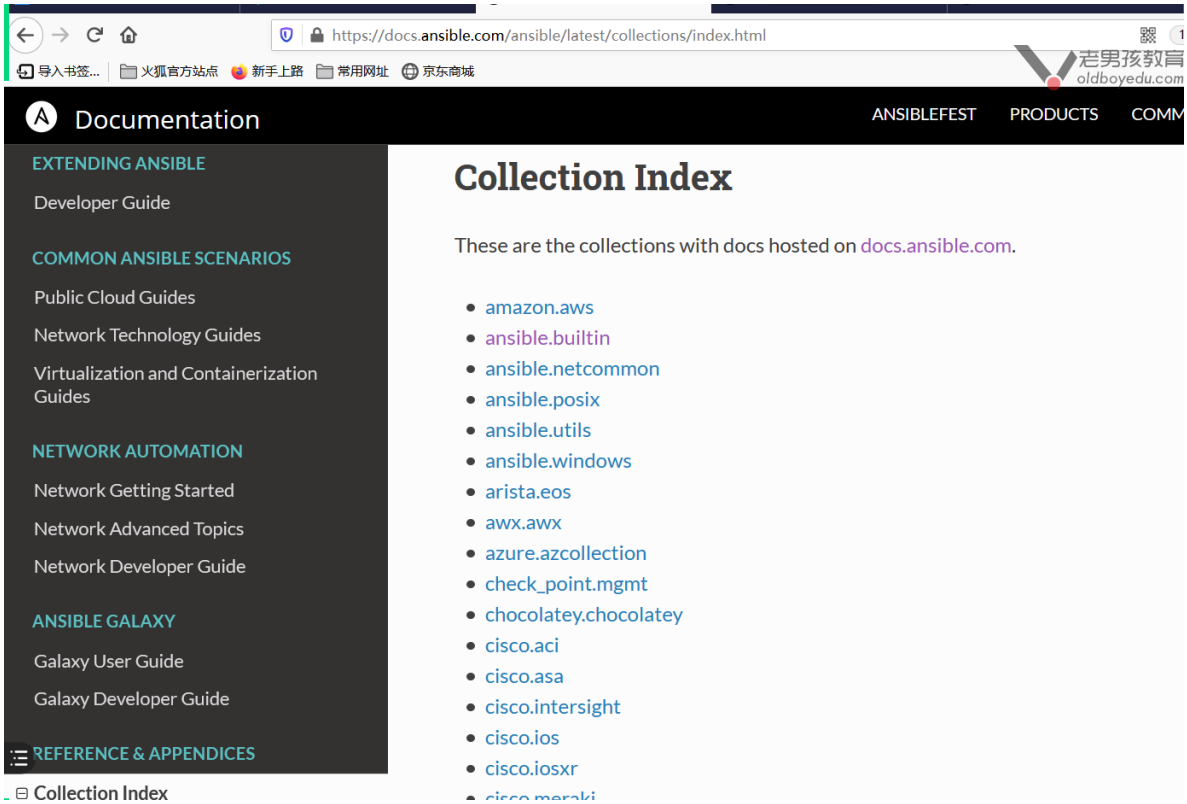
02 Ansible Ad-Hoc

Ansible模块最全的参考

- 传送门传送门
- 传送门精确
- 命令行查询模块:`ansible-doc -s copy`




```
ansible-doc -l |grep ali
ansible-doc -l |grep copy
ansible-doc -l |grep yum
ansible-doc yum
```

*1.什么是ad-hoc

ad-hoc简而言之就是“临时命令”，执行完即结束，并不会保存*

*2.ad-hoc模式的使用场景，

临时获取主机的数据、状态。

比如在多台机器上查看某个进程是否启动，或拷贝指定文件到本地，等等*

3. ✨✨ad-hoc模式的命令使用，ansible 'oldboy' -m command -a 'df -h'，含义如下图

*4.使用ad-hoc执行一次远程命令，注意观察返回结果的颜色

- 绿色绿色: 代表被管理端主机没有被修改(成功)
- 黄色黄色: 代表被管理端主机发现变更（成功）
- 红色红色: 代表出现了故障，注意查看提示（失败）
- 紫色(粉色):紫色(粉色) 警告信息,建议

```
ansible  all  -m    ping
```

5. ad-hoc模式的常用模块有如下:

```
command                 # 执行shell命令(不支持管道等特殊字符) 管道
*  > . ..
shell                   # 执行shell命令 支持特殊符号
script                  # 执行shell脚本

yum_repository          # 配置yum仓库 yum源
yum                     # 安装软件
```

```
copy                    # 变更配置文件     远程复制
file                    # 建立目录或文件

service                 # 启动与停止服务     设置开机自启动
systemctl
mount                   # 挂载设备 磁盘 光盘   nfs ....
cron                    # 定时任务 设置/删除定时任务

firewalld                # 防火墙
iptables ※               # 防火墙

get_url                 # 下载软件 wget

。。。。。
压缩解压....
```

6.使用过程中需要先了解ansible-doc帮助手册

```
[root@m01 ~]# ansible-doc -l          # 查看所有模块说明
[root@m01 ~]# ansible-doc copy        # 表示指定模块方法
[root@m01 ~]# ansible-doc -s copy     # 表示指定模块参数
```

1.执行命令模块

1.command命令模块，不支持重定向或管道

command模块		
直接写上命令即可，不支持特殊符号 | > >> {} *		

```
# 默认模块，执行命令
[root@m01 ~]# ansible oldboy  -a "hostname"
```

2.shell模块，如果需要一些管道操作，则使用shell

- 使用起来与command一致, shell模块支持管道 特殊符号

```
[root@m01 ~]# ansible oldboy -m shell -a "ifconfig|grep
eth0" -f 50



[root@m01 ~]# ansible web  -i hosts   -m command   -a 'ip
a |grep eth0'
172.16.1.10 | FAILED | rc=255 >>
Command "|grep" is unknown, try "ip address help".non-zero
return code
172.16.1.7 | FAILED | rc=255 >>
Command "|grep" is unknown, try "ip address help".non-zero
return code
172.16.1.9 | FAILED | rc=255 >>
Command "|grep" is unknown, try "ip address help".non-zero
return code
172.16.1.8 | FAILED | rc=255 >>
Command "|grep" is unknown, try "ip address help".non-zero
return code
[root@m01 ~]# ansible web  -i hosts   -m shell   -a 'ip a
|grep eth0'
172.16.1.7 | CHANGED | rc=0 >>
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast
state DOWN group default qlen 1000
172.16.1.8 | CHANGED | rc=0 >>
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP group default qlen 1000
    inet 10.0.0.8/24 brd 10.0.0.255 scope global eth0
172.16.1.10 | CHANGED | rc=0 >>
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP group default qlen 1000
    inet 10.0.0.10/24 brd 10.0.0.255 scope global eth0
172.16.1.9 | CHANGED | rc=0 >>
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP group default qlen 1000
    inet 10.0.0.9/24 brd 10.0.0.255 scope global eth0
```

3.script脚本模块

- 说明:

    - **把对应脚本传输过去**
    - **运行对应的脚本**
- 应用： 批量执行脚本

```
# 编写脚本
[root@m01 ~]# mkdir -p /server/scripts
[root@m01 ~]# cat /server/scripts/yum.sh
#!/usr/bin/bash
yum install -y iftop

#在本地运行模块，等同于在远程执行，不需要将脚本文件进行推送目标主机执行
[root@m01 ~]# ansible oldboy -m script -a "/server/scripts/yum.sh"

/usr/bin/python2 -Es /usr/sbin/tuned -l -P
/usr/sbin/sshd -D
 \_ sshd: root@pts/0
 |   \_ /bin/sh -c  /root/.ansible/tmp/ansible-tmp-1622793591.59-27092-255580592839178/yum.sh && sleep 0
 |       \_ /bin/bash /root/.ansible/tmp/ansible-tmp-1622793591.59-27092-255580592839178/yum.sh
 |           \_ /usr/bin/python /usr/bin/yum install ipvsadm
```

- command 模块用于执行简易的命令，不包含特殊符号，管道，重定向，通配符
- shell 与command 类似，　　支持含特殊符号，管道，重定向，通配符
- script 分发脚本并执行脚本

2.软件管理模块

yum

yum模块		
name=	**指定软件名字** sl cowsay 软件名字-版本	
state= ✲✲✲✲✲	**状态**(present 或 installed 安装软件) absent或removed 删除 latest更新	
download_only=true	仅下载,不安装.	
enablerepo	安装的时候**临时**开启被关闭的yum源.	
exclude	排除	

```
ansible webserver -m yum -a "name=httpd state=present" -i
hosts
ansible webserver -m yum -a "name=httpd state=absent" -i
hosts



ansible  nfs -m yum -a 'name=sl state=installed'
ansible  nfs -m yum -a 'name=sl,cowsay state=installed'
```

```
#示例一、安装当前最新的Apache软件，如果存在则不安装
[root@ansible ~]# ansible webserver -m yum -a "name=httpd
state=present" -i hosts
ansible lb -i hosts   -m yum  -a 'name=httpd
state=present'


Whether to install (present or installed, latest), or
remove (absent or removed) a package.
```

```
#示例二、安装当前最新的Apache软件，通过epel仓库安装
[root@ansible ~]# ansible webserver -m yum -a "name=httpd
state=present enablerepo=epel" -i hosts

Repoid of repositories to enable for the install/update
operation. 为了yum安装启动特点的yum源

These repos will not persist beyond the transaction.    #
在本次操作中生效
When specifying multiple repos, separate them with a ",".
 #如果需要指定多个通过逗号分割.

[root@m01 ~]# ansible 172.16.1.5 -i hosts   -m yum  -a
'name=cowsay state=present'
172.16.1.5 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "msg": "No package matching 'cowsay' found available,
installed or updated",
    "rc": 126,
    "results": [
        "No package matching 'cowsay' found available,
installed or updated"
    ]
}
[root@m01 ~]# ansible 172.16.1.5 -i hosts   -m yum  -a
'name=cowsay state=present '
[root@m01 ~]# ansible 172.16.1.5 -i hosts   -m yum  -a
'name=cowsay state=present enablerepo=epel'
172.16.1.5 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "changes": {
        "installed": [
            "cowsay"
```

```
        ]
    },
    "msg": "",
    "rc": 0,
    "results": [
        "Loaded plugins: fastestmirror\nLoading mirror
speeds from cached hostfile\n * base: mirrors.aliyun.com\n
* extras: mirrors.aliyun.com\n * updates:
mirrors.aliyun.com\nResolving Dependencies\n--> Running
transaction check\n---> Package cowsay.noarch 0:3.04-4.el7
will be installed\n--> Finished Dependency
Resolution\n\nDependencies
Resolved\n\n================================================
================================\n Package          Arch
           Version                  Repository
Size\n================================================
============================\nInstalling:\n cowsay
  noarch           3.04-4.el7              epel
42 k\n\nTransaction
Summary\n================================================
==============================\nInstall  1
Package\n\nTotal download size: 42 k\nInstalled size: 77
k\nDownloading packages:\nRunning transaction
check\nRunning transaction test\nTransaction test
succeeded\nRunning transaction\n  Installing : cowsay-
3.04-4.el7.noarch                                      1/1
\n  Verifying  : cowsay-3.04-4.el7.noarch
                  1/1 \n\nInstalled:\n  cowsay.noarch
0:3.04-4.el7
     \n\nComplete!\n"
    ]
}
```


#示例三、通过互联网的rpm进行安装

```
[root@ansible ~]# ansible webserver -m yum -a
"name=https://mirrors.tuna.tsinghua.edu.cn/zabbix/zabbix/5
.0/rhel/7/x86_64/zabbix-agent-5.0.0-1.el7.x86_64.rpm
state=present" -i hosts




#示例四、安装最新版本的Apache软件，如果存在则更新Apache (了解)
[root@ansible ~]# ansible webserver -m yum -a "name=httpd
state=latest" -i hosts

#示例五、更新所有的软件包，但排除和kernel相关的
[root@ansible ~]# ansible 172.16.1.41 -m yum -a "name=*
state=latest exclude=kernel" -i hosts

yum  -y update  #升级系统所有的软件包  name=* state=latest
#exclude 排除




#示例六、删除Apache软件
[root@ansible ~]# ansible webserver -m yum -a "name=httpd
state=absent" -i hosts


#安装多个软件包
[root@m01 ~]# ansible web  -m yum -a
"name=tree,cowsay,lrzsz state=installed" -i hosts
```

3.yum安装软件模块

```
[root@m01 ~]# ansible oldboy -m yum -a "name=httpd
state=installed"
name          #指定要安装的软件包名称
state         #指定使用yum的方法
    installed，present              #安装软件包
    removed，absent                 #移除软件包
    latest                          #安装最新软件包
list=ansible                       #列出当前仓库可用的软件包   yum
list  ansible  查找软件包
enablerepo 开启某个yum源
disablerepo="epel,zabbix"          #安装软件时，不从哪些仓库获取

download_only=true                 #仅下载软件包，不安装
```

yum源

yum_repository yum源的模块	**yum源配置文件**	
name	[nginx-stable]	
description	name=nginx stable repo	
baseurl与右边一致	baseurl=http://nginx.org/packages/centos/$releasever/$basearch/	
enabled=yes	enabled=1	
gpgcheck=yes	gpgcheck=1	
gpgkey=与右边一致	gpgkey=https://nginx.org/keys/nginx_signing.key	
file=nginx	nginx.repo	

yum_repository yum源的模块	yum源配置文件	
state		

```
yum_repository

[root@web01 ~]# cat /etc/yum.repos.d/nginx.repo
[nginx-stable]
name=nginx stable repo
baseurl=http://nginx.org/packages/centos/$releasever/$base
arch/
gpgcheck=1
enabled=1
gpgkey=https://nginx.org/keys/nginx_signing.key
module_hotfixes=true

[root@m01 ~]# cat  /etc/yum.repos.d/epel.repo
[epel]
name=Extra Packages for Enterprise Linux 7 - $basearch
baseurl=http://mirrors.aliyun.com/epel/7/$basearch
failovermethod=priority
enabled=1
gpgcheck=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7




#yum源模块

name       #yum源的名字
baseurl    #???
file       #指定yum配置文件的路径和名称   注意不需要以.repo结尾   默
认使用 name的内容作为文件名
enabled yes/no    是否开启yum源 默认是 yes 开启
```

```
state     #absent(删除)/present(配置 安装 这个是默认的)
description #描述信息


[php]    #yum_repository  -a name
baseurl = http://us-east.repo.webtatic.com/yum/el7/x86_64/
 #yum_repository  -a baseurl
enabled = 0         #yum_repository  -a enabled
name = php repo     #yum_repository  -a description




#给 lb负载均衡 设置 php源  状态关闭

[root@m01 ~]# ansible lb -i hosts  -m yum_repository  -a
'name=php description="php repo" baseurl="http://us-
east.repo.webtatic.com/yum/el7/x86_64/"      enabled=no
state=present'
172.16.1.5 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "repo": "php",
    "state": "present"
}
172.16.1.6 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "repo": "php",
    "state": "present"
}
[root@m01 ~]# ansible
[root@m01 ~]# ansible lb -i hosts   -a 'ls -l
/etc/yum.repos.d/'
172.16.1.5 | CHANGED | rc=0 >>
total 48
```

```
-rw-r--r--. 1 root root 2523 Apr 25 10:49 CentOS-Base.repo
-rw-r--r--. 1 root root 1309 Apr  8  2020 CentOS-CR.repo
-rw-r--r--. 1 root root  649 Apr  8  2020 CentOS-
Debuginfo.repo
-rw-r--r--. 1 root root  314 Apr  8  2020 CentOS-
fasttrack.repo
-rw-r--r--. 1 root root  630 Apr  8  2020 CentOS-
Media.repo
-rw-r--r--. 1 root root 1331 Apr  8  2020 CentOS-
Sources.repo
-rw-r--r--. 1 root root 7577 Apr  8  2020 CentOS-
Vault.repo
-rw-r--r--. 1 root root  616 Apr  8  2020 CentOS-x86_64-
kernel.repo
-rw-r--r--  1 root root  664 Jun  4 16:39 epel.repo
-rw-r--r--  1 root root  398 May 18 09:49 nginx.repo
-rw-r--r--  1 root root   94 Jun  5 09:48 php.repo
172.16.1.6 | CHANGED | rc=0 >>
total 48
-rw-r--r--. 1 root root 2523 Apr 25 10:49 CentOS-Base.repo
-rw-r--r--. 1 root root 1309 Apr  8  2020 CentOS-CR.repo
-rw-r--r--. 1 root root  649 Apr  8  2020 CentOS-
Debuginfo.repo
-rw-r--r--. 1 root root  314 Apr  8  2020 CentOS-
fasttrack.repo
-rw-r--r--. 1 root root  630 Apr  8  2020 CentOS-
Media.repo
-rw-r--r--. 1 root root 1331 Apr  8  2020 CentOS-
Sources.repo
-rw-r--r--. 1 root root 7577 Apr  8  2020 CentOS-
Vault.repo
-rw-r--r--. 1 root root  616 Apr  8  2020 CentOS-x86_64-
kernel.repo
-rw-r--r--. 1 root root  664 Apr 25 10:49 epel.repo
-rw-r--r--  1 root root  398 May 26 11:43 nginx.repo
-rw-r--r--  1 root root   94 Jun  5 09:48 php.repo
[root@m01 ~]#
[root@m01 ~]# ansible lb -i hosts   -a 'cat
/etc/yum.repos.d/php.repo'
172.16.1.5 | CHANGED | rc=0 >>
[php]
```

```
baseurl = http://us-east.repo.webtatic.com/yum/el7/x86_64/
enabled = 0
name = php repo
172.16.1.6 | CHANGED | rc=0 >>
[php]
baseurl = http://us-east.repo.webtatic.com/yum/el7/x86_64/
enabled = 0
name = php repo
```

3.文件管理模块

ansible文件管理模块，主要涉及copy文件拷贝、file文件创建、get_url文件下载

file文件创建模块

- 文件,目录 创建,删除
-

file模块		
path	路径或文件	
state	file模块的state状态，对应不同的功能 directory 创建目录 touch 创建文件 link 创建软连接 absent 删除	
onwer		
group		
mode		
recurse	recurse=yes 只有 state 为 directory的时候 才能使用.	

```
# 1. 创建目录
[root@m01 ~]# ansible web     -m  file  -a
'path=/code/src/nginx    state=directory'  -i hosts

# 2. 创建文件
ansible web     -m  file  -a
 'path=/code/src/nginx/lidaoav.com     state=touch'  -i
hosts
  709  ansible web     -m  file  -a
 'path=/code/src/nginx/lidaoav.com     state=touch'  -i
hosts


# 3. 递归修改权限 所有者
ansible web     -m  file  -a  'path=/code/src/
state=directory owner=nobody mode=600 recurse=yes'  -i
hosts
```

```
#1.创建目录
[root@m01 ~]# ansible oldboy -m file -a "path=/tmp/oldboy
state=directory"

#2.创建文件
[root@m01 ~]# ansible oldboy -m file -a "path=/tmp/tt
state=touch mode=555 owner=root group=root"

#3.递归授权权限
[root@m01 ~]# ansible oldboy -m file -a "path=/data
owner=oldboylinux.cn group=oldboylinux.cn recurse=yes"

path            #指定远程主机目录或文件信息
recurse         #递归授权
state           #状态
    directory   #在远端创建目录
    touch       #在远端创建文件
    link        #link或hard表示创建链接文件
    absent      #表示删除文件或目录
    mode        #设置文件或目录权限
    owner         #设置文件或目录属主信息
    group       #设置文件或目录属组信息
```

copy文件拷贝模块

- 远程拷贝

copy模块		
src	从哪里来 源（ansible本地目录）	
dest	到哪里去 目标 （目标服务器目录）	
backup	是否开启备份功能,如果目标存在,覆盖之前进行备份.	
onwer		
group		
mode		

```
#1.拷贝文件文件至被控节点
[root@m01 ~]# ansible oldboy -m copy -a "src=/etc/hosts
dest=/tmp/test.txt"

#2.对远端已有文件进行备份，按照时间信息备份
[root@m01 ~]# ansible oldboy -m copy -a "src=/etc/hosts
dest=/tmp/test.txt backup=yes"


#3.复制目录 并修改所有者与权限
ansible web  -m copy    -a 'src=/etc/sysconfig/network-
scripts/  dest=/tmp/    owner=nobody  group=nobody
mode=600'  -i  hosts

#4 content 内容 写入文件内容    重定向 >
ansible web  -m copy    -a 'content="oldboylinux.cn"
dest=/tmp/lidao.txt'  -i  hosts
ansible web     -a 'cat  /tmp/lidao.txt'  -i  hosts
```

```
#3.向被控端主机写入数据，并且会覆盖远端文件内原有数据信息
[root@m01 ~]# ansible oldboy -m copy -a
"content='oldboylinux.cn' dest=/tmp/oldboy"
```

```
src             #推送数据的源文件信息
dest            #推送数据的目标路径
backup          #对推送传输过去的文件，进行备份
content         #直接批量在被管理端文件中添加内容
group           #将本地文件推送到远端，指定文件属组信息
owner           #将本地文件推送到远端，指定文件属主信息
mode            #将本地文件推送到远端，指定文件权限信息
```

get_url文件下载模块

- ansible中的wget命令

```
#1.通过get_url下载文件或者软件
[root@m01 ~]# ansible webservers -m get_url -a
"url=http,https  dest=/opt mode=0777" -i ./hosts
ansible web   -m get_url  -a
'url=https://mirrors.tuna.tsinghua.edu.cn/zabbix/zabbix/5.
0/rhel/7/x86_64/zabbix-get-5.0.0-1.el7.x86_64.rpm
dest=/tmp/ '




#2.下载一个文件前先进行md5校验，通过则下载，不通过则失败
ansible webservers -m get_url -a "url=http,https
dest=/opt mode=0777 checksum=md5:76eb3af80ffd" -i ./hosts
url          #文件在网络上的具体位置
dest         #下载到被控端的哪个目录下
checksum     #校验(md5  sha256)
```

4.服务管理模块

- **systemd** (systemctl命令)模块
- service (5678)

systemd

```
#01 开、关服务
systemctl stop  crond



#02 开机自启动服务 （关闭）

###systemd
开启或关闭 服务
```

```
ansible all   -m systemd   -a  'name=crond  state=stopped'
ansible all   -m systemd   -a  'name=crond  state=started'

ansible all   -m systemd   -a  'name=crond  state=reloaded
或restarted'


###开机自启动


ansible all   -m systemd   -a  'name=crond  enabled=yes '


###开机自启动并启动服务
ansible all   -m systemd   -a  'name=crond  enabled=yes
state=started'



daemon_reload 未来我们修改了 systemctl对应的配置的时候 需要执行
```

ansible管理服务的启动与停止，使用service

- 实现服务开启关闭/重启， 开机自启动

```
#1.启动crond服务，并加入开机自启
```

```
[root@m01 ~]# ansible webservers -m service -a "name=crond
state=started enabled=yes"


[root@m01 ~]# ansible all  -i hosts   -m  service  -a
'name=crond state=started  enabled=yes '  -f 20




#2.停止crond服务，并删除开机自启
[root@m01 ~]# ansible webservers -m service -a "name=crond
state=stopped enabled=no"


[root@m01 ~]# ansible lb  -i hosts   -m  service  -a
'name=crond state=stopped  enabled=no '  -f 20


#3.重启crond服务
[root@m01 ~]# ansible webservers -m service -a "name=crond
state=restarted"


#4.重载crond服务  优雅的重启 重新读取配置文件
[root@m01 ~]# ansible webservers -m service -a "name=crond
state=reloaded"

name         # 定义要启动服务的名称
state        # 指定服务状态
    started      #启动服务
    stopped      #停止服务
    restarted    #重启服务
    reloaded     #重载服务
enabled          #开机自启
```

模块对比	systemd(7 8)	service (5678)
服务名称	name	name
状态	state	state
是否开机自启动	enabled	enabled
系统重新读取system配置	daemon_reload	无
指定运行级别	无	runlevel
推荐与建议	centos 7 8 rockylinux 使用	适用于C5 6

5.用户管理模块

ansible管理用户与组使用user、group模块

- oldigrl uid 888 gid 888

1.group组模块

```
[root@m01 ~]# ansible oldboy -m group -a "name=oldgirl
gid=888"

name              #指定创建的组名
gid               #指定组的gid
state
    absent        #移除远端主机的组
    present       #创建远端主机的组（默认）
```

8.user模块

```
#1.创建用户指定uid和gid，不创建家目录也不允许登陆
[root@m01 ~]# ansible oldboy -m user -a "name=oldgirl
uid=888 group=888 shell=/sbin/nologin create_home=no"
```

```
#2.删除用户   指定用户名即可
userdel

[root@m01 ~]# ansible webservers -m user -a "name=tmd
state=absent" -i ./hosts

#3.给新创建的用户生成ssh密钥对
[root@m01 ~]# ansible webservers -m user -a "name=oo
uid=6677 group=adm generate_ssh_key=yes ssh_key_bits=2048
ssh_key_file=.ssh/id_rsa" -i ./hosts

generate_ssh_key=yes
ssh_key_bits=2048
ssh_key_file=.ssh/id_rsa  #私钥



#4.将明文密码进行hash加密，然后进行用户创建

passwd oldboy
1   #明文密码

1   #加密后是
fdslkjalkdsjflklkjakfdslafdsakjadsfsfdsafdsafdsafdsa

[root@m01 ~]# ansible localhost -m debug -a "msg={{
'123456' | password_hash('sha512', 'salt') }}"
localhost | SUCCESS => {
    "msg": "$6$salt$MktMKPZJ6t59"
}



[root@m01 ~]# ansible webservers -m user -a 'name=xlw
password=$6$salt$MktMKPZJ6t59 create_home=yes
shell=/bin/bash' -i ./hosts

uid             #指定用户的uid
group           #指定用户组名称
groups          #指定附加组名称
```

```
password          #给用户添加密码（记得单引号）     -a
"name=oldboy  password='加密后的密码'"
shell             #指定用户登录shell
create_home       #是否创建家目录
state             #present /absent
```

6.定时任务模块

1.crond定时任务模块

#注释说明	# -a name					
*	*	*	*	*	/bin/sh /server/scripts/yum.sh &>/dev/null	
minute	hour	day	month	weekday	job	state

```
# 正常使用crond服务（默认没写的时间都算*表示）
[root@m01 ~]# crontab -l
#yum install  脚本
* * * * *  /bin/sh /server/scripts/yum.sh   &>/dev/null


-m  cron
-a
name   #必须要添加一个
minute   hour    day   month    weekday     job
state  present(添加 默认)/absent(删除)

*         *      *      *         *          /bin/sh
/server/scripts/yum.sh   &>/dev/null
minute   hour    day   month    weekday     job


# 使用ansible添加一条定时任务
```

```
[root@m01 ~]# ansible webservers -m cron -a "minute=*
hour=* day=* month=* weekday=* job='/bin/sh test.sh'"
[root@m01 ~]# ansible webservers -m cron -a "job='/bin/sh
test.sh'"

# 设置定时任务注释信息，防止重复，name设定
[root@m01 ~]# ansible webservers -m cron -a "name='cron01'
job='/bin/sh test.sh'"

# 删除相应定时任务
[root@m01 ~]# ansible webservers -m cron -a "name='ansible
cron02' minute=0 hour=0 job='/bin/sh test.sh'
state=absent"

# 注释相应定时任务，使定时任务失效
[root@m01 scripts]# ansible oldboy -m cron -a
"name='ansible cron01' minute=0 hour=0 job='/bin/sh
test.sh' disabled=yes"



[root@m01 ~]# ansible lb -i hosts  -m  cron -a
'name="sync02" minute="*/2" job="/sbin/ntpdate
ntp1.aliyun.com  &>/dev/null"   '
172.16.1.6 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "envs": [],
    "jobs": [
        "sync  time by lidao996 ",
        "sync  time by lidao996",
        "sync02"
    ]
}
172.16.1.5 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
```

```
    "envs": [],
    "jobs": [
        "sync  time by lidao996 ",
        "sync  time by lidao996",
        "sync02"
    ]
}
[root@m01 ~]#
[root@m01 ~]# ansible lb -i hosts -a 'crontab -l'
172.16.1.6 | CHANGED | rc=0 >>
#Ansible: sync  time by lidao996
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync  time by lidao996
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync02
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
172.16.1.5 | CHANGED | rc=0 >>
#Ansible: sync  time by lidao996
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null

#Ansible: sync  time by lidao996
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync02
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
[root@m01 ~]# ansible lb -i hosts  -m  cron -a
'name="sync02" minute="*/2" job="/sbin/ntpdate
ntp1.aliyun.com  &>/dev/null" disabled=yes  '
[root@m01 ~]# ansible lb -i hosts  -m  cron -a
'name="sync02" minute="*/2" job="/sbin/ntpdate
ntp1.aliyun.com  &>/dev/null" disabled=yes  '
172.16.1.6 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "envs": [],
    "jobs": [
        "sync  time by lidao996 ",
        "sync  time by lidao996",
        "sync02"
    ]
```

```
}
172.16.1.5 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "envs": [],
    "jobs": [
        "sync  time by lidao996 ",
        "sync  time by lidao996",
        "sync02"
    ]
}
[root@m01 ~]#
[root@m01 ~]#
[root@m01 ~]#
[root@m01 ~]# ansible lb -i hosts -a 'crontab -l'
172.16.1.6 | CHANGED | rc=0 >>
#Ansible: sync  time by lidao996
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync  time by lidao996
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync02
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
172.16.1.5 | CHANGED | rc=0 >>
#Ansible: sync  time by lidao996
*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null

#Ansible: sync  time by lidao996
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null
#Ansible: sync02
#*/2 * * * * /sbin/ntpdate   ntp1.aliyun.com  &>/dev/null



-m cron
-a
name #指定名字
minute
hour
day
```

```
month
weekday
state  present/absent
disabled  是否注释
```

7.磁盘挂载模块

9.mount挂载模块

```
#在backup服务器上安装nfs
#配置
#创建目录 修改所有者
#启动服务并开机自启动
#backup上面进行挂载(本地测试)
#web服务器进行挂载


#在backup服务器上安装nfs
ansible 172.16.1.41 -i hosts    -m yum  -a 'name=nfs-utils
state=present'

##配置
cat  /etc/exports
/data-lidao/   172.16.1.0/24(rw,all_squash)  #默认压缩为
nfsnobody用户

ansible 172.16.1.41  -i hosts   -m  copy   -a
'content="/data-lidao/   172.16.1.0/24(rw,all_squash)"
dest=/etc/exports backup=yes'

#创建目录 修改所有者
[root@m01 ~]# ansible  172.16.1.41  -m file  -a
'path=/data-lidao/  owner=nfsnobody  group=nfsnobody
state=directory ' -i hosts

#启动服务并开机自启动
ansible  172.16.1.41 -i hosts   -m service  -a
'name=rpcbind  state=started  enabled=yes'
ansible  172.16.1.41 -i hosts   -m service  -a 'name=nfs
state=started  enabled=yes'
```

```
#backup上面进行挂载(本地测试)
ansible  172.16.1.41 -i hosts   -m mount  -a
'src=172.16.1.41:/data-lidao/  path=/mnt/  fstype=nfs
state=mounted'

#web服务器进行挂载
挂载到web服务器的 /code/upload/img
[root@m01 ~]# ansible web -i hosts  -m mount    -a
'src=172.16.1.41:/data-lidao    path=/code/upload/img
fstype=nfs  state=mounted'









#
#10.0.0.7作为nfs服务端，10.0.0.8作为nfs客户端挂载
[root@m01 ~]# ansible web01 -m yum -a 'name=nfs-utils
state=present' -i ./hosts
[root@m01 ~]# ansible web01 -m file -a 'path=/data
state=directory' -i ./hosts
[root@m01 ~]# ansible web01 -m copy -a 'content="/data
172.16.1.0/24(rw,sync,no_all_squash)" dest=/etc/exports' -
i ./hosts
[root@m01 ~]# ansible web01 -m systemd -a "name=nfs
state=started enabled=yes" -i ./hosts

#配置挂载
[root@m01 ~]# ansible web02 -m mount -a
"src=172.16.1.7:/data path=/data fstype=nfs opts=defaults
state=present"
[root@m01 ~]# ansible web02 -m mount -a
"src=172.16.1.7:/data path=/data fstype=nfs opts=defaults
state=mounted"
[root@m01 ~]# ansible web02 -m mount -a
"src=172.16.1.7:/data path=/data fstype=nfs opts=defaults
state=unmounted"
```

```
[root@m01 ~]# ansible web02 -m mount -a
"src=172.16.1.7:/data path=/data fstype=nfs opts=defaults
state=absent"



-m mount
-a
src    指定源
path   指定目标 挂载点
fstype   指定文件系统类型 nfs
state

present      # 仅修改配置       开机挂载，仅将挂载配置写
入/etc/fstab
mounted      # 挂载+修改配置     挂载设备，并将配置写入/etc/fstab

unmounted    # 卸载设备，不会清除/etc/fstab写入的配置
absent       # 卸载设备，会清理/etc/fstab写入的配置

remounted    #重新挂载
```

- 传送门 mount模块

8.防火墙管理模块 主要看iptables

Linux下防火墙主要分为Selinux与Firewalld

1.Selinux防火墙

```
[root@m01 ~]# ansible webservers -m selinux -a
"state=disabled" -i ./hosts
```

2.firewalld防火墙

```
[root@m01 ~]# ansible webservers -m systemd -a
"name=firewalld state=started" -i ./hosts
```

```
[root@m01 ~]# ansible webservers -m firewalld -a
"service=http immediate=yes permanent=yes state=enabled" -
i ./hosts
[root@m01 ~]# ansible webservers -m firewalld -a
"port=8080-8090/tcp immediate=yes permanent=yes
state=enabled" -i ./hosts

service                 #指定开放或关闭的服务名称
port                    #指定开放或关闭的端口
masquerade              #开启地址伪装
immediate               #临时生效
permanent               #是否添加永久生效
state                   #开启或是关闭

zone                    #指定配置某个区域
rich_rule               #配置富规则
source                  #指定来源IP
```

3. iptables模块

```
iptables    -t filter  -I INPUT  -s 10.0.0.0/24   -p tcp
--dport 3306    -j DROP
iptables    -t filter  -I INPUT  -s 10.0.0.0/24   -p tcp
--dport 3306    -j DROP

-m iptables action=insert -a table=filter  chain=INPUT
 source=10.0.0.0/24 protocol=tcp destination_port=3306
jump=DROP

action  append(默认)/insert

  664  ansible  172.16.1.51 -i hosts   -m iptables -a '
table=filter action=insert chain=INPUT  source=10.0.0.0/24
protocol=tcp destination_port=3306   jump=DROP'
  666  ansible  172.16.1.51 -i hosts   -m iptables -a '
table=filter action=insert chain=INPUT  source=172.16.1.61
protocol=tcp destination_port=3306   jump=DROP'
  669  ansible  172.16.1.51 -i hosts   -m iptables -a '
table=filter action=insert chain=INPUT  source=172.16.1.61
protocol=tcp destination_port=3306   jump=DROP
state=absent'
```

```
-m iptables
-a
table                #-t
action               #默认是append追加-A  insert插入-I
chain                #指定链
source               #-s 指定源ip    ※※※※※
destination          #-d  指定目标ip
protocal             #-p 指定协议
source_port          #--sport指定源端口
destination_port     #--dport指定目标端口 ※※※※
jump                 #-j DROP/ACCEPT
state                #present(默认,添加规则)   absent(删除)
```

```
#这可以使用nginx  db01 backup nfs
#ansible  ad-hoc练习案例
#  nfs01


1.安装nginx服务                #yum_repository/yum
2.编写简单网页测试内容         #copy content
3.启动服务不加入开机自启        #systemd/service
4.放行对应的端口                #iptables

1.安装nginx服务
#yum_repository
ansible 172.16.1.31  -i hosts   -m  yum_repository  -a
 'name=nginx  description="nginx repo"
baseurl=http://nginx.org/packages/centos/7/x86_64/
enabled=yes gpgcheck=no state=present'
#yum
[root@m01 ~]# ansible 172.16.1.31  -i hosts   -m  yum  -a
'name=nginx state=installed'

2.编写简单网页测试内容
ansible 172.16.1.31  -i hosts   -m  copy -a
'content="backup.oldoby.com"
dest=/usr/share/nginx/html/index.html '
```

```
3.启动服务不加入开机自启        #systemd/service
[root@m01 ~]# ansible 172.16.1.31  -i hosts   -m  systemd
-a 'name=nginx  state=started enabled=yes'

4.放行对应的端口                 #iptables
[root@m01 ~]# ansible 172.16.1.31  -i hosts   -m  iptables
-a 'table=filter action=append chain=INPUT protocol=tcp
destination_port=80   jump=ACCEPT'
```