. #!/usr/bin/env python
3. # coding: utf8

导入模块

1. import shutil
2. from collections import namedtuple
3. # DataLoader用于解析yaml/json/ini文件
4. from ansible.parsing.dataloader import DataLoader
5. # VariableManager用于分析ansible用到的变量
6. from ansible.vars.manager import VariableManager
7. # InventoryManager用于分析主机文件
8. from ansible.inventory.manager import InventoryManager
9. from ansible.playbook.play import Play
10. # task\_queue\_manager管理任务队列
11. from ansible.executor.task\_queue\_manager import TaskQueueManager
12. import ansible.constants as C # ansible的常量（不会变化的数据）

设置参数

1. #创建元组，将选项加入，如：connection：连接，module\_path：模块路径，forks：进程数量等
2. Options = namedtuple('Options', ['connection', 'module\_path', 'forks', 'become', 'become\_method', 'become\_user', 'check', 'diff'])
3. # 创建具体的实例对象
4. # connection有三个选择local/ssh/smart
5. # local表示在本机执行，ssh表示通过ssh协议执行，smart表示自动选择
6. options = Options(connection='smart', module\_path=['/to/mymodules'], forks=10, become=None, become\_method=None, become\_user=None, check=False, diff=False)
7. loader = DataLoader() #负责查找和读取YAML、JSON和INI文件
8. passwords = dict() # 用于存储加密密码、远程连接密码等
9. # 声明被ansible管理的主机有哪些，可以把各主机用逗号分开形成字符串
10. # 也可以使用主机清单文件路径，将路径放到列表中
11. # inventory = InventoryManager(loader=loader, sources='localhost,')
12. inventory = InventoryManager(loader=loader, sources=['myansi/hosts'])
13. #变量管理器负责合并所有不同的源，从而为每个上下文提供变量的统一视图。
14. variable\_manager = VariableManager(loader=loader, inventory=inventory)
15. #脚本执行时屏幕显示的结果结构及信息
16. play\_source = dict(
17. name="Ansible Play", # Play名称
18. # hosts='localhost', # 在哪些主机上执行命令
19. hosts='webservers', # 在上面inventory定影的myansi/hosts中查找
20. gather\_facts='no', # 不收集主机信息
21. tasks=[
22. # 以下是执行的命令
23. dict(action=dict(module='yum', args='name=httpd state=latest'), register='shell\_out'),
24. #dict(action=dict(module='debug', args=dict(msg='{{shell\_out}}')))
25. ]
26. )
27. #上面导入的对象，play\_source执行的任务有哪些，变量到的分析
28. play = Play().load(play\_source, variable\_manager=variable\_manager, loader=loader)

创建实例并执行命令

1. tqm = None
2. try:
3. #tqm是taskQueueManager任务管理器生成的实例
4. tqm = TaskQueueManager(
5. inventory=inventory,    #主机清单
6. variable\_manager=variable\_manager,    #参数管理
7. loader=loader,    #json等语法分析
8. options=options,    #选项
9. passwords=passwords,    #密码
10. )
11. result = tqm.run(play) # tqm实例中的run方法开始执行play中的任务
12. finally:
13. if tqm is not None:    #如果tqm不为none
14. tqm.cleanup()        #清理
15. shutil.rmtree(C.DEFAULT\_LOCAL\_TMP, True) #删除ansible执行任务是生成的临时目录

步骤二：测试执行脚本

1. [root@localhost ~]# python ansible\_adhoc.py
2. PLAY [Ansible Play]\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
3. \*\*\*\*\*\*\*\*
4. TASK [yum]\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
5. \*\*\*\*\*\*\*\*
6. ok: [node2.tedu.cn]
7. ok: [node3.tedu.cn]
8. 您在 /var/spool/mail/root 中有新邮件

## 4 案例4：playbook编程

### 4.1 问题

1. 编写python程序
2. 利用该程序执行前面课程中的playbook

### 4.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写脚本

1. [root@localhost ~]# vim run\_pb.py
2. #!/usr/bin/env python
3. # coding: utf8

导入模块

1. from collections import namedtuple
2. from ansible.parsing.dataloader import DataLoader
3. from ansible.vars.manager import VariableManager
4. from ansible.inventory.manager import InventoryManager
5. from ansible.executor.playbook\_executor import PlaybookExecutor

设置参数

1. # 初始化需要的对象
2. Options = namedtuple(
3. 'Options',
4. [
5. 'connection',
6. 'remote\_user',
7. 'ask\_sudo\_pass',
8. 'verbosity',
9. 'ask\_pass',
10. 'module\_path',
11. 'forks',
12. 'become',
13. 'become\_method',
14. 'become\_user',
15. 'check',
16. 'listhosts',
17. 'listtasks',
18. 'listtags',
19. 'syntax',
20. 'sudo\_user',
21. 'sudo',
22. 'diff'
23. ]
24. )
25. # 初始化需要的对象
26. ops = Options(
27. connection='smart',
28. remote\_user=None,
29. ask\_pass=None,
30. sudo\_user=None,
31. forks=5,
32. sudo=None,
33. ask\_sudo\_pass=False,
34. verbosity=5,
35. module\_path=None,
36. become=None,
37. become\_method=None,
38. become\_user=None,
39. check=False,
40. diff=False,
41. listhosts=None,
42. listtags=None,
43. listtasks=None,
44. syntax=None
45. )
46. # 用来加载解析yaml文件或JSON内容,并且支持vault的解密
47. loader = DataLoader()
48. # 设置密码,需要是dict类型
49. passwords = dict()
50. # 根据inventory加载对应变量,此处参数可以有两种格式：hosts文件或ip列表
51. inventory = InventoryManager(
52. loader=loader,
53. sources=['myansi/hosts']
54. )
55. # 管理变量的类，包括主机，组，扩展等变量
56. variable\_manager = VariableManager(
57. loader=loader,
58. inventory=inventory
59. )

创建实例并执行lamp.yml文件完成tasks任务

1. def run\_pb(pb\_path):
2. # playbooks就填写yml文件
3. playbook = PlaybookExecutor(
4. playbooks=pb\_path,
5. inventory=inventory,
6. variable\_manager=variable\_manager,
7. loader=loader,
8. options=ops,
9. passwords=passwords
10. )
11. #开始执行
12. result = playbook.run()
13. return result
14. if \_\_name\_\_ == '\_\_main\_\_':
15. run\_pb(pb\_path=['myansi/lamp.yml'])

步骤二：测试执行脚本

1. [root@localhost ~]# python ansible\_adhoc.py
2. PLAY [configure web service]\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
3. \*\*\*\*\*\*\*
4. TASK [Gathering Facts]\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
5. \*\*\*\*\*\*\*
6. ok: [node2.tedu.cn]
7. ok: [node3.tedu.cn]
8. TASK [install web app]\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
9. \*\*\*\*\*\*\*
10. ok: [node3.tedu.cn] => (item=[u‘httpd’， u‘php’， u‘php-mysql
11. ’])
12. .........
13. .........

## 5 案例5：ansible模块开发

### 5.1 问题

修改构建工程，要求如下：

1. 编写ansible模块，使用shutil模块拷贝文件
2. 数据源用变量名yuan
3. 数据目标变量用mudi

### 5.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写脚本

创建模块路径

1. [root@localhost myansi]# mkdir mylib
2. [root@localhost myansi]# cd mylib
3. #设置ansible查找模块的路径
4. [root@localhost myansi]# export ANSIBLE\_LIBRARY=$(pwd)/mylib

创建模块，模块用于将管理机上的文件拷贝到目标主机的指定目录

1. [root@localhost mylib]# vim mylib/rcopy.py
2. #!/usr/bin/env python

导入所需要的模块

1. import shutil
2. from ansible.module\_utils.basic import AnsibleModule

创建模块入口

1. def main():
2. ##使用AnsibleModule类中的argument\_spec来接收yuan、mudi两个参数，参数必须提供并且是字符串类型
3. mokuai = AnsibleModule(
4. argument\_spec=dict(
5. yuan=dict(required=True, type='str'),
6. mudi=dict(required=True, type='str')
7. )
8. )

执行动作

1. #将yuan拷贝到mudi
2. shutil.copy(mokuai.params['yuan'], mokuai.params['mudi'])

返回结果

1. #拷贝完成后，返回json数据
2. mokuai.exit\_json(change=True)

编写主程序代码

1. if \_\_name\_\_ == '\_\_main\_\_':
2. main()
3. [root@localhost myansi]# ansible dbservers -m rcopy -a "yuan=/etc/hosts mudi=/opt"
4. node1.tedu.cn | SUCCESS => {
5. “change”: true，
6. “changed”: false
7. }