

大型架构及配置技术

NSD ARCHITECTURE

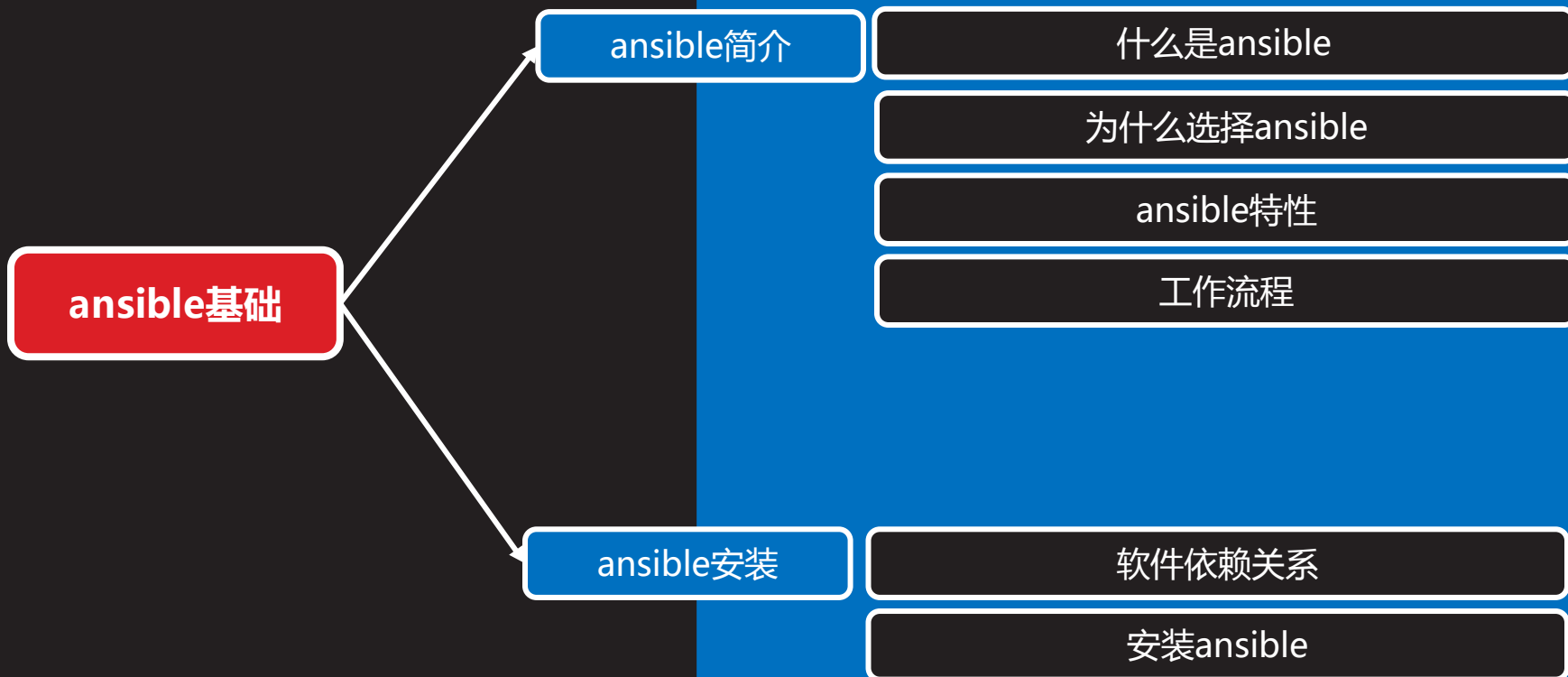
DAY01

内容

上午	09:00 ~ 09:30	ansible基础
	09:30 ~ 10:20	
	10:30 ~ 11:20	
	11:30 ~ 12:00	ad-hoc
下午	14:00 ~ 14:50	
	15:00 ~ 15:50	批量配置管理
	16:10 ~ 17:10	
	17:20 ~ 18:00	总结和答疑



ansible基础



ansible简介

什么是ansible

- ansible是2013年推出的一款IT自动化和DevOps软件，2015年被RedHat收购。是基于Python研发，糅合很多老运维工具的优点，实现了批量操作系统配置，批量程序部署，批量运行命令等功能
- ansible可以实现：
 - 自动化部署APP
 - 自动化管理配置项
 - 自动化持续交付
 - 自动化（AWS）云服务管理



为什么选择ansible

- 选择一款配置管理软件，无外乎从以下几点来权衡利弊
 - 活跃度（社区）
 - 学习成本
 - 使用成本
 - 编码语言
 - 性能
 - 使用是否广泛



为什么选择ansible (续1)

自动化工具	Watch (关注)	Star点赞	Fork (复制)	Contributors (贡献者)
Ansible	1690	24105	8220	2776
SaltStack	576	7835	3643	1832
Puppet	502	4533	1872	446
Chef	424	4907	2038	510



为什么选择ansible（续2）

- ansible优点
 - 只需要SSH和Python即可使用
 - 无客户端
 - ansible功能强大，模块丰富
 - 上手容易，门槛低
 - 基于Python开发，做二次开发更容易
 - 使用公司比较多，社区活跃



ansible特性

- 模块化设计，调用特定的模块完成特定任务
- 基于Python语言实现
 - paramiko
 - PyYAML (半结构化语言)
 - Jinja2
- 其模块支持JSON等标准输出格式，可以采用任何编程语言重写



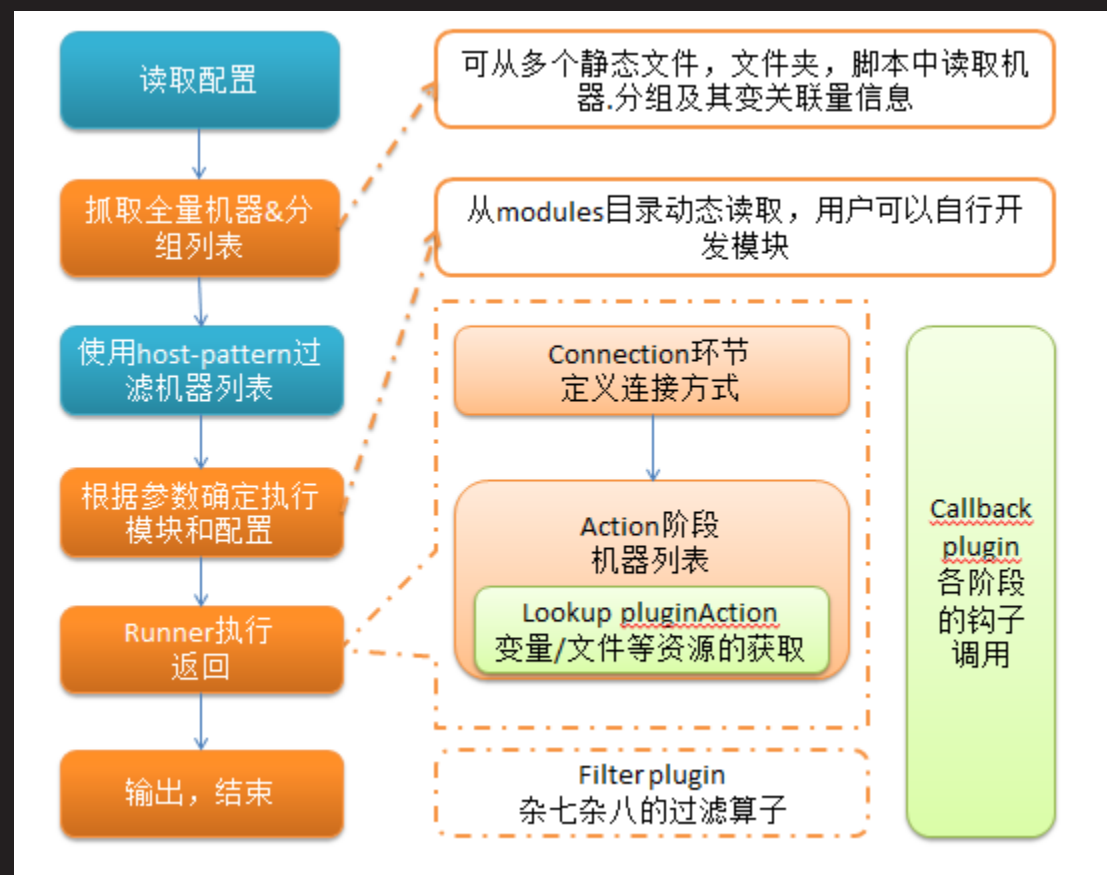
ansible特性（续1）

- 部署简单
- 主从模式工作
- 支持自定义模块
- 支持playbook
- 易于使用
- 支持多层部署
- 支持异构IT环境



工作流程

- ansible大体执行过程



ansible安装

软件依赖关系

- 对管理主机
 - 要求Python 2.6 或Python 2.7
- ansible 使用以下模块，都需要安装
 - paramiko
 - PyYAML
 - Jinja2
 - httplib2
 - six



软件依赖关系（续1）

- 对于被托管主机
 - ansible默认通过SSH协议管理机器
 - 被管理主机要开启ssh服务，允许ansible主机登录
 - 在托管节点上也需要安装Python2.5或以上的版本
 - 如果托管节点上开启了SELinux，需要安装libselinux-python



安装ansible

- ansible可以基于源码运行
- 源码安装
 - pip , 需要配置扩展软件包源extras
 - git

```
yum install epel-release  
yum install git python2-pip
```
 - pip安装依赖模块

```
pip install paramiko PyYAML Jinja2 httplib2 six
```



安装ansible (续1)

- ansible源码下载
 - `git clone git://github.com/ansible/ansible.git`
 - `yum install python-setuptools python-devel`
 - `python setup.py build`
 - `python setup.py install`
- pip方式安装
 - `pip install ansible`



安装ansible (续2)

- yum扩展源安装简单，自动解决依赖关系（推荐）
 - <http://mirror.centos.org/.../.../extras/>
 - yum install ansible
- 安装完成以后验证
 - ansible --version

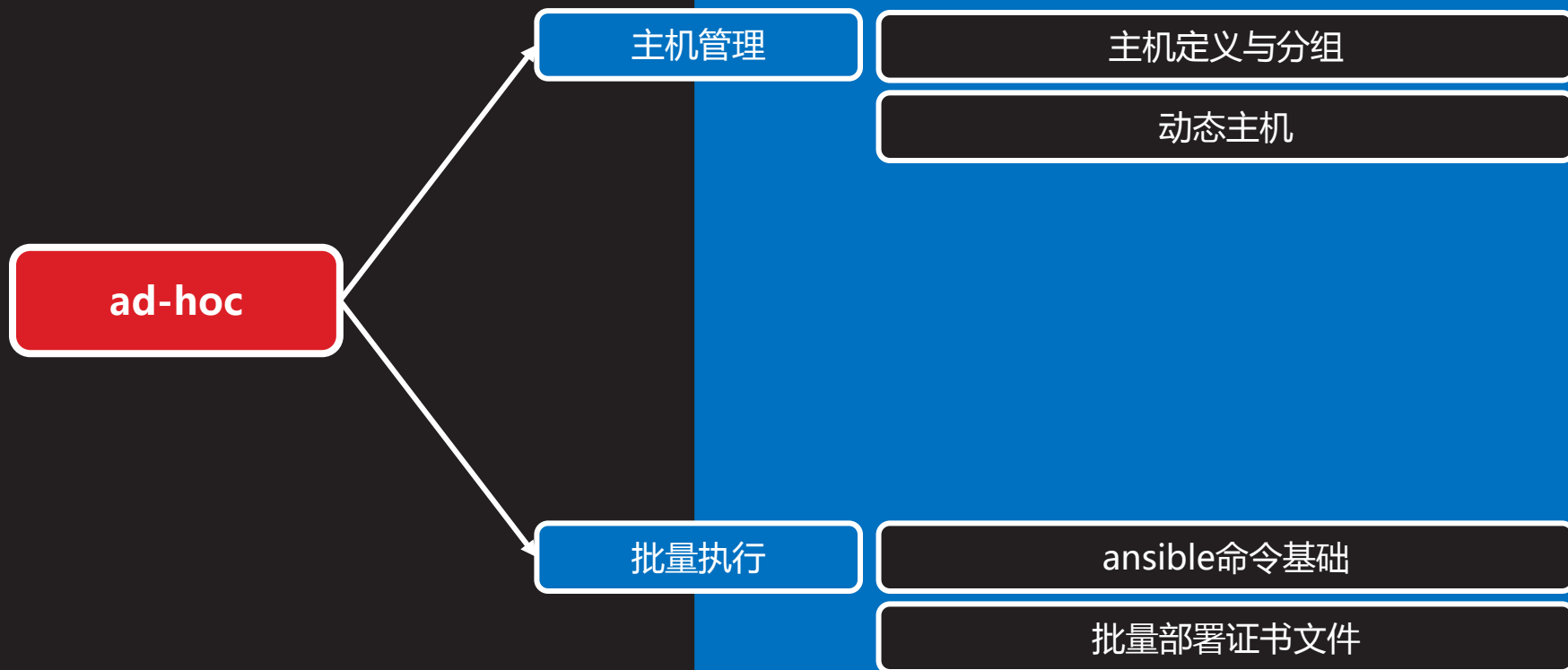


案例1：环境准备

1. 启动6台虚拟机
2. 禁用selinux和firewalld
3. 编辑/etc/hosts
4. 配置yum扩展源并在管理节点安装ansible



ad-hoc



主机管理



主机定义与分组

- 安装ansible之后可以做一些简单的任务
- ansible配置文件查找顺序
 - 首先检测ANSIBLE_CONFIG变量定义的配置文件
 - 其次检查当前目录下的 ./ansible.cfg 文件
 - 再次检查当前用户家目录下 ~/ansible.cfg 文件
 - 最后检查/etc/ansible/ansible.cfg文件
- /etc/ansible/ansible.cfg是ansible的默认配置文件路径



主机定义与分组（续1）

- ansible.cfg 配置文件
 - inventory定义托管主机地址配置文件
 - 先编辑/etc/ansible/hosts文件，写入远程主机的地址。
- 格式
 - # 表示注释
 - [组名称]
 - 主机名称或ip地址，登录用户名，密码、端口等信息
- 测试
 - ansible [组名称] --list-hosts



主机定义与分组（续2）

- inventory 参数说明
 - ansible_ssh_host
 - 将要连接的远程主机名与你想要设定的主机别名不同，可以通过此变量设置
 - ansible_ssh_port
 - ssh端口号：如果不是默认的端口号，通过此变量设置
 - ansible_ssh_user
 - 默认的ssh用户名



主机定义与分组（续3）

- inventory 参数说明
 - ansible_ssh_pass
 - ssh密码（这种方式并不安全,我们强烈建议使用--ask-pass或SSH密钥）
 - ansible_sudo_pass
 - sudo密码（建议使用 --ask-sudo-pass）
 - ansible_sudo_exe（new in version 1.8）
 - sudo命令路径（适用于1.8及以上版本）



主机定义与分组（续4）

- inventory 参数说明
 - ansible_connection
 - 与主机的连接类型，如：local，ssh或paramiko，1.2以前默认使用paramiko，1.2以后默认使用'smart'，它会根据是否支持ControlPersist来判断'ssh'方式是否可行
 - ansible_ssh_private_key_file
 - ssh使用的私钥文件，适用于有多个密钥，而你不想使用SSH代理的情况



主机定义与分组（续5）

- inventory 参数说明
 - ansible_shell_type
 - 目标系统的shell类型，默认情况下，命令的执行使用'sh'语法，可设置为'csh'或'fish'
 - ansible_python_interpreter
 - 目标主机的python路径，适用情况：系统中有多个Python，或者命令路径不是"/usr/bin/python"



主机定义与分组（续6）

- 分组定义、范围定义样例

[web]

web1

web2

[db]

db[1:2]

[cache]

192.168.1.16

[app1:children]

web

db



主机定义与分组（续7）

- 分组定义、范围定义样例

```
[web]
```

```
web[1:2]
```

```
[web:vars]
```

```
ansible_ssh_user="root"
```

```
ansible_ssh_pass="pwd "
```

```
ansible_ssh_port="22"
```

```
[cache]
```

```
c01 ansible_ssh_user="root" ansible_ssh_pass="pwd"
```



主机定义与分组（续8）

- 自定义配置文件
 - 创建文件夹myansible
 - 创建配置文件ansible.cfg

```
[defaults]
inventory = myhost
```
 - 配置主机文件

```
[nginx]
192.168.1.11
192.168.1.12
192.168.1.13
```
 - ansible nginx --list-hosts



案例2：主机定义与分组

1. 熟悉ansible配置文件
2. 定义主机，分组和子组练习
3. 自定义文件，多配置路径练习



动态主机

- 无限可能
 - ansible Inventory包含静态和动态的Inventory，静态Inventory指在文件/etc/ansible/hosts中指定的主机和组，动态Inventory指通过外部脚本获取主机列表，按照其要求格式返回给ansilbe命令
- Json
 - JSON（JavaScript Object Notation，JavaScript对象表示法），一种基于文本独立于语言的轻量级数据交换格式



动态主机（续1）

- 注意事项：
 - 主机部分必须是列表格式
 - Hostdata行，其中的"hosts" 部分可以省略，但使用时，必须是"hosts"



动态主机（续2）

- 脚本输出主机列表

```
#!/usr/bin/python
import json
hostlist = {}
hostlist["bb"] = ["192.168.1.15", "192.168.1.16"]
hostlist["192.168.1.13"] = {
    "ansible_ssh_user":"root","ansible_ssh_pass":"pwd"
}
hostlist["aa"] = {
    "hosts" : ["192.168.1.11", "192.168.1.12"],
    "vars" : {
        "ansible_ssh_user":"root","ansible_ssh_pass":"pwd"
    }
}
print(json.dumps(hostlist))
```



动态主机（续3）

- 脚本输出样例

```
{
  "aa" : {
    "hosts" : ["192.168.1.11", "192.168.1.12"],
    "vars" : {
      "ansible_ssh_user" : "root",
      "ansible_ssh_pass" : "pwd"
    }
  },
  "bb" : ["192.168.1.15", "192.168.1.16"],
  "192.168.1.13": { "ansible_ssh_user" : "root",
    "ansible_ssh_pass" : "pwd"}
}
```



案例3：动态主机

1. 脚本输出主机列表



批量执行



ansible命令基础

- ansible <host-pattern> [options]
 - host-pattern 主机或定义的分组
 - -M 指定模块路径
 - -m 使用模块，默认command模块
 - -a or --args 模块参数
 - -i inventory文件路径，或可执行脚本
 - -k 使用交互式登录密码
 - -e 定义变量
 - -v 详细信息，-vvvv开启debug模式



ansible命令基础（续1）

- 列出要执行的主机
 - `ansible all --list-hosts`
- 批量检测主机
 - `ansible all -m ping`
- 批量执行命令
 - `ansible all -m command -a 'id' -k`



批量部署证书文件

- 每次交互输入密码太麻烦
- 密码写入配置文件安全性很差
- 不同主机不同密码
- 使用key方式认证
- 给所有主机部署公钥

```
ansible all -m authorized_key -a "user=root exclusive=true  
manage_dir=true key='$(< /root/.ssh/authorized_keys)'" -k -v
```



案例4：批量部署证书文件

1. 创建一对密钥
2. 给所有主机部署密钥



批量配置管理

批量配置管理

模块

ansible-doc和ping模块

command模块

shell|raw模块

script模块

copy模块

lineinfile|replace模块

yum模块

service模块

setup模块

模块



ansible-doc和ping模块

- ansible-doc
 - 模块的手册相当与shell的man，很重要
 - `ansible-doc -l` 列出所有模块
 - `ansible-doc modulename` 查看帮助
- ping 模块
 - 测试网络连通性, ping模块没有参数
 - 注：测试ssh的连通性
`ansible host-pattern -m ping`



command模块

- command模块

- 默认模块，远程执行命令

- 用法

- `ansible host-pattern -m command -a '[args]'`

- 查看所有机器负载

- `ansible all -m command -a 'uptime'`

- 查看日期和时间

- `ansible all -m command -a 'date +%F_%T'`



command模块（续1）

- command模块注意事项：
 - 该模块通过-a跟上要执行的命令可以直接执行，若命令里有如下字符则执行不成功
 - "<" , ">" , "|" , "&"
 - 该模块不启动shell直接在ssh进程中执行，所有使用到shell的命令执行都会失败
 - 下列命令执行会失败


```
ansible all -m command -a 'ps aux|grep ssh'
```

```
ansible all -m command -a 'set'
```



shell|raw模块

- shell | raw 模块
 - shell 模块用法基本和command一样，区别是shell模块是通过/bin/sh进行执行命令，可以执行任意命令
 - raw模块，用法和shell模块一样，可以执行任意命令
 - 区别是raw没有chdir、creates、removes参数
 - 执行以下命令查看结果


```
ansible t1 -m command -a 'chdir=/tmp touch f1'
```

```
ansible t1 -m shell -a 'chdir=/tmp touch f2'
```

```
ansible t1 -m raw -a 'chdir=/tmp touch f3'
```



script模块

- script模块
 - 命令太复杂？
 - 在本地写脚本，然后使用script模块批量执行
`ansible t1 -m script -a 'urscript'`
 - 注意：该脚本包含但不限于shell脚本，只要指定Shabang解释器的脚本都可运行



案例5：练习模块

1. 练习使用command , shell , raw, script模块



copy模块

- copy 模块
 - 复制文件到远程主机
 - src：复制远程主机的文件到本地，绝对路径和相对路径都可，路径为目录时会递归复制。若路径以"/"结尾，只复制目录里的内容，若不以"/"结尾，则复制包含目录在内的整个内容，类似于rsync
 - dest：必选项。远程主机的绝对路径，如果源文件是一个目录，那该路径必须是目录



copy模块（续1）

- copy 模块
 - backup：覆盖前先备份原文件，备份文件包含时间信息。有两个选项：yes|no
 - force：若目标主机包含该文件，但内容不同，如果设置为yes，则强制覆盖，设为no，则只有当目标主机的目标位置不存在该文件时才复制。默认为yes
 - 复制文件


```
ansible t1 -m copy -a 'src=/root/alog dest=/root/a.log'
```
 - 复制目录


```
ansible t1 -m copy -a 'src=urdir dest=/root/'
```



lineinfile|replace模块

- lineinfile | replace 模块
 - 类似sed的一种行编辑替换模块
 - path 目的文件
 - regexp 正则表达式
 - line 替换后的结果

```
ansible t1 -m lineinfile -a 'path="/etc/selinux/config"
regexp="^SELINUX=" line="SELINUX=disabled"
```

- 替换指定字符

```
ansible t1 -m replace -a 'path="/etc/selinux/config"
regexp="^(SELINUX=).*" replace="\1disabled"
```



案例6：模块练习

1. 使用copy模块同步数据
2. 使用lineinfile模块编辑文件
3. 使用replace模块修改文件



yum模块

- yum模块
 - 使用yum包管理器来管理软件包
 - config_file : yum的配置文件
 - disable_gpg_check : 关闭gpg_check
 - disablerepo : 不启用某个源
 - enablerepo : 启用某个源
 - name : 要进行操作的软件包名字 , 也可传递一个url 或一个本地的rpm包的路径
 - state : 状态 (present , absent , latest)



yum模块（续1）

- yum模块

- 删除软件包

- ```
ansible t1 -m yum -a 'name="lrzsz" state=absent'
```

- 删除多个软件包

- ```
ansible t1 -m yum -a 'name="lrzsz,lftp" state=absent'
```

- 安装软件包

- ```
ansible t1 -m yum -a 'name="lrzsz"'
```

- 安装多个软件包

- ```
ansible t1 -m yum -a 'name="lrzsz,lftp"'
```



service模块

- service模块
 - name : 必选项 , 服务名称
 - enabled : 是否开机启动 yes|no
 - sleep : 执行restarted , 会在stop和start之间沉睡几秒钟
 - state : 对当前服务执行启动 , 停止、重启、重新加载等操作 (started , stopped , restarted , reloaded)

```
ansible t1 -m service -a 'name="sshd" enabled="yes" state="started"
```



setup模块

- setup模块
 - 主要用于获取主机信息，playbooks里经常会用的另一个参数gather_facts与该模块相关，setup模块下经常用的是filter参数
 - filter过滤所需信息
`ansible t1 -m setup -a 'filter=ansible_distribution'`



案例7：综合练习

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



总结和答疑

总结和答疑

批量部署证书文件

问题现象

故障分析及排除

批量部署证书文件

问题现象

- 部署证书文件时报错

"msg": "Using a SSH password instead of a key is not possible because Host Key checking is enabled and sshpass does not support this. Please add this host's fingerprint to your known_hosts file to manage this host."



故障分析及排除

- 原因分析
 - ansible.cfg文件的host_key_checking字段没有修改或着修改错误
- 解决方案
 - 修改ansible.cfg文件
`host_key_checking = False`

