linora(Yu Zhang)

Droi | 上海市徐汇区普天信息软件园

StackStorm

官方文档 v1.0

目录

[一. StackStorm 概览 2](#_Toc504143943)

[1.1 简介 2](#_Toc504143944)

[1.2 工作原理 2](#_Toc504143945)

[二. 安装 4](#_Toc504143946)

[2.1 RHEL/CentOS 7 4](#_Toc504143947)

[2.1.1 安装步骤概览 5](#_Toc504143948)

[2.1.2 系统需求 5](#_Toc504143949)

[2.1.3 最小化安装 6](#_Toc504143950)

[2.1.4 认证配置 11](#_Toc504143951)

[2.1.5 安装WebUI并设置SSL终点 12](#_Toc504143952)

[2.1.6 设置ChatOPS 13](#_Toc504143953)

[2.1.7 安全提示 14](#_Toc504143954)

[2.1.8 升级为Brocade Workflow Composer 15](#_Toc504143955)

[三. 快速入门 16](#_Toc504143956)

[3.1 开始使用Stackstorm with CLI 16](#_Toc504143957)

[3.2 认证 17](#_Toc504143958)

[3.3 使用Actions 17](#_Toc504143959)

[3.4 定义rule 18](#_Toc504143960)

[3.5 部署rule 19](#_Toc504143961)

[3.6 部署Examples 20](#_Toc504143962)

[3.7 Datastore 20](#_Toc504143963)

[四. 定制架构 21](#_Toc504143964)

[4.1 说明 21](#_Toc504143965)

[4.2 定制过程 21](#_Toc504143966)

[4.3 验证 21](#_Toc504143967)

[五．salt with st2组合开发实例 23](#_Toc504143968)

# StackStorm 概览

## 1.1 简介

StackStorm 是一个为服务及工具集提供集成及自动化的平台。它可以把你已有的基础设施及应用环境打包，使你可以更轻松的实现自动化任务。功能聚焦在基于event触发的action（事件驱动）。

StackStorm 助力你完成一些常用自动化操作，示例如下：

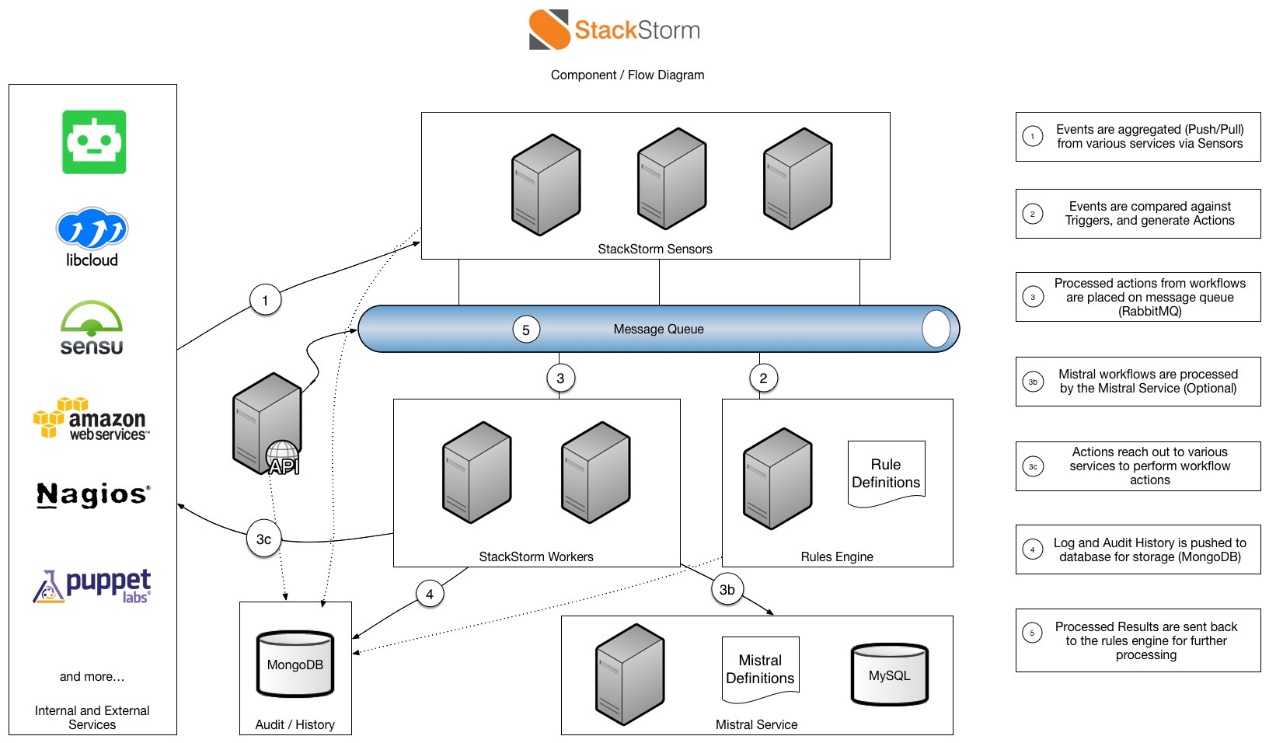
**故障排除：** Nagios、Sensu、New Relic及其他监控系统可触发StackStorm完成一系列的故障检查工作，可将检查结果post到类似HipChat或JIRA的平台中。

**自动修复：**识别或验证硬件失败，合理的offload故障节点并邮件通知管理人员大概的Downtime，如果中途出现任何非预期错误，立刻冻结workflow并调用PagerDuty请求人工介入。

**持续部署：**使用Jenkins进行构建测试，预备一个新的AWS cluster，使用Load balance，支持前滚 & 回滚，基于NewRelic app性能数据。

StackStorm帮助你组合以上甚至更多操作落地为rule、workflow或action。Rule及Workflow以code的形式存储于StormStack中，是StormStack的核心内容，这些内容可被后期重用，也可以把内容分享与开源社区（[StackStorm community](https://www.stackstorm.com/community/)）。

## 1.2 工作原理



StackStorm支持可插拔组件，这些组件分为两大类：sensor及action组件类。

**Sensors：**Python插件，当外部系统出现一个event，会被sensor处理，接着触发trigger。

**Actions：**使StackStorm向外集成。有一般action（如：ssh，REST call），集成Action（如：OpenStack、Docker及Puppet），定制Action。Action可以是Python Plugins或者任何脚本，只要添加几行元数据即可添加到StackStorm中使用。Action可直接在CLI或者API中调用，也可以在rule或者workflow中被调用。

**Triggers：**是StackStorm对外部事件的一个表示。有一般trigger及集成trigger（如：Sensu 报警, JIRA 问题更新）。可以自己实现sensor plugin以扩展sensor 类型。

**Rules：**映射trigger到对应action（或者workflow），应用匹配规范，也会映射trigger payload到action的输入。

**Workflows：**将action组合在一起，定义执行顺序、转换条件及数据传递。大部分的自动化任务不止一个action动作。workflow是原子性操作，可以被手动调用，也可被在rule中被trigger触发。

**Packs：**是内容部署的单位。将集成 （triggers and actions）及自动化任务 (rules and workflows)分组以便于管理及分享。在[StackStorm Exchange](https://exchange.stackstorm.org/)你将找到大量的共享packs。

**Action执行审计：**记录保存详细的触发器上下文及执行结果，日志信息可与外部系统集成，例如常用的日志分析工具：LogStash，Splunk，statsd，syslog。

StackStorm为模块化架构。它通过通信消息总线由松散成对的服务组件构成，支持水平扩展。StackStorm具有WebUI、CLI Client及完整的REST API。同时支持Python client。

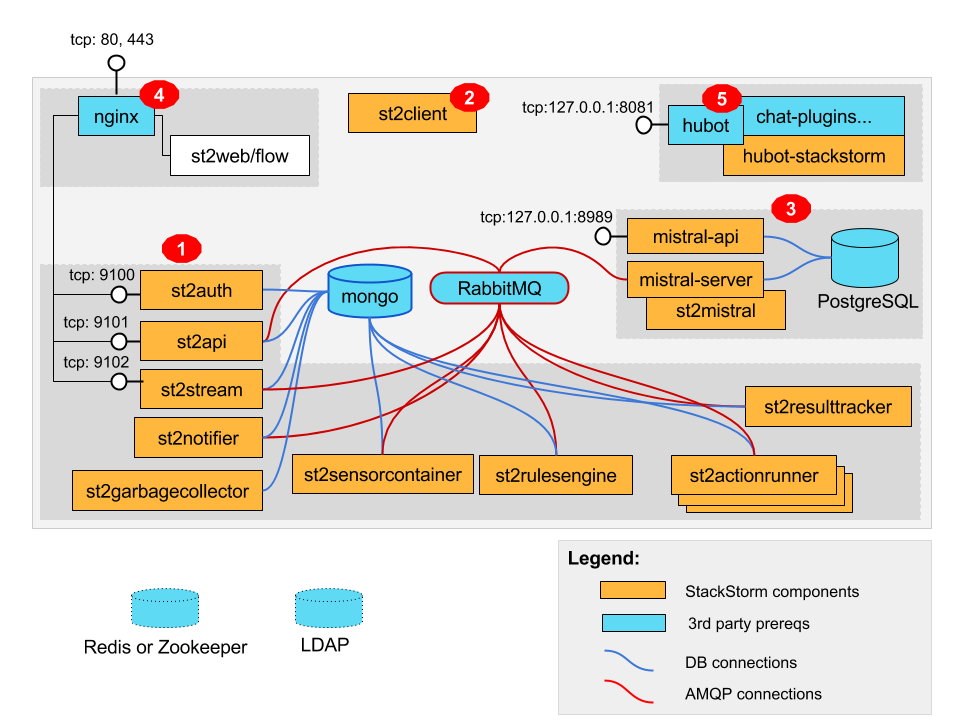
# 安装

快速安装参考：<https://docs.stackstorm.com/install/index.html>

## 2.1 RHEL/CentOS 7

参考：<https://docs.stackstorm.com/install/rhel7.html>

如果你想进行一键安装，请参考一级目录中的【快速安装参考】url中相关内容。本节内容为RHEL/CentOS 7下的定制安装详细过程，安装的相关组件参照下图：



**图片来源：**<https://docs.stackstorm.com/install/overview.html>

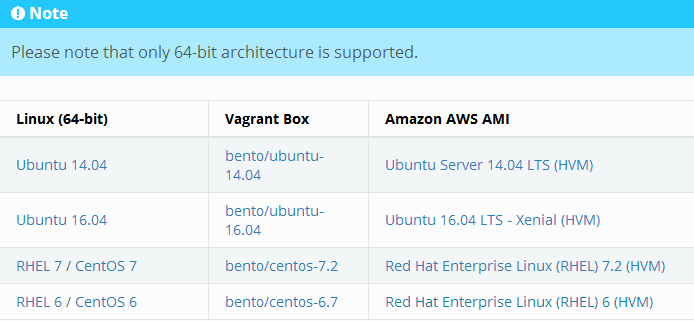
**说明：**StackStorm努力保持文档为最新版本，但是要想了解最新版本中修改了哪些内容，还是强烈建议直接查看安装脚本中的内容：<https://github.com/StackStorm/st2-packages/tree/master/scripts>。

### 2.1.1 安装步骤概览

* [系统需求](https://docs.stackstorm.com/install/rhel7.html#system-requirements)
* [最小化安装](https://docs.stackstorm.com/install/rhel7.html#minimal-installation)
  + [调整Selinux策略](https://docs.stackstorm.com/install/rhel7.html#adjust-selinux-policies)
  + [安装依赖包](https://docs.stackstorm.com/install/rhel7.html#install-dependencies)
  + [设置Repositories](https://docs.stackstorm.com/install/rhel7.html#setup-repositories)
  + [安装StackStorm组件](https://docs.stackstorm.com/install/rhel7.html#install-st2-components)
  + [设置Datastore加密](https://docs.stackstorm.com/install/rhel7.html#setup-datastore-encryption)
  + [设置Mistral数据库](https://docs.stackstorm.com/install/rhel7.html#setup-mistral-database)
  + [SSH及sudo配置](https://docs.stackstorm.com/install/rhel7.html#configure-ssh-and-sudo)
  + [启动服务](https://docs.stackstorm.com/install/rhel7.html#start-services)
  + 校验
* [认证配置](https://docs.stackstorm.com/install/rhel7.html#configure-authentication)
* [安装WebUI 及设置 SSL Termination](https://docs.stackstorm.com/install/rhel7.html#install-webui-and-setup-ssl-termination)
* [设置ChatOps](https://docs.stackstorm.com/install/rhel7.html#setup-chatops)
* [安全提示](https://docs.stackstorm.com/install/rhel7.html#a-note-on-security)
* [升级为Brocade Workflow Composer](https://docs.stackstorm.com/install/rhel7.html#upgrade-to-bwc)（商业版-收费）

### 2.1.2 系统需求

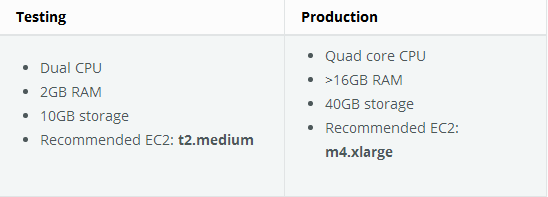
1. **支持的操作系统：**



1. **包依赖：**

操作系统可以选择最小化安装方式，并勾选OpenSSH Server，其他依赖包StackStorm将自动安装。

1. **硬件需求（最低）：**



1. **目录存储空间：**

确保 /var 及 /opt分别有1GB的可用空间，否则RabbitMQ及MongoDB可能不会正常运行。

1. **端口：**

* nginx (80, 443)
* mongodb (27017)
* rabbitmq (4369, 5672, 25672)
* postgresql (5432)
* st2auth (9100)
* st2api (9101)
* st2stream (9102)

### 2.1.3 最小化安装

**1. Selinux策略调整（Enforce mode下）**

查看目录策略：

# getenforce

如果命令返回为Enforcing，接着执行如下命令：

# SELINUX management tools, not available for some minimal installations

sudo yum install -y policycoreutils-python

# Allow network access for nginx

sudo setsebool -P httpd\_can\_network\_connect 1

# Allow RabbitMQ to use port '25672', otherwise it will fail to start

sudo semanage port --list | grep -q 25672 || sudo semanage port -a -t amqp\_port\_t -p tcp 25672

**2. 安装依赖包**

提示：StackStorm v2.2.0及以上版本，应该优先选择MongoDB 3.4。

安装MongoDB、RabbitMQ及PostgreSQL：

sudo yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

*# Add key and repo for the latest stable MongoDB (3.4)*

sudo rpm --import https://www.mongodb.org/static/pgp/server-3.4.asc

sudo sh -c ‘cat <<EOT > /etc/yum.repos.d/mongodb-org-3.4.repo

[mongodb-org-3.4]

name=MongoDB Repository

baseurl=https://repo.mongodb.org/yum/redhat/\$releasever/mongodb-org/3.4/x86\_64/

gpgcheck=1

enabled=1

gpgkey=https://www.mongodb.org/static/pgp/server-3.4.asc

EOT’

sudo yum -y install mongodb-org

sudo yum -y install rabbitmq-server

sudo systemctl start mongod rabbitmq-server

sudo systemctl enable mongod rabbitmq-server

*# Install and configure postgres*

sudo yum -y install postgresql-server postgresql-contrib postgresql-devel

*# Initialize PostgreSQL*

sudo postgresql-setup initdb

*# Make localhost connections to use an MD5-encrypted password for authentication*

sudo sed -i "s/\(host.\*all.\*all.\*127.0.0.1\/32.\*\)ident/\1md5/" /var/lib/pgsql/data/pg\_hba.conf

sudo sed -i "s/\(host.\*all.\*all.\*::1\/128.\*\)ident/\1md5/" /var/lib/pgsql/data/pg\_hba.conf

*# Start PostgreSQL service*

sudo systemctl start postgresql

sudo systemctl enable postgresql

**3. 设置repository**

执行如下脚本，将检查当前平台的架构并设置合适的StackStorm repository：

curl -s https://packagecloud.io/install/repositories/StackStorm/stable/script.rpm.sh | sudo bash

**4. 安装StackStorm组件**

# sudo yum install -y st2 st2mistral

假如你的RabbitMQ，MongoDB 或 PostgreSQL没有运行在本地系统，请调整以下设置：

* **RabbitMQ连接配置：**/etc/st2/st2.conf及/etc/mistral/mistral.conf
* **MongoDB连接配置：**/etc/st2/st2.conf
* **PostgreSQL连接配置：**/etc/mistral/mistral.conf

配置方式参考：https://docs.stackstorm.com/install/config/config.html

**5. 设置Datastore加密**

内置KV store允许用户创建加密后的值。使用对称加密法（AES256）。如果需要生成加密key，执行如下命令：

DATASTORE\_ENCRYPTION\_KEYS\_DIRECTORY**=**"/etc/st2/keys"

DATASTORE\_ENCRYPTION\_KEY\_PATH**=**"${DATASTORE\_ENCRYPTION\_KEYS\_DIRECTORY}/datastore\_key.json"

sudo mkdir -p ${DATASTORE\_ENCRYPTION\_KEYS\_DIRECTORY}

sudo st2-generate-symmetric-crypto-key --key-path ${DATASTORE\_ENCRYPTION\_KEY\_PATH}

*# Make sure only st2 user can read the file*

sudo chgrp st2 ${DATASTORE\_ENCRYPTION\_KEYS\_DIRECTORY}

sudo chmod o-r ${DATASTORE\_ENCRYPTION\_KEYS\_DIRECTORY}

sudo chgrp st2 ${DATASTORE\_ENCRYPTION\_KEY\_PATH}

sudo chmod o-r ${DATASTORE\_ENCRYPTION\_KEY\_PATH}

*# set path to the key file in the config*

sudo crudini --set /etc/st2/st2.conf keyvalue encryption\_key\_path ${DATASTORE\_ENCRYPTION\_KEY\_PATH}

sudo st2ctl restart-component st2api

1. **设置Mistral**  **Database**

运行如下命令设置Mistral Database：

*# Create Mistral DB in PostgreSQL*

cat << EHD | sudo -u postgres psql

CREATE ROLE mistral WITH CREATEDB LOGIN ENCRYPTED PASSWORD 'StackStorm';

CREATE DATABASE mistral OWNER mistral;

EHD

*# Setup Mistral DB tables, etc.*

/opt/stackstorm/mistral/bin/mistral-db-manage --config-file /etc/mistral/mistral.conf upgrade head

*# Register mistral actions*

/opt/stackstorm/mistral/bin/mistral-db-manage --config-file /etc/mistral/mistral.conf populate

1. **配置SSH及SUDO**

为了运行本地或远程shell actions，StackStorm用一个特殊系统用户（一般是stanley）。对于远程linux actions，SSH被采用。建议配置基于pulic key的ssh访问远程主机。建设本机也采用相同的方式。

**创建StackStorm系统用户，启用无密码sudo，设置本地ssh。执行如下命令：**

*# Create an SSH system user (default `stanley` user may already exist)*

sudo useradd stanley

sudo mkdir -p /home/stanley/.ssh

sudo chmod 0700 /home/stanley/.ssh

*# Generate ssh keys*

sudo ssh-keygen -f /home/stanley/.ssh/stanley\_rsa -P ""

*# Authorize key-based access*

sudo sh -c 'cat /home/stanley/.ssh/stanley\_rsa.pub >> /home/stanley/.ssh/authorized\_keys'

sudo chown -R stanley:stanley /home/stanley/.ssh

*# Enable passwordless sudo*

sudo sh -c 'echo "stanley ALL=(ALL) NOPASSWD: SETENV: ALL" >> /etc/sudoers.d/st2'

sudo chmod 0440 /etc/sudoers.d/st2

*# Make sure `Defaults requiretty` is disabled in `/etc/sudoers`*

sudo sed -i -r "s/^Defaults\s+\+?requiretty/# Defaults +requiretty/g" /etc/sudoers

**在远程目标主机配置基于public key的SSH访问：**

远程可参照上一步骤，详细过程略。

**如果不使用默认系统及路径，需要修改/etc/st2/st2.conf文件中的对应配置：**

[system\_user]

user = stanley

ssh\_key\_file = /home/stanley/.ssh/stanley\_rsa

1. **启动服务**

* Start services:

*sudo st2ctl start*

* Register sensors, rules and actions:

*sudo st2ctl reload*

1. **验证**

运行如下脚本，并确保全部成功：

st2 --version

st2 -h

*# List the actions from a 'core' pack*

st2 action list --pack**=**core

*# Run a local shell command*

st2 run core.local -- date -R

*# See the execution results*

st2 execution list

*# Fire a remote comand via SSH (Requires passwordless SSH)*

st2 run core.remote hosts**=**'localhost' -- uname -a

*# Install a pack*

st2 pack install st2

使用如下管理脚本管理服务：

sudo st2ctl start|stop|status|restart|restart-component|reload|clean

到此为止，你已经成功完成StackStorm的最小化安装，并可以愉快的玩耍了。

但是，你还可以选择安装WebUI，还可以通过SSL及认证功能加固安全体系，实现ChatOPS。

### 2.1.4 认证配置

本节示例为基于文件的简单认证配置过程，详细的认证配置请参考：[Authentication](https://docs.stackstorm.com/authentication.html) ，如PAM及LDAP认证。

**1. 创建http用户并设置password**

# Install htpasswd utility if you don't have it

sudo yum -y install httpd-tools

# Create a user record in a password file.

echo 'Ch@ngeMe' | sudo htpasswd -i /etc/st2/htpasswd st2admin

**2. 在**/etc/st2/st2.conf**文****件中配置及启用认证**

[auth]

# ...

enable = True

backend = flat\_file

backend\_kwargs = {"file\_path": "/etc/st2/htpasswd"}

# ...

**3. 重启st2api服务**

sudo st2ctl restart-component st2api

**4. 验证**

# Get an auth token to use in CLI or API

st2 auth st2admin

# A shortcut to authenticate and export the token

export ST2\_AUTH\_TOKEN=$(st2 auth st2admin -p 'Ch@ngeMe' -t)

# Check that it works

st2 action list

### 2.1.5 安装WebUI并设置SSL终点

Nginx被为WebUI提供静态文件服务、http到https的重定向、SSL终点服务及【st2auth 和st2api】 API endpoints的反向代理。安装st2web及nginx包可启用，在/etc/ssl/st2中生成新的证书或者放置已存在的证书，配置nginx with StackStorm’s以支持【[site config file st2.conf](https://github.com/StackStorm/st2/tree/master/conf/nginx/st2.conf)】。

Nginx版本必须是1.7.5或以上版本，REDHAT的官方repository中版本较老，按照如下命令安装适合的版本：

*# Add key and repo for the latest stable nginx*

sudo rpm --import http://nginx.org/keys/nginx\_signing.key

sudo sh -c ‘cat <<EOT > /etc/yum.repos.d/nginx.repo

[nginx]

name=nginx repo

baseurl=http://nginx.org/packages/rhel/\$releasever/x86\_64/

gpgcheck=1

enabled=1

EOT’

*# Install st2web and nginx*

sudo yum install -y st2web nginx

*# Generate a self-signed certificate or place your existing certificate under /etc/ssl/st2*

sudo mkdir -p /etc/ssl/st2

sudo openssl req -x509 -newkey rsa:2048 -keyout /etc/ssl/st2/st2.key -out /etc/ssl/st2/st2.crt \

-days 365 -nodes -subj "/C=US/ST=California/L=Palo Alto/O=StackStorm/OU=Information \

Technology/CN=**$(**hostname**)**"

*# Copy and enable the supplied nginx config file*

sudo cp /usr/share/doc/st2/conf/nginx/st2.conf /etc/nginx/conf.d/

*# Disable default\_server configuration in existing /etc/nginx/nginx.conf*

sudo sed -i 's/default\_server//g' /etc/nginx/nginx.conf

sudo systemctl restart nginx

sudo systemctl enable nginx

如果修改了Nginx的默认配置及路径，需要在st2web对应配置文件中做对应修改：/opt/stackstorm/static/webui/config.js。

访问https://${ST2\_HOSTNAME}即可进入WebUI。

如果开启了防火墙，请使用如下命令开放访问：

firewall-cmd --zone=public --add-service=http --add-service=https

firewall-cmd --zone=public --permanent --add-service=http --add-service=https

外部API访问URL示例： https://${EXTERNAL\_IP}/api/v1/${REST\_ENDPOINT}，curl示例：

curl -X GET -H 'Connection: keep-alive' -H 'User-Agent: manual/curl' -H 'Accept-Encoding: gzip, deflate' -H 'Accept: \*/\*' -H 'X-Auth-Token: <YOUR\_TOKEN>' https://1.2.3.4/api/v1/actions

同样，可以访问auth REST endpoints通过如下url：

https://${EXTERNAL\_IP}/auth/v1/${AUTH\_ENDPOINT}

st2命令行接口支持debug模式，示例如下：

st2 --debug action list

### 2.1.6 设置ChatOPS

如果你已经在运行一个Hubot实例，你可以安装[hubot-stackstorm plugin](https://github.com/StackStorm/hubot-stackstorm)并配置如下对应的StackStorm环境变量。否则，启用StackStorm ChatOPS最简单的方式是安装[st2chatops](https://github.com/stackstorm/st2chatops/)包。

**1. 验证chatops是否已经安装，对应通知rule是否启用**

# Ensure chatops pack is in place

ls /opt/stackstorm/packs/chatops

# Create notification rule if not yet enabled

st2 rule get chatops.notify || st2 rule create /opt/stackstorm/packs/chatops/rules/notify\_hubot.yaml

**2. 添加NodeJS v6 repository**

curl -sL https://rpm.nodesource.com/setup\_6.x | sudo -E bash -

**3. 安装st2chatops包**

sudo yum install -y st2chatops

**4. 配置/opt/stackstorm/chatops/st2chatops.env**

配置文件指向StackStorm及在使用的Chat Service。最小化需求是：生成一个[API key](https://docs.stackstorm.com/authentication.html#authentication-apikeys)，并把它设置为变量ST2\_API\_KEY的值。默认st2api 和st2auth 部署在一台主机上。如果不是这种部署方式，还需要设置ST2\_API及ST2\_AUTH\_URL变量，并将ST2\_HOSTNAME指向正确的主机。

以下的示例是使用Slack做为Chat目标。先在Slack web admin页面创建一个Bot，复制认证token到HUBOT\_SLACK\_TOKEN变量。

假如你使用一个不同的Chat Service，需设置正确的环境变量在 Chat service adapter settings 部分 in st2chatops.env：

[Slack](https://github.com/slackhq/hubot-slack), [HipChat](https://github.com/hipchat/hubot-hipchat),[Yammer](https://github.com/athieriot/hubot-yammer), [Flowdock](https://github.com/flowdock/hubot-flowdock), [IRC](https://github.com/nandub/hubot-irc) , [XMPP](https://github.com/markstory/hubot-xmpp)。

**5. 启动服务**

sudo systemctl start st2chatops

# Start st2chatops on boot

sudo systemctl enable st2chatops

**6. Reload st2 packs以使用chatops.notify rule被注册**

sudo st2ctl reload --register-all

**7. 使用ChatOPS**

参考：[ChatOps](https://docs.stackstorm.com/chatops/index.html)。

### 2.1.7 安全提示

默认情况下RabbitMQ，MongoDB及PostgreSQL初始化后都为免密码登录方式，基于安全性考虑，还是建议设置复杂度较高的密码。

提示：如果使用安装脚本，那么这些设置由脚本自动完成。

设置这些模块的密码不在本文档范围内，建议参考如下官方文档进行设置：

* MongoDB - <https://docs.mongodb.com/manual/tutorial/enable-authentication/>, <https://docs.mongodb.com/manual/core/authorization/>
* RabbitMQ - <https://www.rabbitmq.com/authentication.html>
* PostgreSQL - <https://www.postgresql.org/docs/9.4/static/auth-methods.html>

设置完对应模块的密码后，对应StackStorm配置文件也要做相应变更：

1. **StackStorm - /etc/st2/st2.conf**

* database.username - MongoDB database username.
* database.password - MongoDB database password.
* messaging.url - RabbitMQ transport url (amqp://<username>:<password>@<hostname>:5672)

1. **Mistral - /etc/mistral/mistral.conf**

* database.connection - PostgreSQL database connection string (postgresql+psycopg2://<username>:<password>@<hostname>/mistral)
* transport\_url - RabbitMQ transport url (rabbit://<username>:<password>@<hostname>:5672)

网络安全最佳实践建议：

* 通信加密，通过启用SSL/TLS for MongoDB、RabbitMQ及PostgreSQL。
* 配置服务紧在localhost监听，如果特殊需要，也可监听内网IP，尽量不要让MongoDB、RabbitMQ及PostgreSQL监听外网IP地址。
* 对于StackStorm专用RabbitMQ、MongoDB及PostgreSQL，尽量不要允许外部访问，仅限API本身访问（通过防火墙限制）。
* 尽量使用网络隔离，如使用DMZs。

### 2.1.8 升级为Brocade Workflow Composer

商用收费版本，略。

# 快速入门

参考：<https://docs.stackstorm.com/start.html>

本节将指导你完成一个简单的自动化任务：一个外部事件触发的trigger并执行对应action任务。

## 3.1 开始使用Stackstorm with CLI

使用StackStorm最好的入口是其命令行接口，尝试使用如下命令：

st2 --version

*# Get help. It's a lot. Explore.*

st2 -h

*# Authenticate and export the token. Make sure ST2\_AUTH\_URL and*

*# ST2\_API\_URL are set correctly (http vs https, endpoint, and etc).*

*# Replace the username and password in the example below appropriately.*

export ST2\_AUTH\_TOKEN**=**`st2 auth -t -p 'Ch@ngeMe' st2admin`

*# List the actions from the 'core' pack*

st2 action list --pack**=**core

st2 trigger list

st2 rule list

*# Run a local shell command*

st2 run core.local -- date -R

*# See the execution results*

st2 execution list

*# Run a shell command on remote hosts. Requires passwordless SSH configured.*

st2 run core.remote hosts**=**'localhost' -- uname -a

所有StackStorm操作都在REST API、Python及Javascript bindings中可用，详细参考：[CLI and Python Client](https://docs.stackstorm.com/reference/cli.html) 。

也可以通过WebUI做一些操作：检查历史记录、运行Actions、配置rules及安装packs等等。通过https://{YOUR\_ST2\_IP} URL登录，登录类似st2 CLI，默认用户名密码是：st2admin/Ch@ngeMe。

## 3.2 认证

CLI登录方式：st2 login st2admin --password 'Ch@ngeMe'，这会生成一个token，需要妥善保管，其他的认证方式参见：[docs](https://docs.stackstorm.com/authentication.html) 。

## 3.3 使用Actions

StackStorm默认只提供一些普通的Actions，你可以从StackStorm Exchange下载Actions，或者将自己的脚本部署为Action，以达到扩展的目的；查看可用actions命令：st2 action list，action名称通过ref定位，格式为pack.action\_name（如：core.local）；

查看action对应帮助命令：st2 action get <ref>或st2 action <ref> --help，示例：

*# List all the actions in the library*

st2 action list

*# Get action metadata*

st2 action get core.http

*# Display action details and parameters.*

st2 run core.http –help

# Using `--` to separate arguments

st2 run core.local -- ls -al

# Equivalent using `cmd` parameter

st2 run core.local cmd="ls -al"

# Crazily complex command passed with `cmd`

st2 run core.remote hosts='localhost' cmd="for u in bob phill luke; do echo \"Logins by \$u per day:\"; grep \$u /var/log/secure | grep opened | awk '{print \$1 \"-\" \$2}' | uniq -c | sort; done;"

# HTTP REST call to st2 action endpoint

st2 run -j core.http url="https://docs.stackstorm.com" method="GET"

# List of executions (most recent at the bottom)

st2 execution list

# Get execution by ID

st2 execution get <execution\_id>

# Get only the last 5 executions

st2 execution list -n 5

## 3.4 定义rule

StackStorm rule定义事件发生时执行actions或者workflows。事件被sensors监听。当sensors捕获一个事件后，触发某个trigger。trigger直接trip rule，rule检查criteria定义，如果匹配，就执行对应action。

rule 示例：

---

name: "sample\_rule\_with\_webhook"

pack: "examples"

description: "Sample rule dumping webhook payload to a file."

enabled: true

**trigger:**

type: "core.st2.webhook"

parameters:

url: "sample"

**criteria:**

trigger.body.name:

pattern: "st2"

type: "equals"

**action:**

ref: "core.local"

parameters:

cmd: "echo \"{{trigger.body}}\" >> ~/st2.webhook\_sample.out"

以上示例中给出的是一个webhook类型的trigger，对应url为https://{host}/api/v1/webhooks/sample；其中criteria条件中指定，如果payload中对应值为st2，则触发对应action；示例中action操作为：显示trigger.body到~/st2.webhook\_sample.out文件，这里对应的家目录为stanley用户；更详细的rule介绍参见：[Rules](https://docs.stackstorm.com/rules.html) 。

**trigger相关信息获取：**

# List all available triggers

st2 trigger list

# Check details on Interval Timer trigger

st2 trigger get core.st2.IntervalTimer

# Check details on the Webhook trigger

st2 trigger get core.st2.webhook

## 3.5 部署rule

rule可被自动load，也可使用如下api部署：

# Create the rule

st2 rule create /usr/share/doc/st2/examples/rules/sample\_rule\_with\_webhook.yaml

# List the rules

st2 rule list

# List the rules for the examples pack

st2 rule list --pack=examples

# Get the rule that was just created

st2 rule get examples.sample\_rule\_with\_webhook

当rule被部署以后，对应的webhook开始监听：https://{host}/api/v1/webhooks/{url}：

# Post to the webhook

curl -k https://localhost/api/v1/webhooks/sample -d '{"foo": "bar", "name": "st2"}' -H 'Content-Type: application/json' -H 'X-Auth-Token: put\_token\_here'

# Check if the action was executed (this shows the last action)

st2 execution list -n 1

# Check that the rule worked. By default, st2 runs as the stanley user.

sudo tail /home/stanley/st2.webhook\_sample.out

# And for fun, same post with st2

st2 run core.http method=POST body='{"you": "too", "name": "st2"}' url=https://localhost/api/v1/webhooks/sample headers='x-auth-token=put\_token\_here,content-type=application/json' verify\_ssl\_cert=False

# And for even more fun, using basic authentication over https

st2 run core.http url=https://httpbin.org/basic-auth/st2/pwd username=st2 password=pwd

# Check that the rule worked. By default, st2 runs as the stanley user.

sudo tail /home/stanley/st2.webhook\_sample.out

## 3.6 部署Examples

Examples中的rules，定制sensors，actions及workflows默认被安装在 [/usr/share/doc/st2/examples](https://github.com/StackStorm/st2/tree/master/contrib/examples/)目录下，为了部署Examples，可以这些目录下文件copy到/opt/stackstorm/packs/，使用如下命令reload生效：

# Copy examples to st2 content directory

sudo cp -r /usr/share/doc/st2/examples/ /opt/stackstorm/packs/

# Run setup

st2 run packs.setup\_virtualenv packs=examples

# Reload stackstorm context

st2ctl reload --register-all

可以从 [StackStorm Exchange](https://exchange.stackstorm.org/)获得更多的actions, sensors, rules。

## 3.7 Datastore

当大部分的数据需要被Stackstorm检索，你可以考虑将这些数据存储到一些通用变量中。使用Stackstorm的datastore服务可以实现这种需求，定义在datastore中的变量可以在rules及workflows中引用，格式类似：{{st2kv.system.my\_parameter}}。

创建kv示例：

# Create a new key value pair

st2 key set user stanley

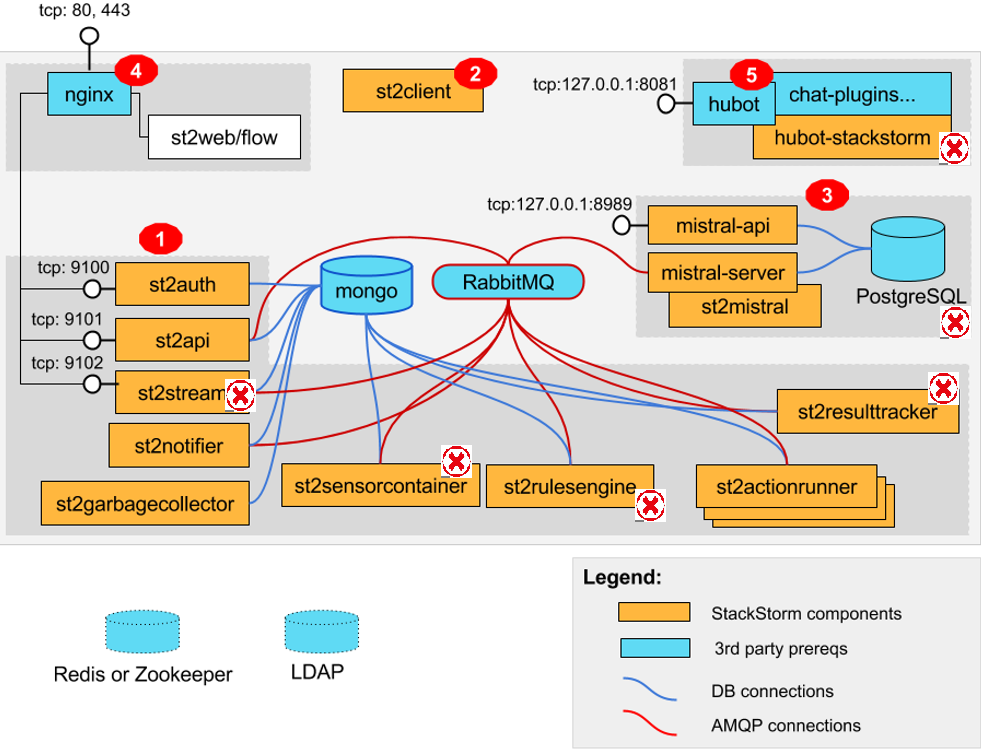
# List the key value pairs in the datastore

st2 key list

有关datastore更加详细的说明，请参考：[Datastore](https://docs.stackstorm.com/datastore.html)。

# 定制架构

## 4.1 说明



图中被标记的都为计划禁用服务（st2stream为必须启用的服务），被禁用服务全部计划用saltstack对应功能实现，只使用Stackstorm中的三个功能模块：WebUI – 历史展示、action定义执行及执行历史存放（替代saltstack自身的ruterner功能）。

## 4.2 定制过程

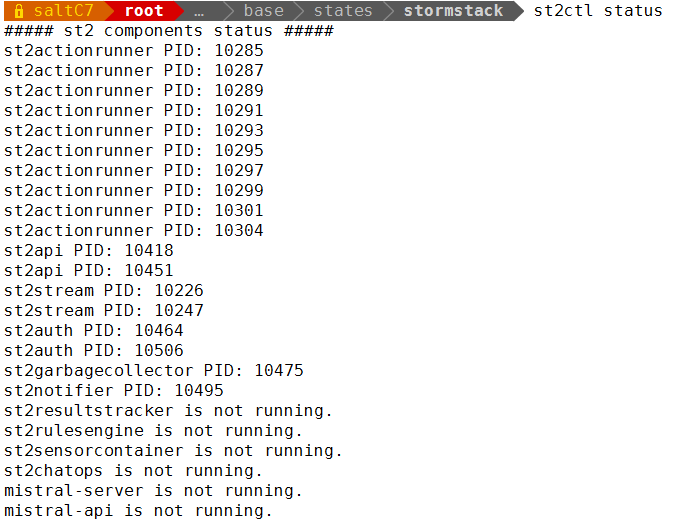
参见：<https://github.com/linora/iSaltstack/tree/master/base/states/stormstack/shell>

## 4.3 验证

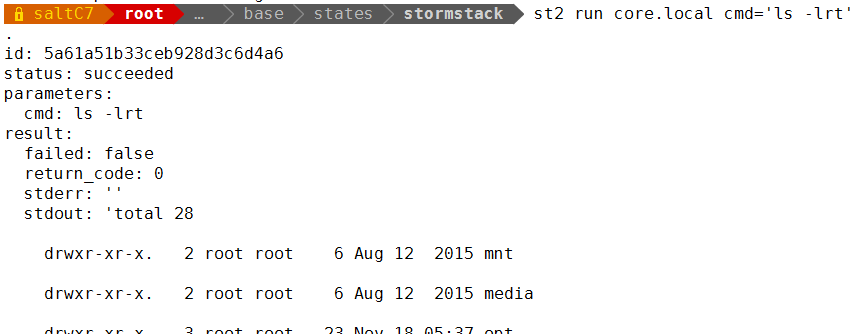
**1. 启动组件**

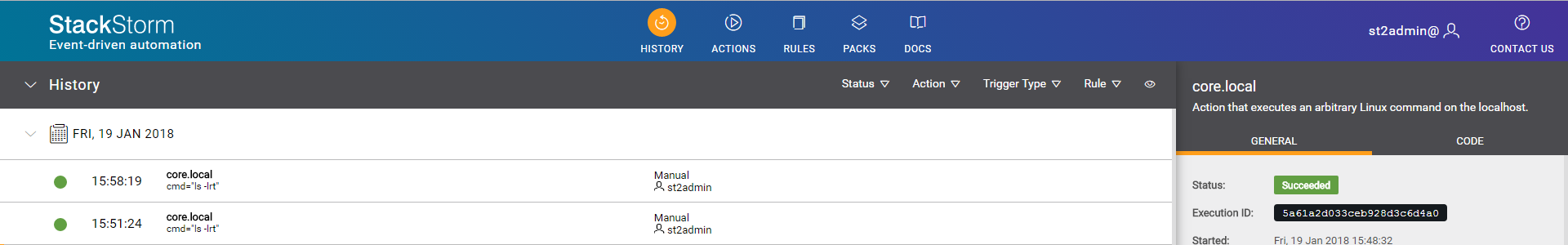
sh shell/custom\_st2.sh

sh shell/start\_custom\_componets.sh



**2. 验证**





# 五．salt with st2组合开发实例