

大型架构及配置技术

NSD ARCHITECTURE

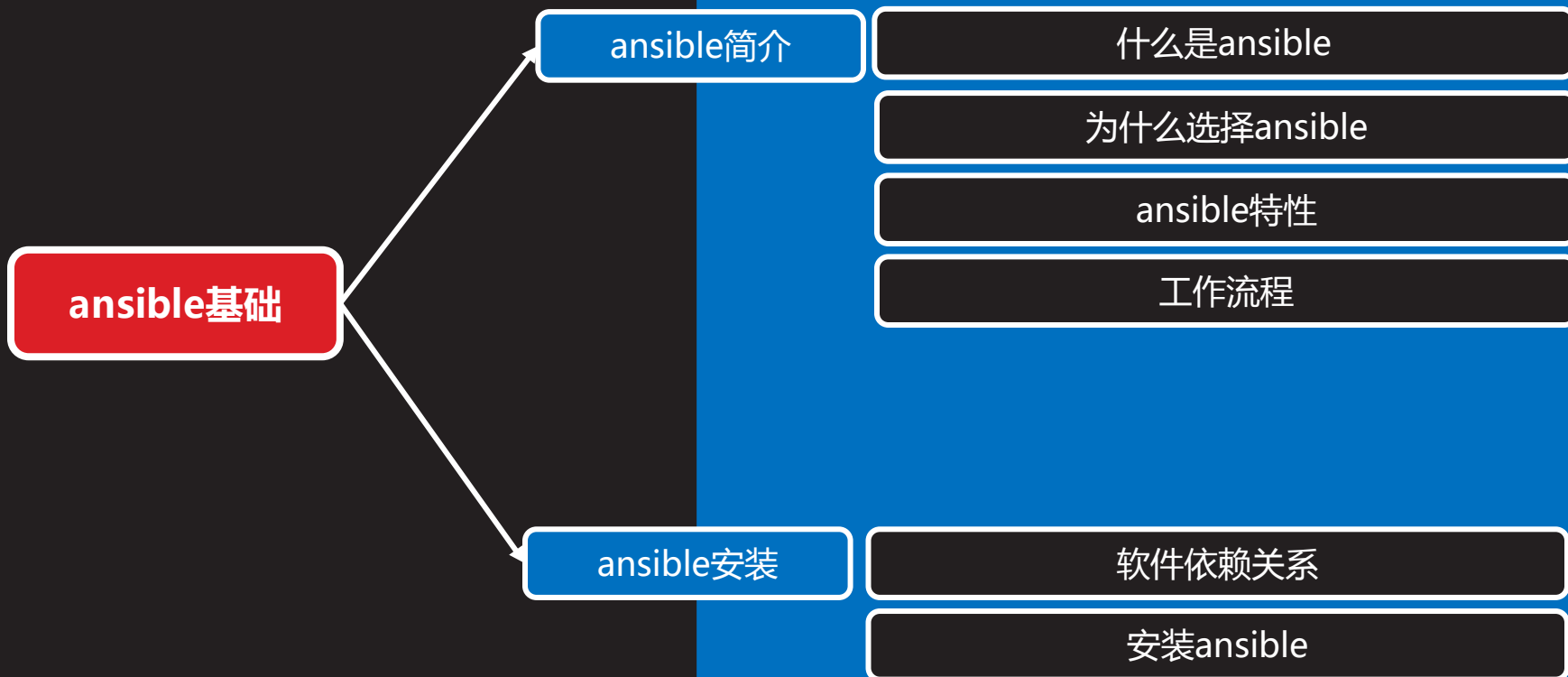
DAY01

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	ansible 基础
	10:30 ~ 11:20	
	11:30 ~ 12:00	ad-hoc
下午	14:00 ~ 15:00	
	15:20 ~ 16:00	批量配置管理
	16:30 ~ 17:30	
	17:30 ~ 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，目前由Redhat已签署Ansible收购协议。其是基于Python研发，糅合了很多老运维工具的优点实现了批量操作系统配置，批量程序的部署，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
 - 自动化的持续交付
 - 自动化的（AWS）云服务管理



为什么要选择ansible

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



为什么要选择ansible

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



为什么要选择ansible

- ansible优点
 - 是仅需要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



为什么要选择ansible

- ansible缺点
 - 对于几千台、上万台机器的操作，还不清楚性能、效率情况如何，需要进一步了解。



ansible特性

- 模块化设计，调用特定的模块来完成特定任务
- 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，可采用任何编程语言重写



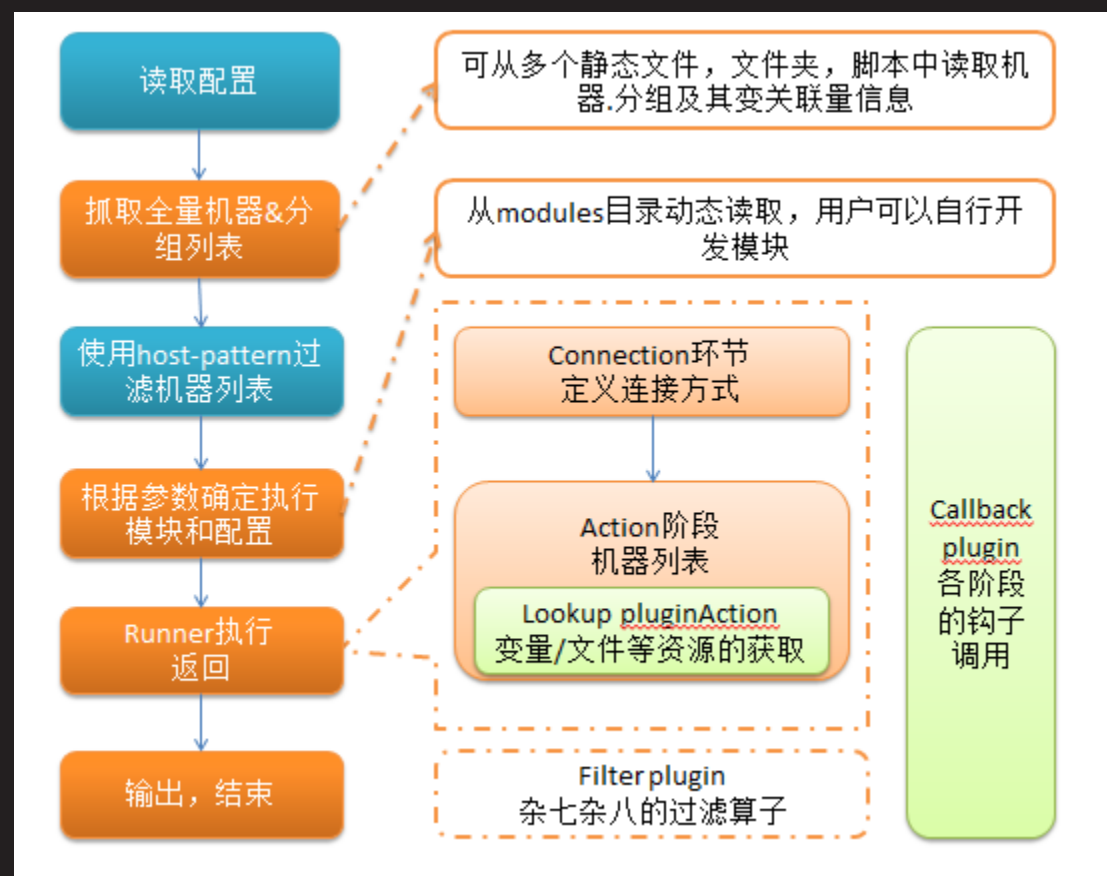
ansible特性

- 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 使用了以下模块，都需要安装
 - paramiko
 - PyYAML
 - Jinja2
 - httplib2
 - six



软件依赖关系

- 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 Python 2.5 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

- 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

- 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

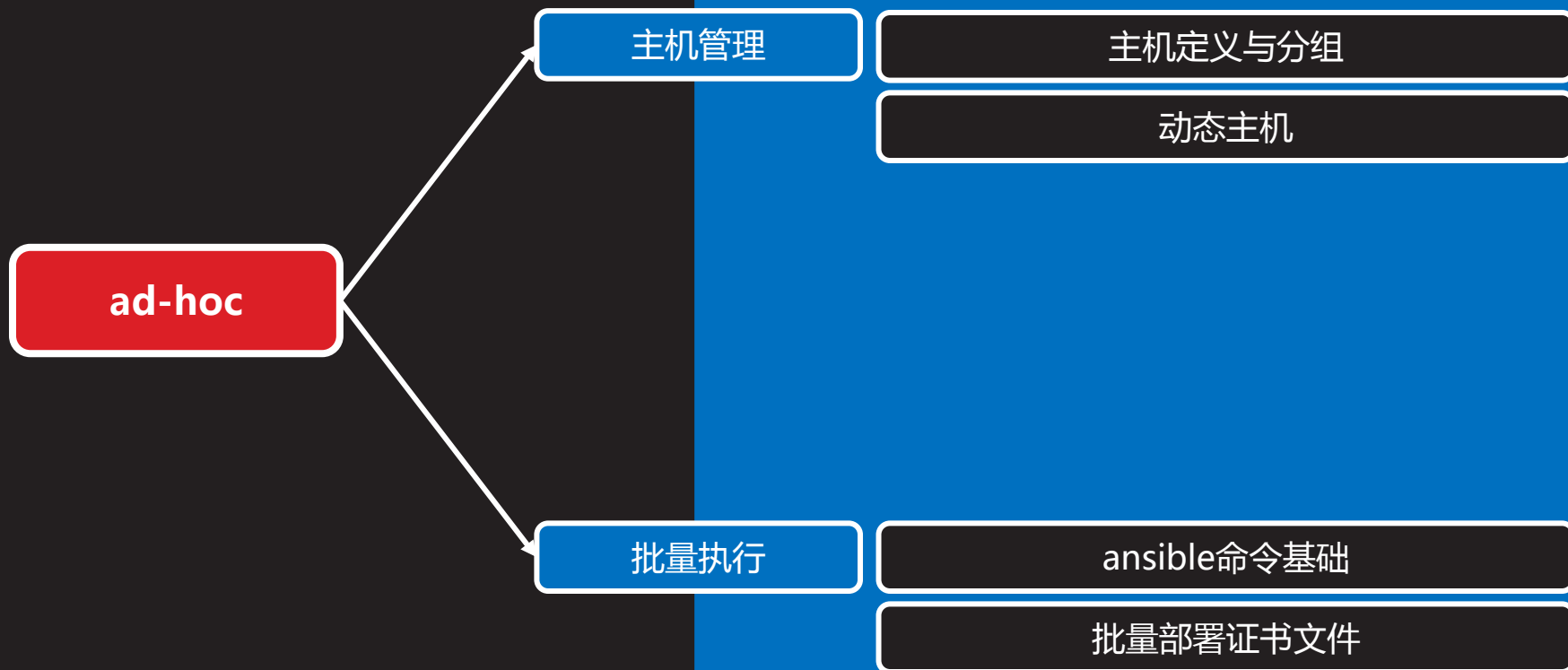


课堂练习

- 1、启动 6 台虚拟机
- 2、禁用 selinux 和 firewalld
- 3、给主机命名，编辑 /etc/hosts
- 4、1台作为管理节点，其他为托管主机
- 5、配置本地 yum 扩展源
- 6、在管理节点安装 ansible



ad-hoc



主机管理



主机定义与分组

- 安装好了 Ansible 之后就可以开始一些简单的任务了
- 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 默认配置文件路径



主机定义与分组

- ansible.cfg 配置文件
 - inventory 是定义托管主机地址配置文件
 - 首先编辑 /etc/ansible/hosts 文件，写入一些远程主机的地址。
- 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

- 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的别名不同的话,可通过此变量设置.
 - 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

- 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

- inventory 参数说明
 - ansible_connection
 - 与主机的连接类型.比如:local, ssh 或者 paramiko.
Ansible 1.2 以前默认使用 paramiko.1.2 以后默认使用 'smart','smart' 方式会根据是否支持 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

- 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.适用于的情况: 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

- 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

- 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

- 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



课堂练习

- 1、熟悉 ansible 配置文件
- 2、定义主机练习
- 3、定义分组练习
- 4、定义子组练习
- 5、自定义文件，多配置路径练习



动态主机

- 无限可能
 - Ansible Inventory实际上是包含静态Inventory和动态Inventory两部分，静态Inventory指的是在文件/etc/ansible/hosts中指定的主机和组，Dynamic Inventory指通过外部脚本获取主机列表，并按照ansible所要求的格式返回给ansible命令的。
- json
 - JSON的全称是“JavaScript Object Notation”，意思是JavaScript对象表示法，它是一种基于文本，独立于语言的轻量级数据交换格式。



动态主机

- 注意事项：
 - 1、主机部分必须是列表格式的；
 - 2、hostdata行，其中的"hosts" 部分可以省略，但如果使用时，必须是"hosts"



动态主机

- 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

- 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"}
}
```



课堂练习

- 脚本演示讲解
- shell 脚本
- python 脚本



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

- 列出要执行的主机，不执行任何操作
 - `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

- 每次交互输入密码比较麻烦
- 密码写入配置文件安全性很差
- 不同主机不同密码，配置文件要上天
- 使用 key 方式认证，是一个不错的选择
- 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`



批量部署证书文件

- 1、创建一对密钥
- 2、给所有主机部署



批量部署证书文件

- 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."
 - 解决方法:
 - 修改 ansible.cfg


```
host_key_checking = False
```



批量配置管理

模块

ansible-doc

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

批量配置管理

模块



模块

- ansible-doc
 - 模块的手册，相当与 shell 的 man
 - 非常重要，非常重要，非常重要
 - 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



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'`



模块

- 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，不过命令里如果有带有如下字符部分则执行不成功
 - "<", ">", "|", "&"
 - 该模块不启动 shell 直接在 ssh 进程中执行，所有使用到 shell 特性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'
```



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'
```



模块

- script模块
 - 复杂命令怎么办？
 - ansible 要上天
 - 直接在本地写脚本，然后使用 script 模块批量执行
 - `ansible t1 -m script -a 'urscript'`
 - 友情提示：该脚本包含但不限于 shell 脚本，只要指定 Sha-bang 解释器的脚本都可运行



课堂练习

- 练习使用
- command , shell , raw, script 模块
- 掌握他们的特性与区别



模块

- copy 模块
 - 复制文件到远程主机
 - src：要复制到远程主机的文件在本地的地址，可以是绝对路径，也可以是相对路径。如果路径是一个目录，它将递归复制。在这种情况下，如果路径使用"/"来结尾，则只复制目录里的内容，如果没有使用"/"来结尾，则包含目录在内的整个内容全部复制，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，那么该路径也必须是个目录



模块

- copy 模块

- backup : 在覆盖之前将原文件备份，备份文件包含时间信息。有两个选项：yes|no
- force : 如果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，如果为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/'`



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"
```



课堂练习

- 练习
- 使用 copy 模块同步数据
- 使用 lineinfile 编辑文件
- 使用 replace 修改文件



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的名字, 也可以传递一个url或者一个本地的rpm包的路径
 - state : 状态 (present , absent , latest)



模块

- 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"'
```



模块

- service模块
 - name : 必选项 , 服务名称
 - enabled : 是否开机启动 yes|no
 - sleep : 如果执行了restarted , 在则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"
```



模块

- setup模块
 - 主要用于获取主机信息，在playbooks里经常会用到的一个参数gather_facts就与该模块相关。setup模块下经常使用的一个参数是filter参数
 - filter 可以过滤到我们需要的信息

```
ansible t1 -m setup -a 'filter=ansible_distribution'
```



课堂练习

- 综合练习
 - 安装 apache
 - 修改 apache 监听的端口为 8080
 - 为 apache 增加 NameServer 配置
 - 设置默认主页
 - 启动服务
 - 设置开机自启动



总结和答疑

总结和答疑

批量执行失败

问题现象

故障分析及排除