SALTSTACK

初识SaltStack

---人生苦短，我用Salt

绿肥(PengYao) 2013-10

关于我

- 绿肥(@绿小小肥)
- Blog: http://pengyao.org/
- SaltStack爱好者,Zabbix爱好者
- 中国SaltStack用户组(China SaltStack User Group)发起人

关于CSSUG

- 中国SaltStack用户组(China SaltStack User Group, aka 'CSSUG')
- SaltStack官方认可，由广大中国SaltStack爱好者、使用者推动，为宣传和发展SaltStack而成立的用户组
- 目前CSSUG建立有Wiki知识库、QQ群、Google Groups及邮件列表用于线上交流，同时也组织线下交流活动
- Site: http://saltstack.cn/
- 当前主要交流SaltStack在DevOps及云环境下的使用经验，同时参与Salt手册中文版翻译工作.

SaltStack是什么?

关于SaltStack

- salt是一个新的基础平台管理工具。只需花费数分钟即可运行起来，扩展性足以支撑管理上万台服务器，数秒钟即可完成数据传递.
- 官网：http://saltstack.com/
- 项目地址：https://github.com/saltstack/salt
- 出生时间：2011年
- 开发语言：Python
- 工作方式：C/S结构(minion/master)，基于zeromq，采用长连接方式，支持Schedule，0.17版本增加了salt-ssh
- 三大功能
  - Remote Execution
  - Config Management
  - Cloud Management

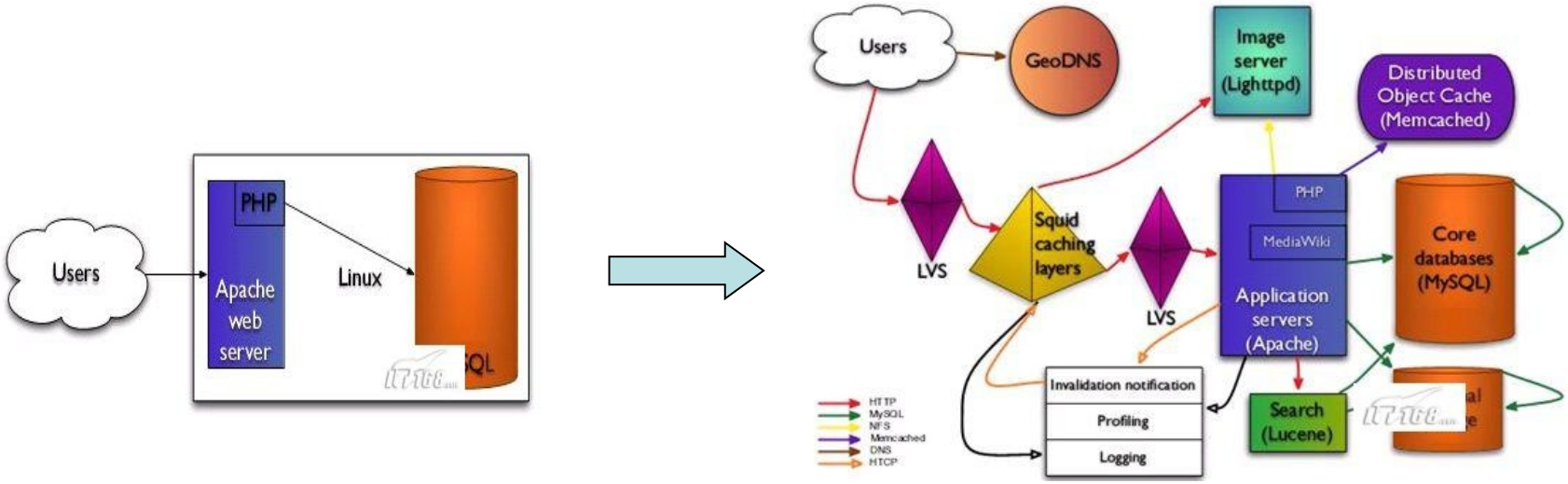
为什么选择SaltStack?

SALTSTACK

需要维护的服务器数量越来越庞大

需要维护的操作系统种类越来越多

需要维护的系统架构越来越复杂



注: 本页图片为wikipedia架构图,图片来源于IT168

服务等级被要求越来越高

Availability %	Downtime per year	Downtime per month*	Downtime per week
90% ("one nine")	36.5 days	72 hours	16.8 hours
99% ("two nines")	3.65 days	7.20 hours	1.68 hours
99.9% ("three nines")	8.76 hours	43.8 minutes	10.1 minutes
99.99% ("four nines")	52.56 minutes	4.32 minutes	1.01 minutes
99.999% ("five nines")	5.26 minutes	25.9 seconds	6.05 seconds
99.9999% ("six nines")	31.5 seconds	2.59 seconds	0.605 seconds
99.99999% ("seven nines")	3.15 seconds	0.259 seconds	0.0605 seconds

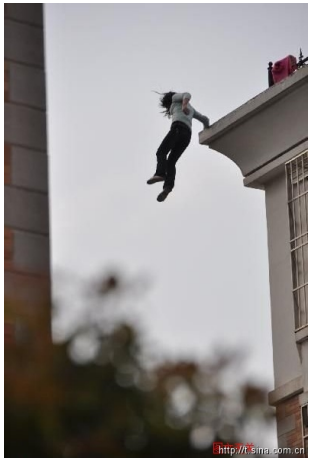
注: 本页数据来源于wikipedia

SALTSTACK

而运维团队规模却越来越小

樊睿：杭州某高新技术开发公司 一名运维，长期以来饱受公司加班的摧残，近段时间，女友应无法忍受他长期加班，遂与之分手。此男悲痛欲绝，伤心之余，格式化并关闭所有服务器之后跳楼自杀。

2分钟前(5评)



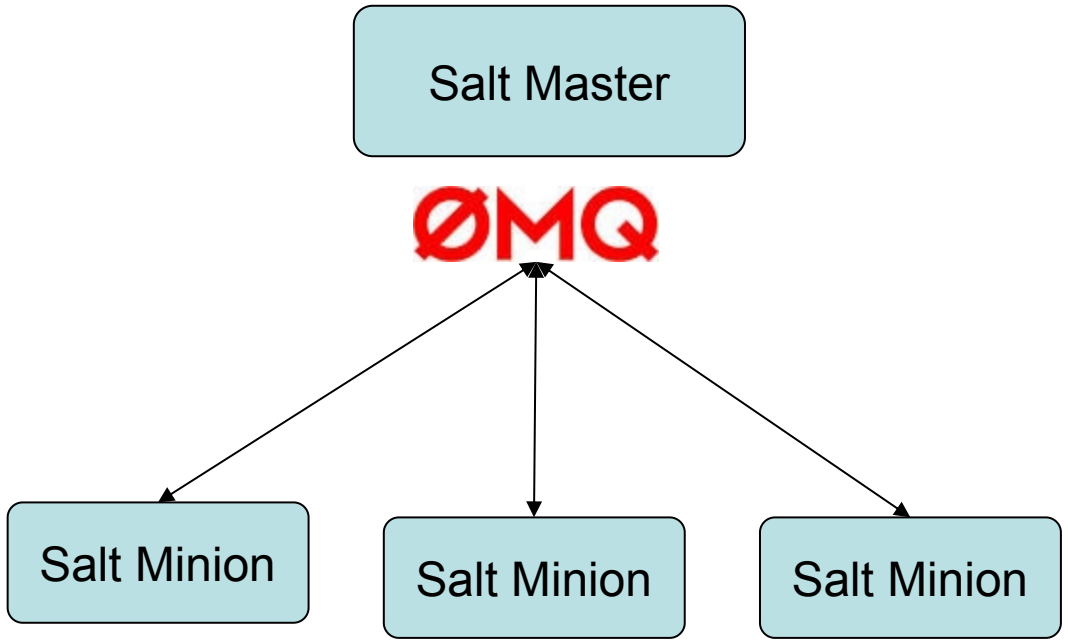
人生苦短

工欲善其事，必先利其器

VS

天下武功，唯快不破

- Salt底层基于ZeroMQ PUB/SUB系统，以广播的形式通知Minions
- 基于如此架构，不管你管控机器数量是个位数还是四位数，数秒钟即可完成消息传递
- 更多内容请访问：http://docs.saltstack.com/ref/topology.html



丰富的远程执行模块

- Salt封装了常用的命令执行模块，包括管理软件包、管理用户、传输文件、管理服务、web服务管理、数据库管理等等。
- 截止2013-10-14，Salt内置的执行模块已经达168种之多
- 还不满足需求？ 自定义模块也非常简单，欲知详情，请访问：http://docs.saltstack.com/ref/modules/index.html
- 独乐乐不如众乐乐，非常希望将代码贡献给Salt项目，开源传万世界，因有我参与.

配置管理，运维的一大步

- Salt从0.9.2开始，提供了配置管理的支持
- Salt封装了常用的状态管理模块，无论文件、软件包、服务、代码版本、数据库等状态
- 截止2013-10-14，Salt已经内置了77种状态管理模块
- 还不满足需求？自定义状态模块也非常简单。欲知详情，请访问：http://docs.saltstack.com/ref/states/writing.html
- 独乐乐不如众乐乐，非常希望将代码贡献给Salt项目，开源传万世界，因有我参与.

模块化设计

- Salt采用模块化设计，提供了丰富的子系统，常用的有：
  - grains：minion端配置信息
  - pillar：配置管理数据存储
  - modules：执行模块
  - states：配置管理模块
  - fileserver：Salt文件服务
  - log：Salt日志系统
  - returner：Report系统
  - runner：基于modules封装的更高级的执行方式
  - reactor：基于产生的事件进一步处理
  - templates：模板系统
  - wheel：管理Master

灵活的架构支持

- Salt Master/Minion: 常用的部署方式
- Salt SSH: 无需Minion，采用SSH进行管理 (0.17引入)
- Salt Syndic: 分布式架构支持
- Salt Multi-Master: 多Master架构(冗余)
- Salt API: REST API,更易于与第三方系统集成
- Salt Halite: Salt Web UI
- Salt Cloud: 公有/私有云管理

开源协议

- Salt采用Apache 2.0协议，Salt的所有项目代码100%开源

如何使用SaltStack?

安装配置SaltStack

注：本页内容基于CentOS 6

- 安装EPEL

```
# rpm -ivh http://mirrors.sohu.com/fedora-epel/6/x86_64/epel-release-6-8.noarch.rpm
```

- 安装salt-master（本操作在管理端进行）

```
# yum install salt-master
# service salt-master start
```

- 被管理端安装salt-minion （本操作在被管理端进行）

```
# yum install salt-minion
# sed -i 's/#master: salt/master: matster_ip/' /etc/salt/minion
# service salt-minion start
```

- 接受minion（本操作在管理端进行）

```
# salt-key -A
```

- ping测试 （本操作在管理端进行）

```
# salt '*' test.ping
```

更多平台的安装请访问：http://docs.saltstack.com/topics/installation/index.html

Salt远程执行

- 远程执行

```
# salt 'test-01' cmd.run 'uptime'
```

其中test-01为需要执行本操作的主机，cmd为命令模块,run为方法，uptime为cmd.run方法的参数，本例为查看test-01主机的已运行时间及load avage

更多Salt模块请访问:

http://docs.saltstack.com/ref/modules/all/index.html

自定义模块方法:

http://docs.saltstack.com/ref/modules/index.html

Salt配置管理

- state文件默认目录：/srv/salt/ （对应master配置文件/etc/salt/master中file_rootfs）
- 配置入口文件：/srv/salt/top.sls
- 设计原则：KISS（Keep It Stupidly Simple）
- 支持多种解析器，如jinja、mako、python、yaml等
- 执行方法：

  //执行test-01主机所有的状态配置（需要在top.sls中指定）

  # salt 'test-01' state.highstate

  //执行test-01主机mysql状态配置

  # salt 'test-01' state.sls mysql

Salt配置管理实例

- 例子为MySQL配置管理，涉及安装MySQL Server，推送配置，MySQL服务状态管理，MySQL db状态管理，MySQL 用户状态管理，MySQL 用户权限管理
- 以CentOS 6为应用平台
- 本配置管理例子源码已经存放在 https://github.com/pengyao/salt-mysql
- 前提：
  - 已部署Salt Master/Minion
  - Minion上已经配置了EPEL仓库
  - MySQL服务的配置已经存放在Master的/srv/salt/mysql/files/my.cnf

MySQL Server软件包安装、配置、服务状态及增加MySQL管理

```
# cat /srv/salt/mysql/server/init.sls
mysql-server:
  pkg.installed:
    - name: mysql-server
  file.managed:
    - name: /etc/my.cnf
    - source: salt://mysql/files/my.cnf
    - require:
      - pkg: mysql-server
  service.running:
    - name: mysqld
    - enable: True
    - watch:
      - pkg: mysql-server
      - file: mysql-server
{% if salt['config.get']('mysql.pass') %}
## support mysql manage
mysqld-manager:
  pkg.installed:
    - name: MySQL-python
    - require:
      - service: mysql-server
  module.wait:
    - name: saltutil.refresh_modules
    - watch:
      - pkg: mysqld-manager
    - order: 1
{% endif %}
```

MySQL Secure Install

$ cat /srv/salt/mysql/server/secure_install.sls

```
include:
  - mysql.server
{% if salt['config.get']('mysql.pass') %}
{% set mysql_root_pass = salt['config.get']('mysql.pass') %}
## mysql secure installation
mysqld_secure:
  file.managed:
    - name: /usr/bin/mysql_secure.sh
    - source: salt://mysql/files/mysql_secure.sh
    - user: root
    - group: root
    - mode: 700
    - require:
      - pkg: mysql-server
  cmd.wait:
    - name: /usr/bin/mysql_secure.sh {{mysql_root_pass}}
    - require:
      - service: mysql-server
    - watch:
      - file: mysqld_secure
    - require_in:
      - module: mysqld_manager
{% endif %}
```

MySQL DBs、Users、Permissions状态管理

```
$ cat /srv/salt/mysql/server/manage.sls
include:
  - mysql.server.secure_install

{% if salt['config.get']('mysql.pass') %}
{% if 'mysql-server' in pillar %}
## mysql database states
{% if 'databases' in pillar['mysql-server'] %}
{% for eachdb in pillar['mysql-server']['databases'] %}
mysql_database_{{eachdb}}:
  mysql_database.present:
    - name: {{eachdb}}
    - require:
      - module: mysqld-manager
{% endfor %}
{% endif %}

## mysql user states
{% if 'users' in pillar['mysql-server'] %}
{% for eachuser in pillar['mysql-server']['users'] %}
{% set username = eachuser['user'] %}
mysql_users_{{username}}:
  mysql_user.present:
    - name: {{username}}
    - host: {{eachuser['host']}}
    - password: {{eachuser['password']}}
    - require:
      - module: mysqld-manager
## mysql user permission
{% if 'permissions' in eachuser%}
{% for eachgrant in eachuser['permissions'] %}
mysql_grants_{{username}}_{{eachgrant['database']}}:
  mysql_grants.present:
    - grant: {{eachgrant['grant']}}
    - database: {{eachgrant['database']}}
    - user: {{username}}
    - host: {{eachuser['host']}}
{% endfor %}
{% endif %}
{% endfor %}
{% endif %}
{% endif %}
```

Pillar

```
$ cat /srv/pillar/mysql/server/init.sls
mysql.server: 'localhost'
mysql.port: 3306
mysql.user: 'root'
mysql.pass: 'rootpass'
mysql.db: 'mysql'
mysql.unix_socket: '/var/lib/mysql/mysql.sock'
```

```
$ cat pillar/mysql/server/manage.sls
mysql-server:
  databases:
    - pengyao
    - saltstack
  users:
    - user: pengyao
      password: pengyaopass
      host: localhost
      permissions:
        - grant: select,insert,update
          database: pengyao.*
        - grant: all
          database: saltstack.*
```

top.sls及应用状态

- top.sls

```
$ cat /srv/salt/top.sls
base:
  '*':
    - mysql.server.manage

$ cat /srv/pillar/top.sls
base:
  '*':
    - mysql.server
    - mysql.server.manage
```

- 应用状态

```
#salt '*' state.highstate
```

- Minion将自动检查本机MySQL Server是否安装、是否已经应用最新配置、服务是否已经为启动状态、是否已经创建了pengyao及saltstack数据库，是否已经存在数据库pengyao用户，对应的用户权限是否正确
- 如果新增数据库，新增用户及权限只需要修改/srv/pillar/mysql/server/manage.sls,重新应用状态即可

人生苦短，我用Salt

欢迎拥抱SaltStack