

# 大型架构及配置技术

**NSD ARCHITECTURE**

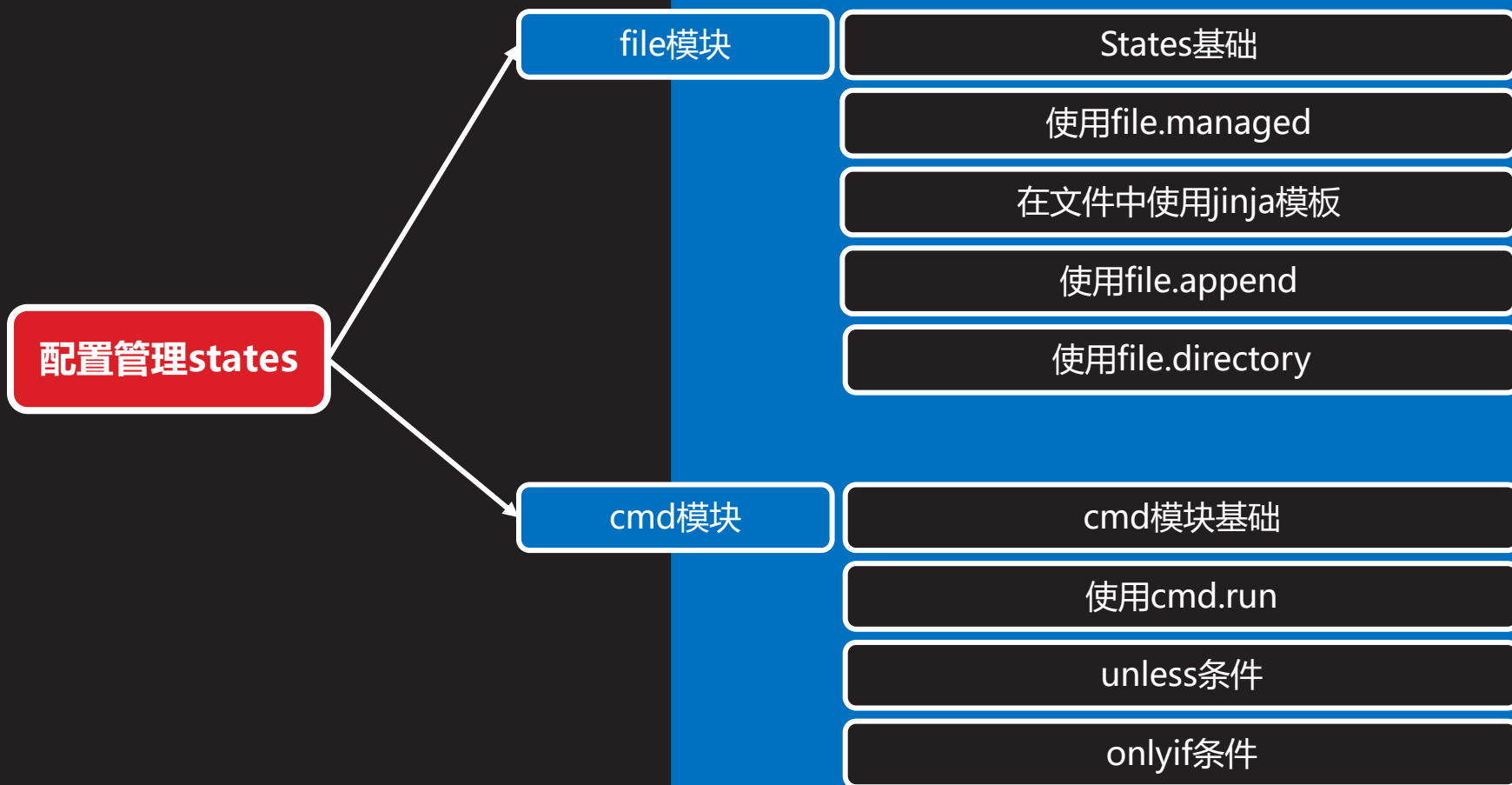
**DAY04**

# 内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	配置管理states
	10:30 ~ 11:20	
	11:30 ~ 12:20	系统初始化
下午	14:00 ~ 14:50	服务器批量部署管理
	15:00 ~ 15:50	
	16:10 ~ 17:00	源码包管理
	17:10 ~ 18:00	总结和答疑



# 配置管理states



# file模块

---

# States基础

- States是satlstack中的配置语言
- 安装软件包、管理配置文件都需要编写一些states sls文件

- states sls使用YAML语法
- 查看所有states列表

```
[root@vh01 pillar]# salt 'vh01.tedu.cn' sys.list_state_modules
```

- 查看states模块功能

```
[root@vh01 pillar]# salt 'vh01.tedu.cn' sys.list_state_functions file
```



# 使用file.managed

- 为不同的环境设置不同的文件目录

```
[root@vh01 salt]# vim /etc/salt/master
```

```
file_roots:
```

```
base:
```

```
- /srv/salt/base
```

```
dev:
```

```
- /srv/salt/dev
```

```
prod:
```

```
- /srv/salt/prod
```

```
[root@vh01 salt]# systemctl restart salt-master.service
```

```
[root@vh01 salt]# mkdir /srv/salt/{base,dev,prod}
```



# 使用file.managed ( 续1 )

- 准备入口文件

```
[root@vh01 salt]# cd /srv/salt/base/
```

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

- 配置分发文件说明

```
[root@vh01 base]# vim dns_file.sls
```

```
resolv_conf:
```

```
  file.managed:
```

```
    - name: /etc/resolv.conf
```

```
    - source: salt://files/resolv.conf
```

```
    - user: root
```

```
    - group: root
```

```
    - mode: 644
```



# 使用file.managed ( 续2 )

- 执行文件分发操作

```
[root@vh01 base]# mkdir files
```

```
[root@vh01 base]# vim files/resolv.conf
```

```
nameserver 192.168.113.2
```

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
```

```
[root@vh01 base]# salt '*' state.highstate saltenv=base
```





# 在文件中使用jinja模板

- 配置分发文件说明

```
[root@vh01 base]# vim dns_file.sls
```

```
resolv_conf:
```

```
  file.managed:
```

- name: /etc/resolv.conf
- source: salt://files/resolv.conf
- user: root
- group: root
- mode: 644
- template: jinja
- defaults:  
 DNS\_IP: 192.168.113.254



# 在文件中使用jinja模板（续1）

- 执行文件分发操作

```
[root@vh01 base]# vim files/resolv.conf
```

```
nameserver {{ DNS_IP }}
```

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
```

```
[root@vh01 base]# salt '*' state.highstate saltenv=base
```



# 使用file.append

- 用于向指定文件中追加文本

```
[root@vh01 base]# vim issue.sls
```

```
issue_file:
```

```
  file.append:
```

- name: /etc/issue
- text:
  - Red Hat Enterprise Linux 7
  - Tedu Ltd Server



# 使用file.append ( 续1 )

- 执行文件分发操作

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

```
    - issue
```

```
[root@vh01 base]# salt '*' state.highstate env=base test
```

```
[root@vh01 base]# salt '*' state.highstate env=base
```



# 使用file.directory

- 用于创建或管理目录

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

```
    - issue
```

```
  'vh01.tedu.cn':
```

```
    - rpm_pkgs
```



# 使用file.directory ( 续1 )

- 创建声明sls文件

```
[root@vh01 base]# cat rpm_pkgs.sls
```

```
rpms:
```

```
  file.directory:
```

- name: /opt/rpm\_pkgs
- user: root
- group: root
- dir\_mode: 755
- file\_mode: 644

- 执行文件分发操作

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
```

```
[root@vh01 base]# salt '*' state.highstate saltenv=base
```



# 案例1：文件分发管理

1. 配置两台minion使用相同的dns配置
2. Dns的ip地址通过jinja模板指定
3. 向第一台minion的/etc/issue文件中追加文本
  - Hello from tedu.cn
4. 第二台minion要求创建/opt/rpm\_pkgs目录，属主、属组为root，目录权限为755，文件权限为644



# cmd模块

---



# cmd模块基础

- cmd模块强制 minion 执行命令
- 通常用于要求服务器满足特定的条件
- 可以限制当某个条件满足时才执行命令
- 也可以限制当某个条件不满足时才执行命令



# 使用cmd.run

- 与文件管理类似，首先定义入口文件

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

```
    - issue
```

```
  'vh01.tedu.cn':
```

```
    - rpm_pkgs
```

```
    - exec_date
```



# 使用cmd.run ( 续1 )

- 定义待执行命令的sls文件

```
[root@vh01 base]# vim exec_date.sls
date > /tmp/salt-run:
  cmd.run
```

- 执行命令

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
[root@vh01 base]# salt '*' state.highstate saltenv=base
```



# unless条件

- 当unless条件不满足时，需要执行令

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

```
    - issue
```

```
  'vh01.tedu.cn':
```

```
    - rpm_pkgs
```

```
    - exec_date
```

```
    - add_user
```



# unless条件（续1）

- 定义待执行命令的sls文件

```
[root@vh01 base]# vim add_user.sls
useradd bob:
  cmd.run:
    - unless: id bob
```

- 执行命令

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
[root@vh01 base]# salt '*' state.highstate saltenv=base
```



# onlyif条件

- 当onlyif条件满足时，需要执行令

```
[root@vh01 base]# vim top.sls
```

```
base:
```

```
  'vh02.tedu.cn':
```

```
    - dns_file
```

```
    - issue
```

```
  'vh01.tedu.cn':
```

```
    - rpm_pkgs
```

```
    - exec_date
```

```
    - del_user
```



# onlyif条件（续1）

- 当onlyif条件满足时，需要执行命令

```
[root@vh01 base]# vim del_user.sls
userdel bob:
  cmd.run:
    - onlyif: id bob
```

- 执行命令

```
[root@vh01 base]# salt '*' state.highstate saltenv=base test
[root@vh01 base]# salt '*' state.highstate saltenv=base
```



# 系统初始化

---

系统初始化

系统初始化

系统初始化概述

配置DNS

History记录时间

内核参数优化

创建用户



# 系统初始化



# 系统初始化概述

- 服务器上架并安装好操作系统后，都会有一些基础的配置操作
- 建议将所有服务器都会涉及的基础配置或者软件部署都归类放在base环境下
- 可以在base环境下创建一个init目录，将系统初始化配置的sls文件均放到init目录下，称其为“初始化模块”



# 配置DNS

- 生产环境下，一个重要的配置就是DNS解析设置
- 强烈建议在内网建立自己的内网DNS服务器

```
[root@vh01 ~]# vim /srv/salt/base/init/dns.sls
```

```
resolv_file:
```

```
  file.managed:
```

- source: salt://init/files/resolv.conf
- user: root
- group: root
- mode: 644



# History记录时间

- 生产环境非常简单实用的功能就是让history记录时间，这样可以清晰地知道什么用户在什么时间执行了什么命令

```
[root@vh01 ~]# vim /srv/salt/base/init/history.sls
```

```
profile_file:
```

```
  file.append:
```

```
    - name: /etc/profile
```

```
    - text:
```

```
      - export HISTTIMEFORMAT="%F %T $(whoami) "
```



# 内核参数优化

- 在进行系统初始化的过程中，需要对默认的内核参数进行调优，saltstack提供了sysctl状态模块用来进行内核参数的配置

```
[root@vh01 ~]# vim /srv/salt/base/init/sysctl.sls
net.ipv4.ip_forward:
  sysctl.present:
    - value: 1
```



# 创建用户

- 安装好的系统，有时需要统一创建用户，并且设置初始密码

```
[root@vh01 ~]# vim /srv/salt/base/init/add_user.sls
```

```
useradd bob:
```

```
cmd.run:
```

```
- unless: id bob
```

```
echo 123456 | passwd --stdin bob:
```

```
cmd.run:
```

```
- onlyif: id bob
```

```
chage -d0 bob:
```

```
cmd.run:
```

```
- onlyif: id bob
```



## 案例2：系统初始化

1. 配置minion的dns服务器地址为192.168.4.1
2. 配置历史命令显示时间和用户
3. 配置系统能够实现ip转发
4. 配置系统创建用户bob，初始密码为123456，同时要求用户初次登陆必须修改密码



# 服务器批量部署管理

## 服务器批量部署管理

### 软件包管理

pkg模块

使用pkg.installed

require条件

使用jinja模板

### 服务管理

service模块

控制服务状态

使用watch



# 软件包管理

---

# pkg模块

- pkg模块可以实现软件包管理
- 管理的软件包包括红帽RPM包和Ubuntu的deb包等
- 主要的方法有：
  - pkg.installed : 安装软件包
  - pkg.latest : 保持软件包为最新版本
  - pkg.remove : 卸载软件包
  - pkg.purge : 下载软件包，删除配置文件



# 使用pkg.installed

- 创建入口文件

```
[root@vh01 base]# cd /srv/salt/prod/
```

```
[root@vh01 prod]# vim top.sls
```

```
prod:
```

```
  'vh02.tedu.cn':
```

```
    - httpd
```

- 创建软件包安装sls声明文件

```
[root@vh01 prod]# vim httpd.sls
```

```
httpd_pkg_installed:
```

```
  pkg.installed:
```

```
    - name: httpd
```



# 使用pkg.installed ( 续1 )

- 安装软件包

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate saltenv=prod  
test
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate saltenv=prod
```

- 验证

```
[root@vh02 ~]# rpm -q httpd  
httpd-2.4.6-40.el7.centos.x86_64
```



# require条件

- 只有httpd安装了才分发配置文件

```
[root@vh01 prod]# vim httpd.sls
```

```
httpd_pkg_installed:
```

```
  pkg.installed:
```

```
    - name: httpd
```

```
httpd_conf:
```

```
  file.managed:
```

```
    - name: /etc/httpd/conf/httpd.conf
```

```
    - source: salt://files/httpd.conf
```

```
    - require:
```

```
      - pkg: httpd_pkg_installed
```



## require条件（续1）

- 分发配置文件并验证

```
[root@vh02 ~]# rm -f /etc/httpd/conf/httpd.conf
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate env=prod  
test
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate env=prod
```

```
[root@vh02 ~]# ls /etc/httpd/conf/httpd.conf  
/etc/httpd/conf/httpd.conf
```



# 使用jinja模板

- 修改配置文件，通过变量定义端口号

```
[root@vh01 prod]# vim files/httpd.conf
```

```
Listen {{ PORT }}
```

- 修改创建软件包安装sls声明文件

```
httpd_pkg_installed:
```

```
  pkg.installed:
```

```
    - name: httpd
```

```
httpd_conf:
```

```
  file.managed:
```

```
    - name: /etc/httpd/conf/httpd.conf
```

```
    - source: salt://files/httpd.conf
```

```
    - template: jinja
```

```
    - defaults:
```

```
      PORT: 8080
```

```
    - require:
```

```
      - pkg: httpd_pkg_installed
```



# 使用jinja模板（续1）

- 分发配置文件并验证

```
[root@vh02 ~]# rm -f /etc/httpd/conf/httpd.conf
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate env=prod  
test
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate env=prod
```

```
[root@vh02 ~]# grep '^Listen' /etc/httpd/conf/httpd.conf  
Listen 8080
```





## 案例3：部署httpd web服务器

- 通过saltstack部署httpd web服务器
  1. 自动安装httpd软件包
  2. 将master的配置文件分发到客户端
  3. 服务器监听端口用jinja变量实现



# 服务管理



# service模块

- 软件部署完毕后，需要确保服务处于运行状态，并且能够实现开机自启，这就用到了service模块
  - service.running：确保服务处于运行状态
  - service.enabled：开机自启
  - service.disabled：开机不启动
  - service.dead：确保服务处于未运行状态



# 控制服务状态

- 通过sls文件保证httpd服务处于运行状态，并实现开机自启

```
[root@vh01 prod]# vim httpd.sls
```

原有数据保留，追加以下内容：

```
httpd_service_running:
```

```
service.running:
```

```
- name: httpd
```

```
- enable: true
```

```
- restart: true
```



# 控制服务状态（续1）

- 验证

```
[root@vh02 ~]# systemctl is-enabled httpd
```

```
[root@vh02 ~]# systemctl status httpd
```

```
[root@vh01 prod]# salt 'vh02.tedu.cn' state.highstate env=prod
```

```
[root@vh02 ~]# systemctl is-enabled httpd
```

```
[root@vh02 ~]# systemctl status httpd
```



# 使用watch

- 服务如果能够正常启动，需要确保存在配置文件，设置如果配置文件存在，才启动服务

```
[root@vh01 prod]# vim httpd.sls
```

原有数据保留，追加以下内容：

```
httpd_service_running:
```

```
  service.running:
```

```
    - name: httpd
```

```
    - enable: true
```

```
    - restart: true
```

```
    - watch:
```

```
      - file: httpd_conf
```



## 案例4：设置httpd运行状态

- 配置案例3中的httpd服务能够开机自启
- 配置httpd服务务必处于运行状态
- 检查httpd配置文件，当配置文件存在时，服务才处于运行状态



# 源码包管理

---

源码包管理

部署源码包

源码包概述

规划目录结构

定义软件包依赖关系

设定源码编译方式

初始化服务运行环境

设置服务工作运行状态

部署源码包



# 部署源码包

---

# 规划目录结构

- 因为涉及到的目录较多，因此先规划好目录结构

```
[root@vh01 dev]# tree /srv/salt/dev/
```

```
/srv/salt/dev/
```

```
├── initpkg
│   └── pkg_install.sls
├── nginx
│   ├── files
│   │   ├── nginx-1.9.12.tar.gz
│   │   └── nginx.service
│   └── install_nginx.sls
└── top.sls
```



# 定义软件包依赖关系

- 源码编译需要用到编译器等工具

```
[root@vh01 dev]# vim initpkg/pkg_install.sls
```

```
init_pkg_install:
```

```
  pkg.installed:
```

```
    - names:
```

```
      - gcc
```

```
      - gcc-c++
```

```
      - make
```

```
      - autoconf
```

```
      - openssl-devel
```

```
      - pcre-devel
```



# 设定源码编译方式

- 首先通过file.managed模块将源码分发至服务器

```
[root@vh01 dev]# vim nginx/install_nginx.sls
```

```
include:
```

```
- initpkg.pkg_install
```

```
nginx_src_install:
```

```
file.managed:
```

```
- name: /usr/local/src/nginx-1.9.12.tar.gz
```

```
- source: salt://nginx/files/nginx-1.9.12.tar.gz
```

```
- user: root
```

```
- group: root
```

```
- mode: 644
```



# 设定源码编译方式（续1）

- 通过cmd.run进行源码编译

```
[root@vh01 dev]# vim nginx/install_nginx.sls
```

继续上页配置，在nginx\_src\_install下添加

cmd.run:

- name: useradd -s /sbin/nologin nginx && cd /usr/local/src && tar xzf nginx-1.9.12.tar.gz && cd nginx-1.9.12 && ./configure --prefix=/usr/local/nginx --user=nginx --group=nginx && make && make install
- unless: test -d /usr/local/nginx
- require:
  - file: nginx\_src\_install
  - pkg: init\_pkg\_install



# 初始化服务运行环境

- 拷贝服务单元文件

```
[root@vh01 dev]# vim nginx/install_nginx.sls
```

继续上页配置

nginx\_init:

file.managed:

- name: /usr/lib/systemd/system/nginx.service
- source: salt://nginx/files/nginx.service
- user: root
- group: root
- mode: 644

cmd.run:

- name: systemctl daemon-reload



# 设置服务工作运行状态

- 设定服务运行状态

```
[root@vh01 dev]# vim nginx/install_nginx.sls
```

继续上页配置

```
nginx_service:
```

```
  service.running:
```

```
    - name: nginx
```

```
    - enable: true
```

```
    - restart: true
```



# 部署源码包

- 编辑入口文件，并部署nginx服务

```
[root@vh01 dev]# vim top.sls
```

```
dev:
```

```
  'vh02.tedu.cn':
```

```
    - initpkg.pkg_install
```

```
    - nginx.install_nginx
```

```
[root@vh01 dev]# salt 'vh02.tedu.cn' state.highstate saltenv=dev
```





## 案例5：通过源码部署nginx

1. 通过saltstack在目录主机上部署nginx
2. 要求无论是软件包安装，还是文件拷贝，以及服务状态设置都必须在master上完成
3. 所有的服务器部署完毕后，nginx必须处于运行状态，并设置了开机自启



# 总结和答疑

---

总结和答疑

分发文件不成功

问题现象

故障分析及排除

# 分发文件不成功

---

# 问题现象

- 当执行以下语句，进行分发文件时，minion端并没有得到文件

```
# salt -N group1 cp.get_file /etc/hosts /tmp/zhuji
```



# 故障分析及排除

- 原因分析
  - cp.get\_file分发文件时，不能随意指定路径
- 解决办法
  - 将文件拷贝到/srv/salt/files/目录中
  - 通过以下命令进行分发

```
# salt -N group1 cp.get_file salt://files/hosts /tmp/zhuji
```

