**SaltStack实践案例**

**1 案例简述**

通过SaltStack的配置管理实现“中小型Web架构”的自动化部署和配置管理，主要包括以下功能和服务：

系统初始化

Haproxy服务

Keepalived服务

Nginx服务

PHP（FastCGI）服务

Memcached服务

案例架构图如图1-1所示：

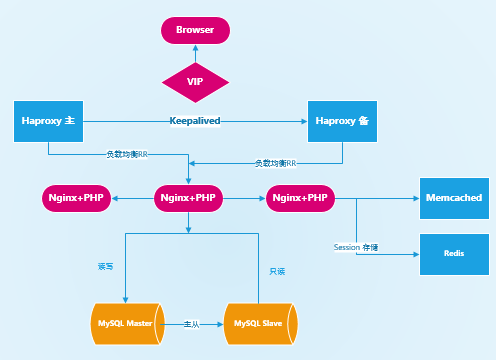


图 1-1 实践案例架构图

案例思路：按照系统初始化、功能模块、业务模块依次进行设计与实现：

系统初始化： 操作系统安装完成之后，通常进行的一些初始设置，比如：安装监控代理、调整内核参数、设置域名解析、安装常用工具等

功能模块： 生产环境使用到的应用，例如、Nginx、PHP、Haproxy、Keepalived等此类应用的安装和管理。

业务模块： 功能模块已经编写了大量基础的功能状态，在业务层面进行引用，因此功能模块需要尽可能的全，且独立。不同的业务类型可以在Include功能模块里面安装和部署。每个业务使用自己的配置文件，最后我们只需要在top.sls里面指定Minion端的某一业务状态即可。

参考中文文档：http://docs.saltstack.cn/zh\_CN/latest/topics/tutorials/starting\_states.html

参考英文文档：https://docs.saltstack.com/en/latest/

**2 环境规划**

环境规划包括两种：

实践案例的网络配置及服务器环境

SaltStack中file\_roots和Pillar\_roots定义的SaltStack环境

**2.1.1 实现环境设置**

需要至少两台以上的虚拟机或者物理机，本教程的实验环境如表2-1所示。

**表2-1 案例实验环境**

|  |  |  |  |
| --- | --- | --- | --- |
| **Hostname** | **IP** | **Roles** | **os** |
| **saltstack-master** | **10.1.1.97** | **master、minion、Haproxy+Keepalived、Nginx+PHP** | **CentOS 6.8 mininal** |
| **saltstack-minion** | **10.1.1.98** | **Minion、Memcached、Haproxy+Keepalived、Nginx+PHP** | **CentOS 6.8 mininal** |

**2.1.2 SaltStack环境设置**

SaltStack环境设置

使用两个环境base和prod，base环境用来存放初始化的功能，prod环境用于放置生产的配置管理功能：

编辑master配置文件

[root@saltstack-master ~]# egrep -v "^#|^$" /etc/salt/master

#Filse Server settings

file\_roots:

base:

- /srv/salt/base

prod:

- /srv/salt/prod

#Pillar settings

pillar\_roots:

base:

- /srv/pillar/base

prod:

- /srv/pillar/prod

参照上面配置对Master配置文件进行修改，建议修改一致。

创建目录结构，默认目录不存在，然后重启saltstack-master：

[root@saltstack-master ~]# mkdir -p /srv/salt/base /srv/salt/prod

[root@saltstack-master ~]# mkdir -p /srv/pillar/base /srv/pillar/prod

[root@saltstack-master ~]# /etc/init.d/saltstack-master restart

Stopping saltstack-master daemon: [ OK ]

Starting saltstack-master daemon: [ OK ]

**2.2 YAML编写技巧**

YAML是YAML Ain's Markup Language的首字符编写，和GUN一样，YAML是一个递归着说‘不’的名字，不对的是YAML说不的对象是XML，YAML语法中，结构通过空格展示，项目用“-”代表。键值对使用“:”分割。

YAML语法规则：

缩进

YAML使用一个固定的缩进风格表示数据层级结构关系。

冒号

Python的字典是简单的键值对，（叫哈希表或关联数组）。

my\_key: my\_value

python中，上面的命令映射为：

{‘my\_key’: ‘my\_value’}

或：

my\_key：

my\_value

字典可以嵌套：

first\_level\_dict\_key:

second\_level\_dict\_key: value\_in\_second\_level\_dict

Python中上面的命令改写成：

{

‘first\_level\_dict\_key:{

‘second\_level\_dict\_key’: ‘value\_in\_second\_level\_dict’

}

}

短横杠

想要表示列表项，使用一个短横杠加一个空格，多个项使用同样的缩进级别作为同一列表的一部分：

- list\_value\_one

- list\_value\_two

- list\_value\_three

列表表示一个键值对的value，例如：一次性安装多个软件包：

my\_dictionary:

- list\_value\_one

- list\_value\_two

- list\_value\_three

Python,上面的命令用python改写：

{‘my\_dictionary’: [‘list\_value\_one’,’’list\_value\_two,’list\_value\_three]}

Jinja使用技巧

Jinja是基于Python的模板引擎，功能类似于PHP的Smarty，J2EE的Freemarker。Salt默认使用yaml\_jinja渲染器。yaml\_jinja的流程是先用jinja2模板引擎处理SLS，然后在调用YANL解析器。

沙箱执行模式，模板的每个部分都在引擎的监督之下执行，模板将会被明确地标记在白名单或黑名单内，这样对于那些不信任的模板也可以执行。

强大的自动HTML转义系统，可以有效地阻止跨站脚本攻击。

模板继承机制，此机制可以使得所有的模板都具有相似一致的布局，也方便了开发人员对模板的修改和管理。

高效的执行效率，Jinja2引擎在模板第一次加载时就把源码转换成Python字节码，加快模板执行时间。

可选的预编译模式。

调试系统融合了标准的Python的TrackBack系统，使得模板编译和运行期间的错误能及时被发现和调试。

语法可配置，可以重新配置Jinja2使得它更好地适应LaTeX或JavaScript的输出。

模板设计人员帮助手册，此手册指导设计人员更好地使用Jinja2引擎的各种方法。[1]

Jinja基本使用

1、File状态使用template参数 - template：jinja

2、模板文件里面变量使用{{name}}，例如：{{PORT}}

3、File状态模板要指定变量列表：

- defaults

PORT: 8080

Jinja变量使用Grains：

{{ grains[‘fqdn\_ip4’] }}

Jinja变量使用执行模块：

{{ salt[‘network.hw\_addr’](‘eth0’) }}

Jinja变量使用pillar：

{{ pilllar[‘apache’][‘PORT’] }}

Jinja逻辑关系：

Jinja主要可以用来给状态增加逻辑关系，当系统环境同时存在CentOS和Ubuntu，Apache软件包的名字是不同的，通过Jinja的逻辑语法指定（使用Grains来判断服务器的操作系统）

{% if grains[‘os’] == ‘Redhat’ %}

apache: httpd

{% elif grains[‘os’] == ‘Debian’ %}

apache: apache2

{% endif %}

**2.3 系统初始化**

通常服务器安装完操作系统之后，都会进行一些基础的设置，生产环境使用SaltStack时，建议将所有的服务器都会进行的基础配置或者软件部署归类放在Base环境下面，本教程中在Base环境下创建一个Init的目录，将系统初始化配置的SLS均放置到Init目录下，可以叫作“初始化模块”。

**2.3.1 Vim设置**

编写测试文件one.sls

[root@saltstack-master ~]# vi /srv/salt/base/init/one.sls

first-sls:

file.managed:

- name: /tmp/foo.conf

- source: salt://init/config/foo.conf

- user: root

- group: root

- mode: 644

#test

[root@saltstack-master init]# salt '\*' state.sls init.one test=True

根据使用习惯设置统一的vim配置文件，使用SaltStack的File状态模块的Managed方法管理vimrc文件。

查看指定states的function及指定state用法：

[root@saltstack-master ~]# salt '\*' sys.list\_state\_functions file

salt '\*' sys.state\_doc file.managed

[root@saltstack-master ~]# mkdir -p /srv/salt/base/init

[root@saltstack-master ~]# mkdir -p /srv/salt/base/config

[root@saltstack-master ~]# cp /etc/vimrc /srv/salt/base/config/

[root@saltstack-master ~]# vim /srv/salt/base/init/vim.sls

/etc/vimrc:

file.managed:

- source: salt://init/config/vimrc

- user: root

- group: root

- mode: 644

- backup: '\*'

SLS文件编写完成之后，需要把/etc/vimrc文件放到/srv/salt/base/init/config目录下面。

注：SaltStack环境下面的目录不存在的都需要新建。

[root@saltstack-master ~]# salt '\*' state.sls init.vim test=True #test参数测试是否能够同步成功，

[root@saltstack-master ~]# salt '\*' sys.doc state | less #查看stata模块用法

saltstack-master.example.com:

----------

ID: sync\_vimrc

Function: file.managed

Name: /etc/vimrc

Result: True

Comment: The file /etc/vimrc is in the correct state

Started: 10:53:08.302890

Duration: 7.408 ms

Changes:

Summary

------------

Succeeded: 1

Failed: 0

------------

Total states run: 1

saltstack-minion.example.com:

----------

ID: sync\_vimrc

Function: file.managed

Name: /etc/vimrc

Result: None

Comment: The file /etc/vimrc is set to be changed

Started: 10:53:08.967117

Duration: 6.296 ms

Changes:

----------

newfile:

/etc/vimrc

Summary

------------

Succeeded: 1 (unchanged=1, changed=1)

Failed: 0

------------

Total states run: 1

**2.3.2 DNS设置**

生产环境中，DNS解析是比较重要的设置，建议在内网建立自己的内网DNS服务器，同样使用SlatStack的File状态模块中的Managed方法管理resolv.conf文件：

[root@saltstack-master ~]# cp /etc/resolv.conf /srv/salt/base/init/config/

[root@saltstack-master ~]# vim /srv/salt/base/init/dns.sls

/etc/resolv.conf:

file.managed:

- source: salt://init/config/resolv.conf

- user: root

- group: root

- mode: 644

- backup: '\*'

dns.sls文件编写完成之后，需要把设置好的resolv.conf放到/srv/salt/base/init/config目录下面。

**2.3.3 History记录时间**

使用history记录时间，可以清楚的知道什么用户什么时间执行了什么命令，对分析系统错误，及安全性有很大帮助，使用SlatStack的File状态模块的Append方法，在/etc/profile里面追加设置：（相当于echo “” >> file）

[root@saltstack-master ~]# salt '\*’ sys.state\_doc file.append | grep -C 5 append

[root@saltstack-master ~]# vim /srv/salt/base/init/history.sls

/etc/profile:

file.append:

- text:

- export HISTTIMEFORMAT="%F %T `whoami` "

#注：编写SLS文件时，使用英文输入法，不然会导致相关报错（Illegal tab character）

2.3.4 命令操作审计

使用logger将输入的命令写入到memssages的一个简单功能，使用SaltStack的File模块的Append方法。建议将memssages日志文件进行统一收集管理，建议使用ELK Stack(Elasticsearch、LogStach、Kibana)。

append\_log:

file.append:

- name: /etc/bashrc

- text:

- export PROMPT\_COMMAND='{ msg=$(history 1 | { read x y; echo $y; });logger "[euid=$(whoami)]":$(who am i):[`pwd`]"$msg"; }'

cmd.run:

- name: source /etc/bashrc

**2.3.5 内核参数优化**

初始化时，需要对默认的内核参数进项调优，SaltStack提供了Sysctl状态模块用来检测内核参数的配置，默认调整的内核参数较多，参考：

<http://blog.sina.com.cn/s/blog_87113ac20102w4za.html>

[root@saltstack-master ~]# cp /etc/sysctl.conf /srv/salt/base/init/config/

[root@saltstack-master ~]# vim /srv/salt/base/init/sysctl.sls

/etc/sysctl.conf:

file.managed:

- source: salt://init/config/sysctl.conf

- user: root

- group: root

- mode: 644

通过结果如图2-1

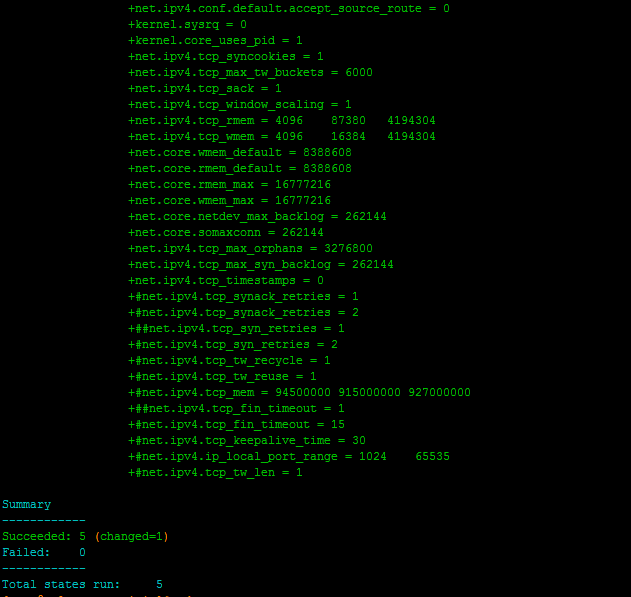


图 2-1

#需要先在本地设置好优化过的内核参数文件，放到/srv/salt/base/init/config目录下面。

参数优化详情可参考上面所示博客，或自行度娘。

[root@saltstack-master ~]# salt '\*' state.sls init.sysctl test=True

**2.3.6 epel仓库**

建议设置epel仓库，放到系统初始化配置当中，由于本教程在安装salt-minion时已经安装过epel源，所以此处只贴出例子，是否需要使用建议在env\_init.sls文件中设置即可。

[root@saltstack-master ~]# vim /srv/salt/base/init/epel.sls

yum\_repo\_release:

pkg.installed:

- sources:

- epel-release: http://mirrors.aliyun.com/epel/6/x86\_64/epel-release-6-8.noarch.rpm

- unless: rpm -qa | grep epel-release-6-8

**2.3.7 ssh设置**

建议在生产服务器对ssh配件文件进行统一管理，修改默认的连接端口

[root@saltstack-master ~]# sed -i 's/\#UseDNS yes/UseDNS no/' /etc/ssh/sshd\_config

[root@saltstack-master ~]# sed -i 's/\#PermitEmptyPasswords no/PermitEmptyPasswords no/' /etc/ssh/sshd\_config

#

[root@saltstack-master ~]# cp /etc/ssh/sshd\_config /srv/salt/base/init/config/

[root@saltstack-master ~]# vim /srv/salt/base/init/ssh.sls

sync-ssh:

file.managed:

- name: /etc/ssh/sshd\_config

- source: salt://init/config/sshd\_config

- user: root

- group: root

- mode: 644

cmd.run:

- name: /etc/init.d/sshd restart

- require:

- file: sync-ssh

service.running:

- name: sshd

- enable: True

- reload: True

- require:

- file: sync-ssh

**2.3.8 crontab设置**

设置定时任务同步系统时间

[root@saltstack-master ~]# vim /srv/salt/base/init/cron.sls

ntpdate-init:

pkg.installed:

- name: ntpdate

set-crontab:

cron.present:

- name: /usr/bin/ntpdate times.aliyun.com >> /dev/null 2>&1

- user: root

- minute: '\*5'

[root@saltstack-master ~]# salt '\*' state.sls init.cron test=True

**2.3.9 安装常用命令**

[root@saltstack-master ~]# vim /srv/salt/base/init/yum.sls

yum-list-init:

pkg.installed:

- names:

- gcc

- gcc-c++

- man

- vim-enhanced

- wget

- telnet

- lsof

- sysstat

- openssh-clients

- lrzsz

- tree

- hdparm

#

**2.3.7 初始化环境引用**

本教程编写的初始化功能SLS文件，统一放到init目录下，方便理解和管理，可以通过在编写一个特别的SLS文件，把init目录下面的初始化功能SLS文件包含进去，然后在top.sls直接引用这个sls文件即可：

[root@saltstack-master ~]# vim /srv/salt/base/init/env\_init.sls

include:

- init.one

- init.vim

- init.dns

- init.history

- init.log

- init.sysctl

- init.epel

- init.ssh

- init.cron

- init.yum

其中one.sls文件是最开始为了测试时创建的sls文件，此处建议在开始编写sls进行同步时，先编写one.sls，然后进行单个sls文件同步测试，下面是从saltstack-master同步到\*的演示。每新增一个功能模块的sls文件，都需要测试同步，同时saltstack-master，salt-minion中日志的级别建议设置成debug，方便排错。

[root@saltstack-master ~]# salt '\*' state.sls init.one

\*:

----------

ID: /tmp/foo.conf

Function: file.managed

Result: True

Comment: File /tmp/foo.conf is in the correct state

Started: 19:05:42.311064

Duration: 13.934 ms

Changes:

Summary

------------

Succeeded: 1

Failed: 0

------------

Total states run: 1

查看到此我们已经编写的sls文件，通过tree命令，最小化安装的CentOS 6.7默认没安装tree，需自行yum安装即可：

[root@saltstack-master ~]# tree /srv/salt/base/

/srv/salt/base/

├── init

│   ├── config

│   │   ├── foo.conf

│   │   ├── resolv.conf

│   │   ├── sshd\_config

│   │   ├── sysctl.conf

│   │   └── vimrc

│   ├── cron.sls

│   ├── del\_cron.sls

│   ├── dns.sls

│   ├── env\_init.sls

│   ├── epel.sls

│   ├── history.sls

│   ├── log.sls

│   ├── one.sls

│   ├── ssh.sls

│   ├── sysctl.sls

│   ├── vim.sls

│   └── yum.sls

└── top.sls

2 directories, 18 files

编写top.sls文件，给Minion指定状态并执行：

[root@saltstack-master ~]# vim /srv/salt/base/top.sls

base:

'\*':

- init.env\_init

注意：生产环境中，每次执行状态，强烈建议先进性测试，确定SaltStack会执行那些操作然后在应用状态到服务器上：

测试：

[root@saltstack-master ~]# salt '\*' state.highstate test=True

注：建议这里不要用salt ‘\*’ state.highstate test=True，需要指定到那台服务器，用正则匹配到指定服务器，避免导致不必要的错误。

…….

Summary

-------------

Succeeded: 24 (unchanged=15, changed=4)

Failed: 0

-------------

Total states run: 24

如果出现上图所示，表示编写的sls文件可以正常执行，然后同步到指定的服务器上面。

#

[root@saltstack-master ~]# salt '\*' state.highstate

下面是把初始化设置应用\*显示结果：（已成功）

[root@saltstack-master ~]# salt '\*' state.highstate

\*:

----------

ID: /tmp/foo.conf

Function: file.managed

Result: True

Comment: File /tmp/foo.conf is in the correct state

Started: 19:29:09.696053

Duration: 6.285 ms

Changes:

----------

ID: /etc/resolv.conf

Function: file.managed

Result: True

Comment: File /etc/resolv.conf is in the correct state

Started: 19:29:09.702465

Duration: 2.294 ms

Changes:

----------

ID: /etc/salt/minion

Function: file.managed

Result: True

Comment: File /etc/salt/minion is in the correct state

Started: 19:29:09.704881

Duration: 2.543 ms

Changes:

----------

ID: /etc/profile

Function: file.append

Result: True

Comment: File /etc/profile is in correct state

Started: 19:29:09.707537

Duration: 1.06 ms

Changes:

----------

ID: /etc/sysctl.conf

Function: file.managed

Result: True

Comment: File /etc/sysctl.conf is in the correct state

Started: 19:29:09.708709

Duration: 2.32 ms

Changes:

Summary

------------

Succeeded: 5

Failed: 0

------------

Total states run: 5

**3 功能模块设置**

初始化系统完成之后，编写具体的功能模块。参照图1-1案例架构图从上往下进行设计与实现，首先编写Haproxy和Keepalived功能模块

**3.1 Haproxy配置管理**

Haproxy是一个开源的高性能的反向代理项目，支持四层和七层的负载均衡，多种负载均衡算法和健康检查等。

Keepalived是一个高可用集群的项目，它是VRRP协议的完美实现，通过Keepalived来管理Haproxy上面的VIP，当注Haproxy发生故障时，将VIP漂移到备用的Haproxy上来继续提供服务。

Haproxy和Keepalived使用源码编译安装的方式，将这两个服务放置在prod环境中。

首先创建目录结构，如下所示：

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/pkg

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/haproxy/package

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/keepalived/package

在每个服务的目录线面创建一个package目录用来存放软件的源码包和需要的相关启动脚本、配置文件等。

**3.1.1 pkg配置**

首先需要使用pkg模块将源码编译依赖的各种包都安装上，使用pkg状态的installed方法，同时使用names列表，通过列表的方式把需要的安装包都列出来：

[root@saltstack-master ~]# vim /srv/salt/prod/pkg/pkg-init.sls

pkg-init:

pkg.installed:

- name:

- gcc

- gcc-c++

- glibc

- make

- autoconf

- openssl

- openssl-devel

**3.1.2 Haproxy服务配置**

首先需要将Haproxy的源码包和管理脚本放置在/srv/salt/prod/haproxy/package目录下，通过<http://www.haproxy.org/下载软件包，这里使用1.6.5>版本。

[root@saltstack-master ~]# wget [http://fossies.org/linux/misc/haproxy-1.6.5.tar.gz -P /usr/local/src/](http://fossies.org/linux/misc/haproxy-1.6.5.tar.gz%20-P%20/usr/local/src/)

由于haproxy官网wget较慢，此处可是使用其他源进行wget或者本地下载完成之后上传到服务器的/usr/local/src目录，建议使用MD5验证文件的完整性。

[root@saltstack-master ~]# cd /usr/local/src/

[root@saltstack-master src]# cp haproxy-1.6.5.tar.gz /srv/salt/prod/haproxy/package/

[root@saltstack-master src]# tar zxvf haproxy-1.6.5.tar.gz

[root@saltstack-master src]# cd /usr/local/src/haproxy-1.6.5/examples/

该目录下存放了Haproxy启动脚本，需要修改默认路径：

[root@saltstack-master examples]# sed -i 's/\/usr\/sbin\/'\$BASENAME'/\/usr\/local\/haproxy\/sbin\/'\$BASENAME'/g' haproxy.init

复制Haproxy的启动脚本到/srv/salt/prod/haproxy/package/下面：

[root@saltstack-master examples]# cp haproxy.init /srv/salt/prod/haproxy/package/

**3.1.3 编写Haproxy安装SLS文件**

编写Haproxy内容如下：

[root@saltstack-master ~]# vim /srv/salt/prod/haproxy/install.sls

include:

- pkg.pkg-init:

haproxy-install:

file.managed:

- name: /usr/local/src/haproxy-1.6.5.tar.gz

- source: salt://haproxy/package/haproxy-1.6.5.tar.gz

- mode: 755

- user: root

- group: root

cmd.run:

- name: cd /usr/local/src && tar zxf haproxy-1.6.5.tar.gz && cd haproxy-1.6.5 && make TARGET=linux26 PREFIX=/usr/local/haproxy && make install PREFIX=/usr/local/haproxy

- unless: test -d /usr/local/haproxy

- require:

- pkg: pkg-init

- file: haproxy-install

Haproxy的服务管理脚本如下所示：

/etc/init.d/haproxy:

file.managed:

- source: //haproxy/package/haproxy.init

- mode: 755

- user: root

- group: root

- require:

- cmd: haproxy-install

设置可以监听非本地IP：

net.ipv4.ip\_nonlocal\_bind:

sysctl.present:

- value: 1

#Haproxy的配置文件存放目录如下：

haproxy-config-dir:

file.directory:

- name: /etc/haproxy

- mode: 755

- user: root

- group: root

#设置Haproxy开机自启动

haproxy-init:

cmd.run:

- name: chkconfig -add haproxy

- unless: chkconfig --list | grep haproxy

- require:

- file: /etc/init.d/haproxy

本文没有把Haproxy的服务管理放置在install.sls里面，因为Haproxy启动需要依赖配置文件，通过两种方法管理Haproxy的配置文件：

直接在需要使用Haproxy的地方引用Haproxy的安装，然后加入Haproxy的配置文件和服务管理。优点：简单明了；缺点：不够灵通用。

使用jinja模板，将Haproxy的基础配置编写完成之后，其他的配置通过Pillar来进行自动生成。优点：灵活通用；缺点：由于需要使用大量的if、for等Jinja模板语法，而且需要配置Pillar来实现配置，比较麻烦，实现起来难度比较大，而且容易出错。

**3.14 Haproxy业务引用**

编写一个业务模块Cluster，然后调用Haproxy来完成配置管理，这样做的好处是把基础服务的配置管理和业务分开，例如负载均衡，有可能是对外的，也可能是内部使用，如果都是用Haproxy，那么Haproxy的安装就是基础功能，配置和启动，可以通过放置在业务模块Cluster中来进行集中管理。

创建cluster目录，并且在cluster目录下创建config目录，用来存放配置文件：

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/cluster/config

将haproxy的配置文件放置在/srv/salt/prod/cluster/config目录下，下面列出本次案例使用的最小化配置：

[root@saltstack-master ~]# vim /srv/salt/prod/cluster/config/haproxy-outside.cfg

global # ------全局配置------

log 127.0.0.1 local0 #日志输出配置，所有日志都记录在本机，通过local0输出

log 127.0.0.1 local1 notice

#log loghost local0 info

maxconn 100000 #最大连接数

chroot /usr/share/haproxy #chroot运行路径

uid 99 #所属用户UID

gid 99 #所属运行的GID

daemon #以后台形式运行haproxy

#debug #调试模式，输出启动信息到标准输出

#quiet #安静模式，启动时无输出

defaults #------默认配置-----

log global

mode http #默认模式{tcp|http|health},tcp是4层,http是7层，health只会返回OK

option httplog #日志类别:http日志格式

option dontlognull #不记录健康检查的日志信息

retries 3 #3次连接失败就认为服务不可用

option redispatch #ServerID对应的服务器挂掉后,强制定向到其他健康服务器

maxconn 100000 #默认最大连接数

timeout connect 5000 #连接超时

timeout client 50000 #客户端超时

timeout server 50000 #服务端超时

listen status #监控页面设置

mode http #http的7层模式

bind 0.0.0.0:8888 #监听端口

stats enable

stats hide-version #隐藏统计页面上的HAproxy版本信息

stats uri /haproxy-status #监控页面URL

stats auth haproxy:saltstack #监控页面用户名和密码

stats admin if TRUE #手工启用、禁用后端服务器

frontend frontend\_www\_vdevops\_com

bind 192.168.1.154:80

mode http

option httplog

log global

default\_backend backend\_www\_vdevops\_com

backend backend\_www\_vdevops\_com

option forwardfor header X-REAL-IP

option httpchk HEAD / HTTP/1.0

balance source

server web-node1 192.168.1.158:80 cookie server01 check inter 2000 rise 30 fall 15

server web-node2 192.168.1.151:80 cookie server02 check inter 2000 rise 30 fall 15

##服务器定义(check指健康状况检查，inter 2000指检测频率;rise 2指从离线状态转换至正常状态需要成功检查的次数；fall 3指失败3次即认为服务器不可用)

编写Haproxy服务管理的SLS文件

[root@saltstack-master ~]# vim /srv/salt/prod/cluster/haproxy-service.sls

include:

- haproxy.install

haproxy-service:

file.managed:

- name: /etc/haproxy/haproxy.cfg

- source: salt://cluster/files/haproxy-outside.cfg

- user: root

- group: root

- mode: 644

service.running:

- name: haproxy

- enable: True

- reload: True

- require:

- cmd: haproxy-install

- watch:

- file: haproxy-service

**3.执行Haproxy状态**

编写完成Haproxy的状态配置后，需要在Top file’里面给Minion指定状态。

[root@saltstack-master ~]# vim /srv/salt/base/top.sls

base:

'\*':

- init.env\_init

prod:

'\*.example.com':

- cluster.haproxy-service

测试并执行状态如下所示：

[root@saltstack-master ~]# salt '\*' state.highstate test=True

--------------------

ID: haproxy-service

Function: service.running

Name: haproxy

Result: None

Comment: Service is set to be started

Started: 08:54:23.478441

Duration: 11.717 ms

Changes:

Summary

-------------

Succeeded: 40 (unchanged=11, changed=3)

Failed: 0

-------------

Total states run: 40

[root@saltstack-master ~]# salt '\*' state.highstate

----------

ID: haproxy-service

Function: service.running

Name: haproxy

Result: True

Comment: Service haproxy is already enabled, and is running

Started: 09:33:51.822457

Duration: 72.527 ms

Changes:

----------

haproxy:

True

Summary

-------------

Succeeded: 40 (changed=5)

Failed: 0

-------------

Total states run: 40

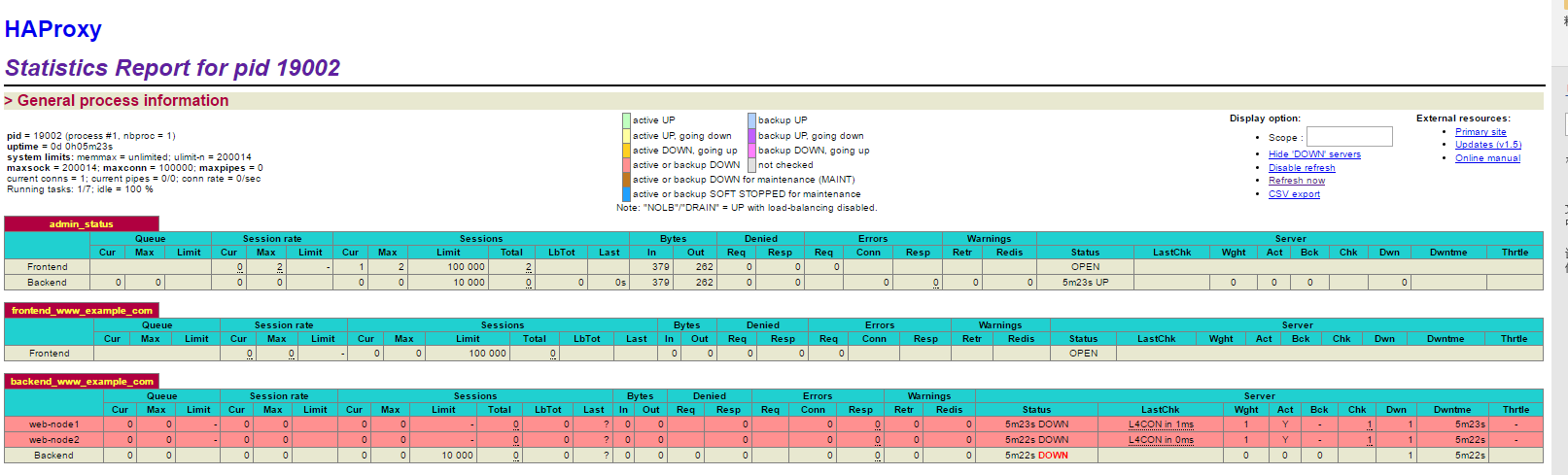
3.1.5 查看Haproxy状态

执行完毕如果没有报错，就表示Haproxy已经正常启动啦，如果有报错查看日志，来定位问题。执行之前确保minion客户端的80和8888端口没有被占用，通过<http://IP:8888/status>查看haproxy的状态，登录账号：haproxy 密码：saltstack

登录成功如下图所示：

#http://10.1.1.97:8888/status

# <http://10.1.1.98:8888/status>



由于前后端的web服务还未启动，目前看到的是Down状态。

3.2 Keepalived 配置管理

放置源码包、Keepalived的启动脚本、sysconfig配置文件在/srv/salt/prod/keepalived/files目录下。

#Master端

[root@saltstack-master ~]# cd /usr/local/src/

[root@saltstack-master src]# wget <http://www.keepalived.org/software/keepalived-1.2.22.tar.gz>

[root@saltstack-master src]# cd /srv/salt/prod/keepalived/files/

[root@saltstack-master files]# tar zxvf keepalived-1.2.22.tar.gz && cd keepalived-1.2.22

将Keepalived需要的init脚本和sysconfig复制到files目录下：

[root@saltstack-master keepalived-1.2.22]# cp keepalived/etc/init.d/keepalived.init /srv/salt/prod/keepalived/files/

[root@saltstack-master keepalived-1.2.22]# cp keepalived/etc/init.d/keepalived.sysconfig /srv/salt/prod/keepalived/files/

修改源码包里面的init脚本

[root@saltstack-master files]# cd /srv/salt/prod/keepalived/files/

[root@saltstack-master files]# rm -rf keepalived-1.2.22

#vim keepalived.init

#将daemon keepalived ${KEEPALIVED\_OPTIONS}

#修改为 daemon /usr/local/keepalived/sbin/keepalived${KEEPALIVED\_OPTIONS}

或者使用sed直接修改：

# [root@saltstack-master files]# sed -i 's/ daemon keepalived \${KEEPALIVED\_OPTIONS}/ daemon \/usr\/local\/keepalived\/sbin\/keepalived \${KEEPALIVED\_OPTIONS}/' keepalived.init

[root@saltstack-master files]# grep daemon keepalived.init

# Startup script for the Keepalived daemon

daemon /usr/local/keepalived/sbin/keepalived${KEEPALIVED\_OPTIONS}

3.2.1 编写Keepalived安装sls

[root@saltstack-master ~]# vim /srv/salt/prod/keepalived/install.sls

include:

- pkg.pkg-init

keepalived-install:

file.managed:

- name: /usr/local/src/keepalived-1.2.22.tar.gz

- source: salt://keepalived/files/keepalived-1.2.22.tar.gz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/local/src && tar zxf keepalived-1.2.22.tar.gz && cd keepalived-1.2.22 && ./configure --prefix=/usr/local/keepalived --disable-fwmark && make && make install

- unless: test -d /usr/local/keepalived

- require:

- file: keepalived-install

keepalived-sysconfig:

file.managed:

- name: /etc/sysconfig/keepalived

- source: salt://keepalived/files/keepalived.sysconfig

- user: root

- group: root

- mode: 755

keepalived-init:

file.managed:

- name: /etc/init.d/keepalived

- source: salt://keepalived/files/keepalived.init

- user: root

- group: root

- mode: 755

cmd.run:

- name: chkconfig keepalived on

- unless: chkconfig --list | grep keepalived

- require:

- file: keepalived-init

keepalived-dir:

file.directory:

- name: /etc/keepalived

- user: root

- group: root

- mode: 744

3.2.2 业务模块

[root@saltstack-master ~]# cd /srv/salt/prod/cluster/files/

[root@saltstack-master files]# vim haproxy-service-keepalived.conf

! Configuration File for keepalived

global\_defs {

notification\_email {

saltstack@example.com

}

notification\_email\_from keepalived@example.com

smtp\_server 127.0.0.1

smtp\_connect\_timeout 30

router\_id {{ROUTEID}}

}

vrrp\_instance haproxy\_ha {

state {{STATEID}}

interface eth0

virtual\_router\_id 36

priority {{PRIORITYID}}

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

10.1.1.92

}

}

在cluster业务目录下面编写haproxy使用Keepalived做高可用的sls文件

[root@saltstack-master files]# cd /srv/salt/prod/cluster/

[root@saltstack-master cluster]# vim haproxy-service-keepalived.sls

include:

- keepalived.install

keepalived-service:

file.managed:

- name: /etc/keepalived/keepalived.conf

- source: salt://cluster/files/haproxy-service-keepalived.conf

- user: root

- group: root

- mode: 644

- template: jinja

{% if grains['fqdn'] == 'saltstack-master.example.com' %}

- ROUTEID: haproxy\_ha

- STATEID: MASTER

- PRIORITYID: 150

{% elif grains['fqdn'] == 'saltstack-minion.example.com' %}

- ROUTEID: haproxy\_ha

- STATEID: BACKUP

- PRIORITYID: 100

{% endif %}

service.running:

- name: keepalived

- enable: True

- watch:

- file: keepalived-service

3.2.3 执行Keepalived状态

编写Keepalived状态管理sls，在top file中指定Minion运行状态。

[root@saltstack-master cluster]# cd /srv/salt/base/

[root@saltstack-master base]# vim top.sls

base:

'\*':

- init.env\_init

prod:

'\*.example.com':

- cluster.haproxy-service

- cluster.haproxy-service-keepalived

#

[root@saltstack-master ~]# tree /srv/salt/prod/cluster/

/srv/salt/prod/cluster/

├── files

│   ├── haproxy-service.cfg

│   └── haproxy-service-keepalived.conf

├── haproxy-service-keepalived.sls

└── haproxy-service.sls

1 directory, 4 files

#测试

[root@saltstack-master base]# salt '\*' state.sls cluster.haproxy-service-keepalived test=True env=prod

[root@saltstack-master base]# salt '\*' state.highstate test=True

Summary

-------------

Succeeded: 48 (unchanged=10, changed=5)

Failed: 0

-------------

#执行

[root@saltstack-master files]# salt '\*' state.highstate

Total states run: 48

----------

ID: keepalived-service

Function: service.running

Name: keepalived

Result: True

Comment: Service keepalived is already enabled, and is running

Started: 10:45:31.813269

Duration: 104.633 ms

Changes:

----------

keepalived:

True

Summary

-------------

Succeeded: 48 (changed=5)

Failed: 0

-------------

Total states run: 48

#

3.2.4 haproxy+Keepalived 测试

执行完毕状态后，目前服务器已经正常运行，saltstack-master.example.com是主节点，使用ip ad li 查看目前的VIP是否在该节点：

[root@saltstack-master ~]# ip ad li

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:0c:29:66:1e:aa brd ff:ff:ff:ff:ff:ff

inet 10.1.1.97/24 brd 10.1.1.255 scope global eth0

inet 10.1.1.92/32 scope global eth0

inet6 fe80::20c:29ff:fe66:1eaa/64 scope link

valid\_lft forever preferred\_lft forever

#关闭主节点的keepalived进程，模拟服务器宕机，然后再次查看VIP：

#Master节点

[root@saltstack-master ~]# ip ad li eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:0c:29:66:1e:aa brd ff:ff:ff:ff:ff:ff

inet 10.1.1.97/24 brd 10.1.1.255 scope global eth0

inet6 fe80::20c:29ff:fe66:1eaa/64 scope link

valid\_lft forever preferred\_lft forever

#Slave节点

[root@saltstack-minion ~]# ip ad li eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:50:56:3b:05:ad brd ff:ff:ff:ff:ff:ff

inet 10.1.1.98/24 brd 10.1.1.255 scope global eth0

inet 10.1.1.92/32 scope global eth0

inet6 fe80::250:56ff:fe3b:5ad/64 scope link

valid\_lft forever preferred\_lft forever

当master的keepalived down掉后，VIP会飘到backup上。

#重启Master节点的Keepalived进程，发现VIP已经切换到主节点

[root@saltstack-master ~]# /etc/init.d/keepalived start

Starting keepalived: [ OK ]

[root@saltstack-master ~]# ip ad li eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:0c:29:66:1e:aa brd ff:ff:ff:ff:ff:ff

inet 10.1.1.97/24 brd 10.1.1.255 scope global eth0

inet 10.1.1.92/32 scope global eth0

inet6 fe80::20c:29ff:fe66:1eaa/64 scope link

valid\_lft forever preferred\_lft forever

#Backup端VIP已不见。

[root@saltstack-minion ~]# ip ad li eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:50:56:3b:05:ad brd ff:ff:ff:ff:ff:ff

inet 10.1.1.98/24 brd 10.1.1.255 scope global eth0

inet6 fe80::250:56ff:fe3b:5ad/64 scope link

valid\_lft forever preferred\_lft forever

3.4 Memcached 配置管理

Memcached是一个高性能的分布式内存对象缓存系统，用于动态web应用以减轻数据库负载，它通过内存中缓存数据和对象来减少读取数据库的次数，从而提高动态数据库驱动网站的访问速度，本次架构使用Memcached来存放存储（后面添加Redis）用户的Session。

负载均衡的环境下遇到的session问题，一般解决方法有三种：

Session保持

Session复制

Session共享

PHP可以在php.ini配置将session存储到memcached中，来实现session共享，这样可以避免后端服务器某一节点宕机时，造成用户请求丢失，用户的访问请求被调度到集群中的其他节点，用户的会话不会丢失。

Memcached的安装比较简单，Memcached依赖于libevent，需要先编译安装libevent，然后编译安装Memcached，同时创建一个管理用户的配置文件，Memcached包括后面要配置的Nginx和PHP都需要www用户进行管理。

Libevent 是一个用C语言编写的、轻量级的开源高性能网络库，主要有以下几个亮点：事件驱动（ event-driven），高性能;轻量级，专注于网络，不如 ACE 那么臃肿庞大；源代码相当精炼、易读；跨平台，支持 Windows、 Linux、 \*BSD 和 Mac Os；支持多种 I/O 多路复用技术， epoll、 poll、 dev/poll、 select 和 kqueue 等；支持 I/O，定时器和信号等事件；注册事件优先级。

#Master端，创建目录结构

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/libevent/files

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/memcached/files

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/user

3.4.1 www用户配置

启动Memcached使用www用户，后面部署Nginx和PHP也使用www用户。

[root@saltstack-master ~]# vim /srv/salt/prod/user/www.sls

www-user-group:

group.present:

- name: www

- gid: 1500

user.present:

- name: www

- fullname: www

- shell: /sbin/nologin

- uid: 1500

- gid: 1500

3.4.2 Libevent配置

[root@saltstack-master ~]# cd /usr/local/src/

[root@saltstack-master src]# wget <http://ufpr.dl.sourceforge.net/project/levent/release-2.0.22-stable/libevent-2.0.22-stable.tar.gz>

[root@saltstack-master src]# cp libevent-2.0.22-stable.tar.gz /srv/salt/prod/libevent/files/

#编写libevent部署SLS：

[root@saltstack-master ~]# vim /srv/salt/prod/libevent/install.sls

libevent-source-install:

file.managed:

- name: /usr/local/src/libevent-2.0.22-stable.tar.gz

- source: salt://libevent/files/libevent-2.0.22-stable.tar.gz

- user: root

- group: root

- mode: 644

cmd.run:

- name: cd /usr/local/src/ && tar zxf libevent-2.0.22-stable.tar.gz && cd libevent-2.0.22-stable && ./configure --prefix=/usr/local/libevent && make && make install

- unless: test -d /usr/local/libevent

- require:

- file: libevent-source-install

3.2.5 Memcached部署

[root@saltstack-master ~]# cd /srv/salt/prod/memcached/files/

[root@saltstack-master files]# wget <http://memcached.org/files/memcached-1.4.27.tar.gz>

#编写Memcached部署sls

[root@saltstack-master files]# vim /srv/salt/prod/memcached/install.sls

include:

- libevent.install

memcached-source-install:

file.managed:

- name: /usr/local/src/memcached-1.4.27.tar.gz

- source: salt://memcached/files/memcached-1.4.27.tar.gz

- user: root

- group: root

- mode: 644

cmd.run:

- name: cd /usr/local/src && tar zxf memcached-1.4.27.tar.gz && cd memcached-1.4.27 && ./configure --prefix=/usr/local/memcached -with-libevent=/usr/local/libevent && make && make install

- unless: test -d /usr/local/memcached

- require:

- cmd: libevent-source-install

- file: memcached-source-install

3.2.6 Memcached服务

安装完毕Memcached后，需要启动Memcached，Memcached源码包中提供了init的服务器管理脚本，我们可以将Memcached的参数写入Memcached.conf里面，启动是调用即可。

[root@saltstack-master files]# vim service.sls

include:

- memcached.install

- user.www

memcached-server:

cmd.run:

- name: /usr/local/memcached/bin/memcached -d -m 1024 -p 11211 -c 4096 -u www

- unless: netstat -nltp | grep 11211

- require:

- cmd: memcached-source-install

- user: www-user-group

memcached-daemon:

cmd.run:

- name: echo "/usr/local/memcached/bin/memcached -d -m 1024 -p 11211 -c 4096 -u www" >> /etc/rc.d/rc.local

- unless: grep memcached /etc/rc.d/rc.local

#执行Memcached状态

#在top file对minion进行指定

base:

'\*':

- init.env\_init

prod:

'\*.example.com':

- cluster.haproxy-service

- cluster.haproxy-service-keepalived

'saltstack-minion.example.com'

- memcached.service

#测试

[root@saltstack-master ~]# salt 'saltstack-minion.example.com' state.sls memcached.service test=True env=prod

Summary

------------

Succeeded: 8 (unchanged=8, changed=2)

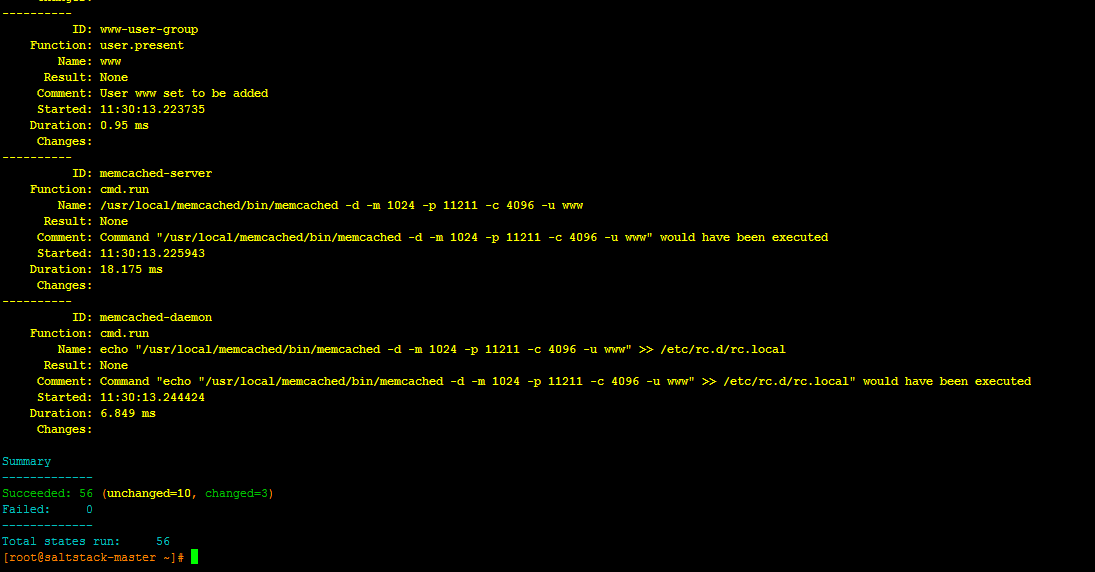
Failed: 0

------------

Total states run: 8

#执行

[root@saltstack-master ~]# salt '\*' state.highstate test=True



3.3 Nginx配置管理

Haproxy+Keepalived自动化配置完成之后，进行Nginx+PHP的自动化配置，同样使用源码包安装的方式进行编译安装。

编写稍微复杂的状态功能模块时，首先进行规划，包括如何设计目录结构，需要应用到那些状态模块和状态件的关系，是否需要Grains和Pillar等。

Nginx+PHP（FastCGI）需要安装的包首先由Nginx和PHP，需要进行编译安装，步骤如下：

所有源码包的编译安装需要依赖一些基础软件包，像gcc、make，初始化环境编写的pkg-init.sls，需要的地方可以直接调用。

源码编译安装Nginx是需要依赖PCRE，需要单独编写安装PCRE的模块，然后Nginx调用即可。

注释：PCRE(Perl Compatible Regular Expressions)是一个Perl库，包括 perl 兼容的正则表达式库。这些在执行正规表达式模式匹配时用与Perl 5同样的语法和语义是很有用的。Boost太庞大了，使用boost regex后，程序的编译速度明显变慢。测试了一下，同样一个程序，使用boost::regex编译时需要3秒，而使用pcre不到1秒。因此改用pcre来解决C语言中使用正则表达式的问题

需要编译安装PHP，同时除了PHP常用的模块外，还应该支持如Memcached和Redis这样的生产常用的第三方模块。

常用使用到的功能函数如下：

使用状态模块：file、cmd、service

使用状态间的关系：require、unless

SLS之间的调用：include

#Master端 创建目录结构

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/pcre/files

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/nginx/files

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/php/files

#下载所需的源码包，并放到各个服务的files目录下：

[root@saltstack-master files]# cd /srv/salt/prod/pcre/files/

[root@saltstack-master files]# wget <https://sourceforge.net/projects/pcre/files/pcre/8.39/pcre-8.39.tar.gz>

#nginx

[root@saltstack-master ~]# cd /srv/salt/prod/nginx/files/

[root@saltstack-master files]# wget <http://nginx.org/download/nginx-1.10.1.tar.gz>

#php

[root@saltstack-master ~]# cd /srv/salt/prod/php/files/

[root@saltstack-master ~]# wget <http://php.net/distributions/php-7.0.8.tar.gz>

3.3.1 PCRE模块

PCRE模块主要是pcre的安装“

#Master端

[root@saltstack-master files]# cd /srv/salt/prod/pcre/

[root@saltstack-master pcre]# vim install.sls

pcre-install:

file.managed:

- name: /usr/local/src/pcre-8.39.tar.gz

- source: salt://pcre/files/pcre-8.39.tar.gz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/loca/src && tar zxf pcre-8.39.tar.gz && cd pcre-8.39 && ./configure --prefix=/usr/local/pcre && make && make install

- unless: test -d /usr/local/pcre

- require:

- file: pcre-install

3.3.2 Nginx模块

#Master端，编写部署nginx的SLS

[root@saltstack-master pcre]# cd /srv/salt/prod/nginx

[root@saltstack-master nginx]# vim install.sls

include:

- pcre.install

- user.www

nginx-install:

file.managed:

- name: /usr/local/src/nginx-1.10.1.tar.gz

- source: salt:/nginx/files/nginx-1.10.1.tar.gz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/local/src && tar zxf nginx-1.10.1.tar.gz && cd nginx-1.10.1 && ./configure --prefix=/opt/nginx --user=www --group=www --with-http\_ssl\_module --with-http\_stub\_status\_module --with-file-aio --with-http\_dav\_module --with-pcre=/usr/local/src/pcre-8.39 && make && make install && chown -R www:www /opt/nginx

- unless: test -d /opt/nginx

- require:

- user: www-user-group

- file: nginx-install

- pkg: pkg-init

- cmd: pcre-install

#sed -i -e 's/1.10.1//g' -e 's/nginx\//WS/g' -e 's/"NGINX"/"WS"/g' /usr/local/src/nginx-1.10.1/src/core/nginx.h #hidden nginx version

#nginx配置文件,配置文件相关参数自行调整

[root@saltstack-master files]# cd /srv/salt/prod/nginx/files/

[root@saltstack-master files]# vim nginx.conf

user www www;

worker\_processes 2;

error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

pid logs/nginx.pid;

worker\_rlimit\_nofile 65535;

events {

use epoll;

worker\_connections 10240;

}

http {

include mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log off;

#append

server\_names\_hash\_bucket\_size 128;

client\_header\_buffer\_size 32k;

large\_client\_header\_buffers 4 32k;

client\_max\_body\_size 50m;

fastcgi\_connect\_timeout 300;

fastcgi\_send\_timeout 300;

fastcgi\_read\_timeout 300;

fastcgi\_buffer\_size 64k;

fastcgi\_buffer\_size 64k;

fastcgi\_buffers 4 64k;

fastcgi\_busy\_buffers\_size 128k;

fastcgi\_temp\_file\_write\_size 256k;

sendfile on;

tcp\_nopush on;

keepalive\_timeout 65;

gzip on;

gzip\_min\_length 1k;

gzip\_buffers 4 16k;

gzip\_http\_version 1.1;

gzip\_comp\_level 2;

gzip\_types text/plain application/javascript application/x-javascript text/javascript text/css application/xml application/xml+rss;

gzip\_vary on;

gzip\_proxied expired no-cache no-store private auth;

gzip\_disable "MSIE [1-6]\.";

tcp\_nodelay on;

server\_tokens off;

server {

listen 80;

server\_name 127.0.0.1;

#charset koi8-r;

access\_log logs/host.access.log main;

location / {

root html;

index index.html index.htm;

}

error\_page 404 /404.html;

location /nginx\_status

{

stub\_status on;

access\_log off;

allow 127.0.0.1

deny all

}

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

#

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's document root

# concurs with nginx's one

#

#location ~ /\.ht {

# deny all;

# another virtual host using mix of IP-, name-, and port-based configuration

#

#server {

# listen 8000;

# listen somename:8080;

# server\_name somename alias another.alias;

# location / {

# root html;

# index index.html index.htm;

# }

#}

# HTTPS server

#

#server {

# listen 443 ssl;

# server\_name localhost;

# ssl\_certificate cert.pem;

# ssl\_certificate\_key cert.key;

# ssl\_session\_cache shared:SSL:1m;

# ssl\_session\_timeout 5m;

# ssl\_ciphers HIGH:!aNULL:!MD5;

# ssl\_prefer\_server\_ciphers on;

# location / {

# root html;

# index index.html index.htm;

# }

#}

include vhost/\*.conf;

}

#nginx daemon脚本

[root@saltstack-master files]# vim nginx-init

#! /bin/sh

# chkconfig: 2345 55 25

# Description: Startup script for nginx webserver on Debian. Place in /etc/init.d and

# run 'update-rc.d -f nginx defaults', or use the appropriate command on your

# distro. For CentOS/Redhat run: 'chkconfig --add nginx'

### BEGIN INIT INFO

# Provides: nginx

# Required-Start: $all

# Required-Stop: $all

# Default-Start: 2 3 4 5

# Default-Stop: 0 1 6

# Short-Description: starts the nginx web server

# Description: starts nginx using start-stop-daemon

### END INIT INFO

# Author: shaonbean

PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

NAME=nginx

NGINX\_BIN=/opt/nginx/sbin/$NAME

CONFIGFILE=/opt/nginx/conf/$NAME.conf

PIDFILE=/opt/nginx/logs/$NAME.pid

case "$1" in

start)

echo -n "Starting $NAME... "

if netstat -tnpl | grep -q nginx;then

echo "$NAME (pid `pidof $NAME`) already running."

exit 1

fi

$NGINX\_BIN -c $CONFIGFILE

if [ "$?" != 0 ] ; then

echo " failed"

exit 1

else

echo " done"

fi

;;

stop)

echo -n "Stoping $NAME... "

if ! netstat -tnpl | grep -q nginx; then

echo "$NAME is not running."

exit 1

fi

$NGINX\_BIN -s stop

if [ "$?" != 0 ] ; then

echo " failed. Use force-quit"

exit 1

else

echo " done"

fi

;;

status)

if netstat -tnpl | grep -q nginx; then

PID=`pidof nginx`

echo "$NAME (pid $PID) is running..."

else

echo "$NAME is stopped"

exit 0

fi

;;

force-quit)

echo -n "Terminating $NAME... "

echo -n "Terminating $NAME... "

if ! netstat -tnpl | grep -q nginx; then

echo "$NAME is not running."

exit 1

fi

kill `pidof $NAME`

if [ "$?" != 0 ] ; then

echo " failed"

exit 1

else

echo " done"

fi

;;

restart)

$0 stop

sleep 1

$0 start

;;

reload)

echo -n "Reload service $NAME... "

if netstat -tnpl | grep -q nginx; then

$NGINX\_BIN -s reload

echo " done"

else

echo "$NAME is not running, can't reload."

exit 1

fi

;;

configtest)

echo -n "Test $NAME configure files... "

$NGINX\_BIN -t

;;

\*)

echo "Usage: $0 {start|stop|force-quit|restart|reload|status|configtest}"

exit 1

;;

esac

#编写nginx服务sls

[root@saltstack-master files]# vim /srv/salt/prod/nginx/service.sls

include:

- nginx.install

nginx-init:

file.managed:

- name: /etc/init.d/nginx

- source: salt://nginx/files/nginx-init

- mode: 755

- user: root

- group: root

cmd.run:

- name: chkconfig --add nginx

- unless: chkconfig --list | grep nginx

- require:

- file: nginx-init

nginx-conf:

file.managed:

- name: /opt/nginx/conf/nginx.conf

- source: salt://nginx/files/nginx.conf

- user: www

- group: www

- mode: 644

nginx-service:

file.directory:

- name: /opt/nginx/conf/vhost

- require:

- cmd: nginx-install

service.running:

- name: nginx

- enable: True

- reload: True

- require:

- cmd: nginx-init

- watch:

- file: /opt/nginx/conf/nginx.conf

#执行测试：

[root@saltstack-master prod]# salt '\*' state.sls nginx.install test=True env=prod

[root@saltstack-master base]# salt '\*' state.highstate test=True

Summary

-------------

Succeeded: 59 (unchanged=9, changed=4)

Failed: 0

-------------

Total states run: 59

#先配置top file

[root@saltstack-master base]# vim top.sls

base:

'\*':

- init.env\_init

prod:

'\*':

- cluster.haproxy-service

- cluster.haproxy-service-keepalived

- nginx.service

'saltstack-minion.example.com':

- memcached.service

3.3.3 PHP（FastCGI）配置管理

编译PHP的源码，使用FastCGI模式，

[root@saltstack-master base]# cd /srv/salt/prod/php/

#编译安装php依赖包安装

#[root@saltstack-master php]# vim pkg-php-init.sls

pkg-php:

pkg.installed:

- names:

- mysql-devel

- openssl-devel

- swig

- libjpeg-turbo

- libjpeg-turbo-devel

- libpng

- libpng-devel

- freetype

- freetype-devel

- libxml2

- libxml2-devel

- zlib

- zlib-devel

- libcurl

- libcurl-devel

- php-pear

#php及插件安装

[root@saltstack-master php]# vim install.sls

include:

- php.pkg-php-init

php-install:

file.managed:

- name: /usr/local/src/php-7.0.8.tar.gz

- source: salt://php/files/php-7.0.8.tar.gz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/local/src && tar zxf php-7.0.8.tar.gz && cd php-7.0.8&& ./configure --prefix=/opt/php-fastcgi --with-pdo-mysql=mysqlnd --with-mysqli=mysqlnd --with-mysql=mysqlnd --with-iconv-dir --with-jpeg-dir --with-png-dir --with-zlib --enable-xml --with-libxml-dir --with-curl --enable-bcmath --enable-shmop --enable-sysvsem --enable-inline-optimization --enable-mbregex --with-openssl --enable-mbstring --with-gd --enable-gd-native-ttf --with-freetype-dir=/usr/lib64 --with-gettext=/usr/lib64 --enable-sockets --with-xmlrpc --enable-zip --enable-soap --disable-debug --enable-opcache --enable-zip --with-config-file-path=/opt/php-fastcgi/etc --enable-fpm --with-fpm-user=www --with-fpm-group=www && make && make install

- require:

- file: php-install

- user: www-user-group

- unless: test -d /opt/php-fastcgi

pdo-plugin:

cmd.run:

- name: cd /usr/local/src/php-7.0.8/ext/pdo\_mysql/ && /opt/php-fastcgi/bin/phpize && ./configure --with-php-config=/opt/php-fastcgi/bin/php-config && make && make install

- unless: test -f /opt/php-fastcgi/lib/php/extensions/\*/pdo\_mysql.so

- require:

- cmd: php-install

php-ini:

file.managed:

- name: /opt/php-fastcgi/etc/php.ini

- source: salt://php/files/php.ini-production

- user: root

- group: root

- mode: 644

php-fpm:

file.managed:

- name: /opt/php-fastcgi/etc/php-fpm.conf

- source: salt://php/files/php-fpm.conf.default

- user: root

- group: root

- mode: 644

php-config:

file.managed:

- name: /opt/php-fastcgi/etc/php-fpm.d/www.conf

- source: salt://php/files/www.conf.default

- user: root

- group: root

- mode: 644

php-fastcgi-service:

file.managed:

- name: /etc/init.d/php-fpm

- source: salt://php/files/init.d.php-fpm

- user: root

- group: root

- mode: 755

cmd.run:

- name: chkconfig --add php-fpm

- unless: chkconfig --list | grep php-fpm

- require:

- file: php-fastcgi-service

service.running:

- name: php-fpm

- enable: True

- require:

- cmd: php-fastcgi-service

- watch:

- file: php-ini

- file: php-fpm

php-info:

cmd.run:

- name: echo "<?php phpinfo(); ?>" >> /opt/nginx/html/phpinfo.php

- unless: test -f /opt/nginx/html/phpinfo.php

#测试执行

[root@saltstack-master php]# salt '\*' state.sls php.pkg-php-init env=prod

[root@saltstack-master php]# salt '\*' state.sls php.install test=True env=prod

Summary

-------------

Succeeded: 28

Failed: 0

-------------

Total states run: 28

#编写top file 指定minion

[root@saltstack-master php]# vim /srv/salt/base/top.sls

base:

'\*':

- init.env\_init

prod:

'\*':

- cluster.haproxy-service

- cluster.haproxy-service-keepalived

- nginx.service

- php.install

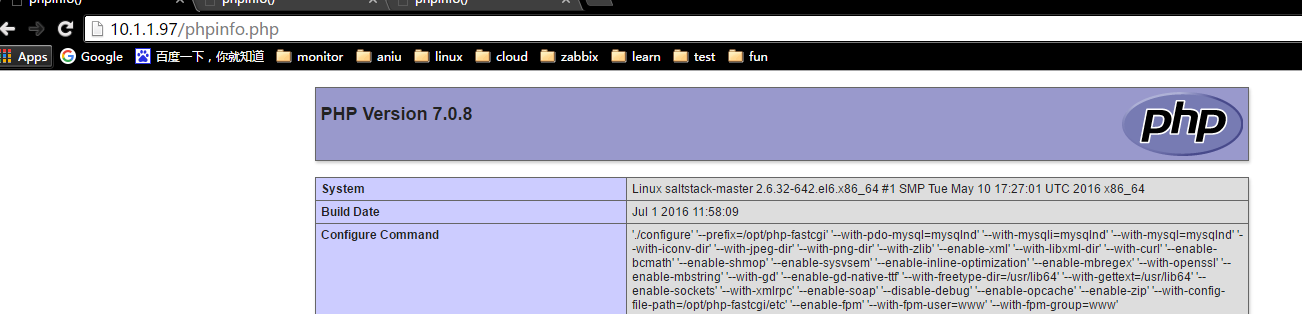
'saltstack-minion.example.com':

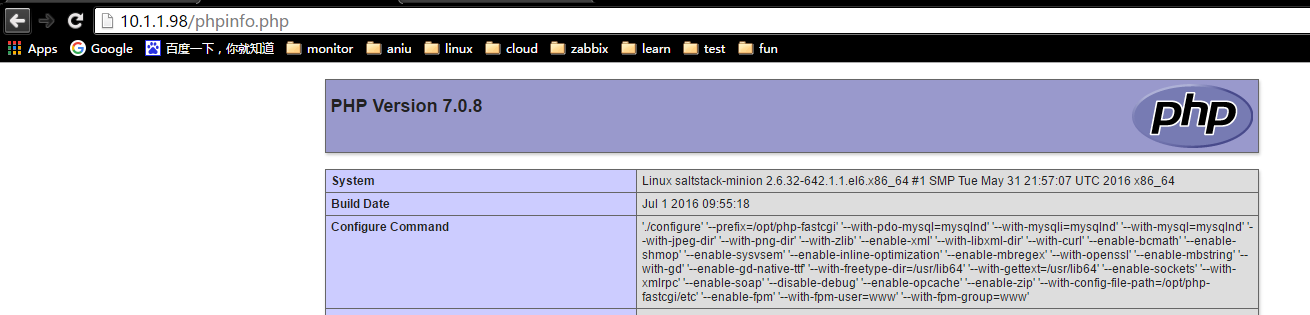
- memcached.service

#

[root@saltstack-master php]# salt '\*' state.highstate test=True

[root@saltstack-master php]# salt '\*' state.highstate





#

3.3.4 PHP Redis模块安装

[root@saltstack-master ~]# cd /srv/salt/prod/php/files/

[root@saltstack-master files]# wget <http://pecl.php.net/get/redis-3.0.0.tgz>

[root@saltstack-master files]# cd /srv/salt/prod/php/

[root@saltstack-master php]# vim php-redis.sls

include:

- php.install

redis-plugin:

file.managed:

- name: /usr/local/src/php-redis-3.0.0.tgz

- source: salt://php/files/redis-3.0.0.tgz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/local/src && tar zxf php-redis-3.0.0.tgz && cd php-redis-3.0.0 && /opt/php-fastcgi/bin/phpize && ./configure --with-php-config=/opt/php-fastcgi/bin/php-config && make && make install

- unless: test -f /opt/php-fastcgi/lib/php/extensions/\*/redis.so

- require:

- file: redis-plugin

enable-redis:

file.append:

- name: /opt/php-fastcgi/etc/php.ini

- text:

- extension=redis.so

#PHP Memcache 插件安装

#[root@saltstack-master files]# wget <http://pecl.php.net/get/memcache-3.0.8.tgz>

[root@saltstack-master ~]# cd /srv/salt/prod/php/files/

[root@saltstack-master php]# vim php-memcache.sls

include:

- php.install

memcached-plugin:

file.managed:

- name: /usr/local/src/php-memcached-3.0.8.tgz

- source: salt://php/files/memcached-3.0.8.tgz

- user: root

- group: root

- mode: 755

cmd.run:

- name: cd /usr/local/src && tar zxf php-memcached-3.0.8.tgz && cd php-memcached-3.0.8 && /opt/php-fastcgi/bin/phpize && ./configure --enable-memcache --with-php-config=/opt/php-fastcgi/bin/php-config && make && make install

- unless: test -f /opt/php-fastcgi/lib/php/extensions/\*/memcached.so

- require:

- file: memcached-plugin

enable-memcached:

file.append:

- name: /opt/php-fastcgi/etc/php.ini

- text:

- extension=memcached.so

3.5 业务模块

3.5.1 BBS论坛

使用Nginx+PHP（FastCGI）环境，使用Memcached作为缓存服务器，搭建简单的BBS论坛。

[root@saltstack-master ~]# mkdir -p /srv/salt/prod/web/files

[root@saltstack-master files]# vim bbs.conf

server {

listen 2000;

root /opt/nginx/html;

index index.htm index.html index.php;

location ~ \.php\$

{

fastcgi\_pass unix:/opt/php-fastcgi/php-fpm.sock;

fastcgi\_index index.php

include fastcgi.conf;

}

}

#编写bbs sls

[root@saltstack-master files]# cd ..

[root@saltstack-master web]# vim bbs.sls

include:

- php.install

- nginx.service

web-bbs:

file.managed:

- name: /opt/nginx/conf/vhost/bbs.conf

- source: salt://web/files/bbs.conf

- user: root

- group: root

- mode: 644

- require:

- service: php-fastcgi-service

- watch\_in:

- service: nginx-service

#修改top file，指定minion安装相应模块服务

[root@saltstack-master web]# vim /srv/salt/base/top.sls

base:

'\*':

- init.env\_init

prod:

'\*':

- cluster.haproxy-service

- cluster.haproxy-service-keepalived

- web.bbs

'saltstack-minion.example.com':

- memcached.service

#到此完成中小型web架构案例编写，查看目录结构。

[root@saltstack-master ~]# tree /srv/salt/base/

/srv/salt/base/

├── init

│?? ├── config

│?? │?? ├── foo.conf

│?? │?? ├── minion

│?? │?? ├── resolv.conf

│?? │?? ├── sshd\_config

│?? │?? ├── sysctl.conf

│?? │?? └── vimrc

│?? ├── cron.sls

│?? ├── del\_cron.sls

│?? ├── dns.sls

│?? ├── env\_init.sls

│?? ├── epel.sls

│?? ├── history.sls

│?? ├── log.sls

│?? ├── minion.sls

│?? ├── one.sls

│?? ├── ssh.sls

│?? ├── sysctl.sls

│?? ├── vim.sls

│?? └── yum.sls

├── \_returners

│?? ├── local\_return.py

│?? ├── mysql\_return.py

│?? └── select

└── top.sls

#prod

[root@saltstack-master ~]# tree /srv/salt/prod

/srv/salt/prod

├── cluster

│   ├── files

│   │   ├── haproxy-service.cfg

│   │   └── haproxy-service-keepalived.conf

│   ├── haproxy-service-keepalived.sls

│   └── haproxy-service.sls

├── haproxy

│   ├── files

│   │   ├── haproxy-1.6.5.tar.gz

│   │   └── haproxy.init

│   └── install.sls

├── keepalived

│   ├── files

│   │   ├── keepalived-1.2.22.tar.gz

│   │   ├── keepalived.init

│   │   └── keepalived.sysconfig

│   └── install.sls

├── libevent

│   ├── files

│   │   └── libevent-2.0.22-stable.tar.gz

│   └── install.sls

├── memcached

│   ├── files

│   │   └── memcached-1.4.27.tar.gz

│   ├── install.sls

│   └── service.sls

├── nginx

│   ├── files

│   │   ├── nginx-1.10.1.tar.gz

│   │   ├── nginx.conf

│   │   ├── nginx.conf\_bak

│   │   └── nginx-init

│   ├── install.sls

│   └── service.sls

├── pcre

│   ├── files

│   │   └── pcre-8.39.tar.gz

│   └── install.sls

├── php

│   ├── files

│   │   ├── init.d.php-fpm

│   │   ├── memcache-3.0.8.tgz

│   │   ├── php-7.0.8.tar.gz

│   │   ├── php-fpm.conf.default

│   │   ├── php.ini

│   │   ├── php.ini-production

│   │   ├── redis-3.0.0.tgz

│   │   └── www.conf.default

│   ├── install.sls

│   ├── php-memcache.sls

│   ├── php-redis.sls

│   └── pkg-php-init.sls

├── pkg

│   └── pkg-init.sls

├── user

│   └── www.sls

└── web

├── bbs.sls

└── files

└── bbs.conf

#测试执行

[root@saltstack-master ~]# salt '\*' state.highstate test=True

[root@saltstack-master ~]# salt '\*' state.highstate

Summary

-------------

Succeeded: 85 (changed=4)

Failed: 0

-------------

Total states run: 85

#先测试，在执行。

二、salt实现mysql准备及zabbix分布式监控

1．1、saltstack安装MySQL

环境准备：

|  |  |  |  |
| --- | --- | --- | --- |
| MySQL-master | 10.1.1.100 | CentOS 6.8 |  |
| Mysql-slave | 10.1.1.101 | CentOS 6.8 |  |
| Zabbix-Server | 10.1.1.103 | CentOS 6.8 |  |

2.1、软件包下载

wget -c http://liquidtelecom.dl.sourceforge.net/project/boost/boost/1.59.0/boost\_1\_59\_0.tar.gz -P /usr/local/src/

wget -c http://git.typecodes.com/libs/ccpp/cmake-3.2.1.tar.gz

wget -c [http://cdn.mysql.com//Downloads/MySQL-5.7/mysql-5.7.12.tar.gz -P /usr/local/src/](http://cdn.mysql.com//Downloads/MySQL-5.7/mysql-5.7.12.tar.gz%20-P%20/usr/local/src/)

#

root@saltstack-master[00:57:33]:~$mkdir -p /srv/salt/prod/mysql/files/

root@saltstack-master[00:58:20]:~$cd /srv/salt/prod/mysql/

root@saltstack-master[00:58:28]:/srv/salt/prod/mysql$vim pkg-install.sls

pkg-install:

pkg.installed:

- names:

- gcc

- gcc-c++

- autoconf

- automake

- zlib-devel

- ncurses

- ncurses-devel

- libtool-ltdl

- libtool-ltdl-devel

- libxml++

- libxml++-devel

- cmake

- bison

#安装boost

root@saltstack-master[01:02:26]:/srv/salt/prod/mysql$vim boost-init.sls

boost-init:

file.managed:

- name: /usr/local/src/boost\_1\_59\_0.tar.gz

- source: salt://mysql/files/boost\_1\_59\_0.tar.gz

cmd.run:

- name: cd /usr/local/src && tar zxf boost\_1\_59\_0.tar.gz && mkdir -p /data/mysql/data && mv boost\_1\_59\_0 /data/boost

- unless: test -d /data/mysql

- require:

- file: boost-init

#安装MySQL

root@saltstack-master[01:04:06]:/srv/salt/prod/mysql$vim install.sls

include:

- mysql.pkg-install

- mysql.boost-init

mysql-user:

user.present:

- name: mysql

group.present:

- name: mysql

mysql-init:

file.managed:

- name: /usr/local/src/mysql-5.7.12.tar.gz

- source: salt://mysql/files/mysql-5.7.12.tar.gz

cmd.run:

- name: cd /usr/local/src && tar zxf mysql-5.7.12.tar.gz && cd mysql-5.7.12 && cmake -DCMAKE\_INSTALL\_PREFIX=/data/mysql -DMYSQL\_DATADIR=/data/mysql/data -DSYSCONFDIR=/etc -D

- unless: test -d /data/mysql/bin

- require:

- file: mysql-init

mysql-conf:

file.managed:

- name: /etc/my.cnf

- source: salt://mysql/files/my.cnf

mysql-env:

file.append:

- name: /etc/profile

- text:

- export PATH=/data/mysql/bin:$PATH

cmd.run:

- name: chown -R mysql:mysql /data/mysql && chmod -R go-rwx /data/mysql/data && source /etc/profile

- require:

- file: mysql-init

#mysql-log:

# file.directory:

# - name: /var/log/mysql

# - user: mysql

# - group: mysql

# - mode: 755

mysql-daemon:

file.managed:

- name: /etc/init.d/mysqld

- source: salt://mysql/files/mysql.server

cmd.run:

- name: chkconfig mysqld on && chmod +x /etc/init.d/mysqld

- require:

- file: mysql-daemon

mysql-service:

cmd.run:

- name: /etc/init.d/mysqld start

- unless: ps -ef | grep mysqld | grep -v grep

service.running:

- name: mysqld

- enable: True

- require:

- file: mysql-init

#mysql-safe:

# cmd.run:

# - name: /data/mysql/bin/mysqld --initialize-insecure --user=mysql --basedir=/data/mysql --basedir=/data/mysql/data && mysqld\_safe --user=mysql --datadir=/data/mysql/data/ -

# initial mysql database

# mysql\_secure\_installation it's important

#files目录相关文件请准备好

root@saltstack-master[01:05:38]:/srv/salt/prod/mysql/files$ls

boost\_1\_59\_0.tar.gz cmake-3.2.1.tar.gz my.cnf mysql-5.7.12.tar.gz mysql.server url.txt

#详情参考github：<https://github.com/wh211212/ops-saltstack>

#测试

注意：建议测试的时候指定特定的env环境以及特定的sls文件，由于install.sls较多，建议注释全部，单个执行避免报错。

Mysql-master同步成功之后，同步mysql-slave，然后配置各自的my.cnf，实现mysql主从，参考我的博文Mysql主从同步实现，报错mysql5.6,5.7

博文链接：<http://blog.sina.com.cn/s/blog_87113ac20102w3x7.html>

#Saltstack结合zabbix实现自动监控服务器

设定zabbix的salt环境为dev

root@saltstack-master[01:11:09]:~$mkdir -p /srv/salt/dev/zabbix/files/

base:

- /srv/salt/base

# dev:

# - /srv/salt/dev/services

# - /srv/salt/dev/states

prod:

- /srv/salt/prod

# - /srv/salt/prod/states

dev:

- /srv/salt/dev #新增

#修改master配置文件之后重启salt-master服务

#files目录下文件准备

root@saltstack-master[01:12:45]:/srv/salt/dev/zabbix/files$ls

my.cnf services url.txt zabbix-3.0.3.tar.gz zabbix\_agentd zabbix\_agentd.conf zabbix.conf.php zabbix\_server zabbix\_server.conf

#Zabbix源码包下载

# wget <http://jaist.dl.sourceforge.net/project/zabbix/ZABBIX%20Latest%20Stable/3.0.3/zabbix-3.0.3.tar.gz>

#参考上面github链接

root@saltstack-master[01:14:01]:/srv/salt/dev/zabbix$cat pkg-init.sls

zabbix-pkg-init:

pkg.installed:

- names:

- net-snmp-devel

- curl

- libcurl-devel

- gcc-c++

# - mysql-devel

#saltstack同步zabbix-server的sls文件编写

include:

- zabbix.pkg-init

zabbix-user:

user.present:

- name: zabbix

- shell: /sbin/nologin

group.present:

- name: zabbix

zabbix-server-init:

file.managed:

- name: /usr/local/src/zabbix-3.0.3.tar.gz

- source: salt://zabbix/files/zabbix-3.0.3.tar.gz

- unless: test -f /usr/local/src/zabbix-3.0.3.tar.gz

cmd.run:

- name: cd /usr/local/src && tar zxf zabbix-3.0.3.tar.gz && cd zabbix-3.0.3 && ./configure --prefix=/opt/zabbix --enable-server --enable-agent --with-libcurl --with-mysql=/d

- unless: test -d /opt/zabbix

- require:

- file: zabbix-server-init

zabbix-server-conf:

file.managed:

- name: /opt/zabbix/etc/zabbix\_server.conf

- source: salt://zabbix/files/zabbix\_server.conf

zabbix-agentd-conf:

file.managed:

- name: /opt/zabbix/etc/zabbix\_agentd.conf

- source: salt://zabbix/files/zabbix\_agentd.conf

zabbix-db-set:

file.managed:

- name: /root/.my.cnf

- source: salt://zabbix/files/my.cnf

cmd.run:

- name: /data/mysql/bin/mysql -e "create database zabbix character set utf8 collate utf8\_bin;" && /data/mysql/bin/mysql -e "grant all privileges on zabbix.\* to zabbix@local

- unless: /data/mysql/bin/mysql -e "use zabbix;"

- require:

- file: zabbix-db-set

zabbix-sql-set:

cmd.run:

- name: cd /usr/local/src/zabbix-3.0.3/database/mysql && /data/mysql/bin/mysql -uzabbix -p@Zabbix..0 zabbix < schema.sql && /data/mysql/bin/mysql -uzabbix -p@Zabbix..0 zabbi

- unless: /data/mysql/bin/mysql -e "show create table zabbix.users"

zabbix-port-set:

file.managed:

- name: /etc/services

- source: salt://zabbix/files/services

zabbix-log-set:

file.directory:

- name: /var/log/zabbix

- unless: test -d /var/log/zabbix

cmd.run:

- name: chown -R zabbix:zabbix /var/log/zabbix

- unless: ls -l /var/log/zabbix | awk '/zabbix/{print $3"\t"$4}'

- require:

- file: zabbix-log-set

zabbix-server-daemon:

file.managed:

- name: /etc/init.d/zabbix\_server

- source: salt://zabbix/files/zabbix\_server

- user: root

- group: root

- mode: 755

cmd.run:

- name: chkconfig zabbix\_server on && ln -s /data/mysql/lib/libmysqlclient.so.20 /usr/lib64/ && /etc/init.d/zabbix\_server start

- unless: ps -ef | grep zabbix\_server | grep -v grep

- require:

- file: zabbix-server-daemon

zabbix-pid-set:

file.directory:

- name: /opt/zabbix/pid

- unless: test -d /opt/zabbix/pid

cmd.run:

- name: chown -R zabbix:zabbix /opt/zabbix/pid

- unless: ls -l /opt/zabbix/pid | awk '/zabbix/{print $3"\t"$4}'

- require:

- file: zabbix-pid-set

zabbix-agentd-daemon:

file.managed:

- name: /etc/init.d/zabbix\_agentd

- source: salt://zabbix/files/zabbix\_agentd

- user: root

- group: root

- mode: 755

cmd.run:

- name: chkconfig zabbix\_agentd on && /etc/init.d/zabbix\_agentd start

- unless: ps -ef | grep zabbix\_agentd | grep -v grep

- require:

- file: zabbix-agentd-daemon

zabbix-front-set:

cmd.run:

- name: cp -rf /usr/local/src/zabbix-3.0.3/frontends/php /opt/nginx/html/zabbix && chown -R www:www /opt/nginx/html/zabbix

- unless: test -d /opt/nginx/html/zabbix

zabbix-conf-php:

file.managed:

- name: /opt/nginx/html/zabbix/conf/zabbix.conf.php

- source: salt://zabbix/files/zabbix.conf.php

- user: zabbix

- group: zabbix

- mode: 644

#saltstack实现同步zabbix-agent编写sls文件

zabbix-agent-init:

pkg.installed:

- name: gcc-c++

user.present:

- name: zabbix

- shell: /sbin/nologin

group.present:

- name: zabbix

file.managed:

- name: /usr/local/src/zabbix-3.0.3.tar.gz

- source: salt://zabbix/files/zabbix-3.0.3.tar.gz

- unless: test -f /usr/local/src/zabbix-3.0.3.tar.gz

cmd.run:

- name: cd /usr/local/src && tar zxf zabbix-3.0.3.tar.gz && cd zabbix-3.0.3 && ./configure --prefix=/opt/zabbix --enable-agent && make && make install

- unless: test -d /opt/zabbix

- require:

- file: zabbix-agent-init

zabbix-agentd-conf:

file.managed:

- name: /opt/zabbix/etc/zabbix\_agentd.conf

- source: salt://zabbix/files/zabbix\_agentd.conf

- require:

- file: zabbix-agent-init

zabbix-port-set:

file.managed:

- name: /etc/services

- source: salt://zabbix/files/services

zabbix-log-set:

file.directory:

- name: /var/log/zabbix

- unless: test -d /var/log/zabbix

cmd.run:

- name: chown -R zabbix:zabbix /var/log/zabbix

- unless: ll /var/log/zabbix/ | grep zabbix\*

- require:

- file: zabbix-log-set

zabbix-pid-set:

file.directory:

- name: /opt/zabbix/pid

- unless: test -d /opt/zabbix/pid

cmd.run:

- name: chown -R zabbix:zabbix /opt/zabbix/pid

- unless: ll /opt/zabbix/pid | grep zabbix\*

- require:

- file: zabbix-pid-set

zabbix-agentd-daemon:

file.managed:

- name: /etc/init.d/zabbix\_agentd

- source: salt://zabbix/files/zabbix\_agentd

- user: root

- group: root

- mode: 755

cmd.run:

- name: chkconfig zabbix\_agentd on && /etc/init.d/zabbix\_agentd start

- unless: ps -ef | grep zabbix\_agentd | grep -v grep

- require:

- file: zabbix-agentd-daemon

#