****

**Ceph Teuthology**

目录

[**一．原理** 5](#_Toc531942560)

[1.1 简介 5](#_Toc531942561)

[1.2 架构图 5](#_Toc531942562)

[**二．安装** 5](#_Toc531942563)

[2.1 安装依赖包 5](#_Toc531942564)

[2.2 配置数据库 6](#_Toc531942565)

[2.3 配置paddles 6](#_Toc531942566)

[2.4 配置pulpito 7](#_Toc531942567)

[2.5 配置supervisor 9](#_Toc531942568)

[2.6 配置gitbuilder 10](#_Toc531942569)

[2.7 配置gitbuilder web 10](#_Toc531942570)

[2.8 ntp服务器配置 11](#_Toc531942571)

[c2.9 teuthology任务执行节点配置 11](#_Toc531942572)

[2.10 teuthology管理节点配置 12](#_Toc531942573)

[2.11 日志代理配置 14](#_Toc531942574)

[2.12 Git服务器搭建 15](#_Toc531942575)

[2.13 Jenkins搭建 19](#_Toc531942576)

[2.14 ceph-build运行 19](#_Toc531942577)

[**三．配置和运行** 19](#_Toc531942578)

[3.1 命令列表和teuthology.yaml配置项 19](#_Toc531942579)

[3.2 测试脚本 21](#_Toc531942580)

[3.3 运行测试脚本 22](#_Toc531942581)

[**四．问题集** 22](#_Toc531942582)

[**五．附录** 23](#_Toc531942583)

[5.1 命令参数详解 23](#_Toc531942584)

[ teuthology example.yaml 23](#_Toc531942585)

[ teuthology-ls <archive\_dir> 24](#_Toc531942586)

[ teuthology-lock 24](#_Toc531942587)

[ teuthology-schedule 25](#_Toc531942588)

[ teuthology-queue 26](#_Toc531942589)

[ teuthology-suite 26](#_Toc531942590)

[ teuthology-report 27](#_Toc531942591)

[ teuthology-worker 27](#_Toc531942592)

[ teuthology-results 27](#_Toc531942593)

[5.2 tasks和测试案例添加 27](#_Toc531942594)

[ install 28](#_Toc531942595)

[ ceph 28](#_Toc531942596)

[5.4 参考资料 29](#_Toc531942597)

***\* 版本修订记录 \****

|  |  |  |
| --- | --- | --- |
| ***版本号*** | ***修订时间*** | ***修订内容*** |
| *v1.0* | *2018-11-07* | *初版修订* |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

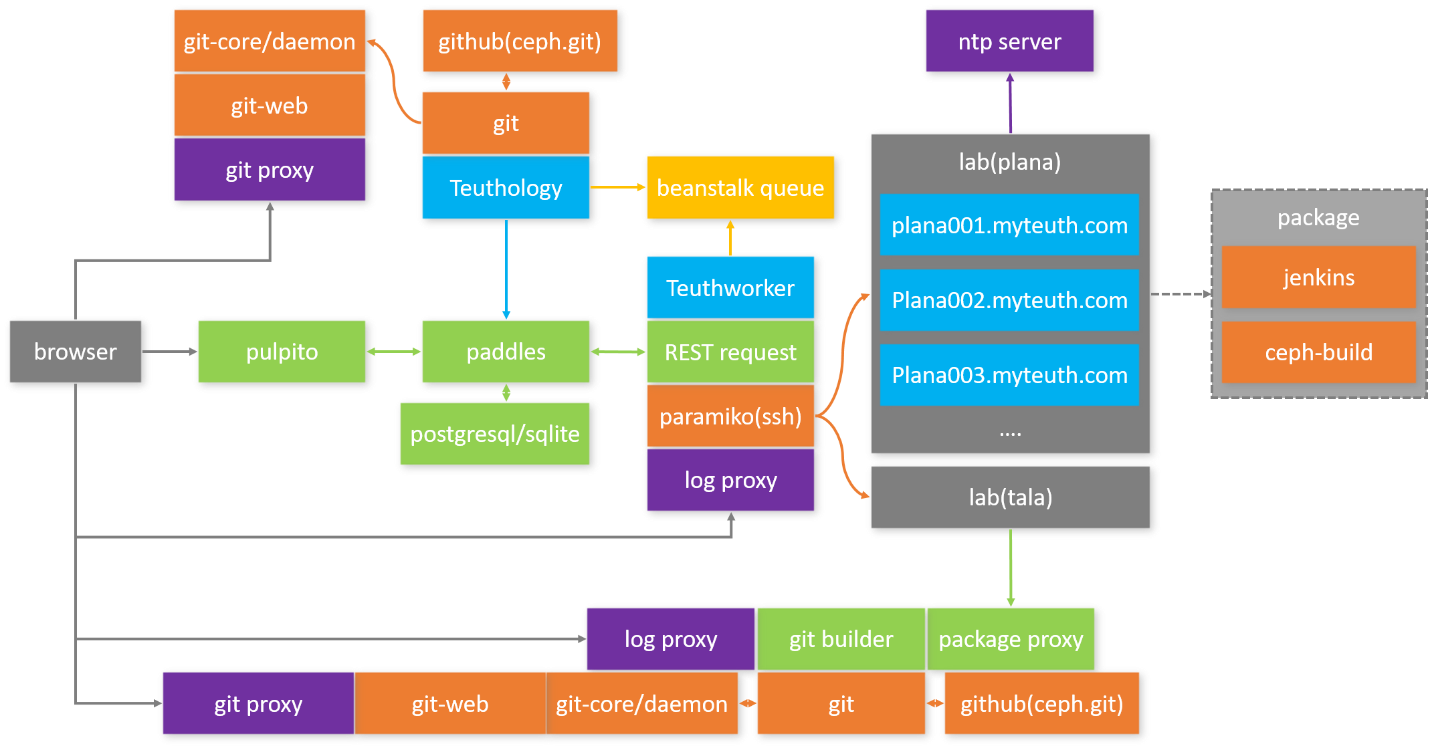
***\* Release Copyleft©free \****

**一．原理**

1.1 简介

teuthology是用python编写的CEPH自动化测试框架，使用由Paramiko模块实现的ssh来管理服务器。

1.2 架构图



**二．安装**

2.1 安装依赖包

|  |
| --- |
| root@admin:~# apt update  root@admin:~# apt-get install python-dev python-virtualenv postgresql postgresql-contrib postgresql-server-dev-all supervisor  root@admin:~# apt-get install git  # 以下为单独安装postgresql步骤  root@admin:~# cd /etc/apt/sources.list.d/  root@admin:/etc/apt/sources.list.d# touch pgdg.list  root@admin:/etc/apt/sources.list.d# vim pgdg.list  root@admin:/etc/apt/sources.list.d#  root@admin:/etc/apt/sources.list.d# cd  root@admin:~# apt install postgresql-10 |

2.2 配置数据库

|  |
| --- |
| root@admin:~# su - postgres  postgres@admin:~$ psql  psql (10.5 (Ubuntu 10.5-0ubuntu0.18.04))  Type "help" for help.  postgres=# \password postgres # 修改用户密码  Enter new password:  Enter it again:  postgres=# create database paddles; # 创建数据库  CREATE DATABASE  postgres=# \l # 查询数据库  List of databases  Name | Owner | Encoding | Collate | Ctype | Access privileges  -----------+----------+----------+-------------+-------------+-----------------------  paddles | postgres | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 |  postgres | postgres | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 |  template0 | postgres | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 | =c/postgres +  | | | | | postgres=CTc/postgres  template1 | postgres | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 | =c/postgres +  | | | | | postgres=CTc/postgres  (4 rows)  postgres=# \? # 查看帮助信息  postgres=# \q # 退出客户端  postgres@admin:~$ |

2.3 配置paddles

|  |
| --- |
| root@admin:~# useradd -m -s /bin/bash -g root -G root paddles  root@admin:~# echo paddles:yang | chpasswd  root@admin:~# mkdir /app/paddles  root@admin:~# chmod -R 775 /app/paddles/  root@admin:~# su - paddles  paddles@admin:~$ cd /app/paddles/  paddles@admin:~$ git clone https://github.com/ceph/paddles.git  paddles@admin:/app/paddles$ cd paddles/  paddles@admin:/app/paddles/paddles$ virtualenv ./virtualenv  paddles@admin:/app/paddles/paddles$ cp config.py.in config.py  paddles@admin:/app/paddles/paddles$ vim config.py  …  server = {  'port': '8080',  'host': '10.65.42.200'  }  …  sqlalchemy = {  …  'url': 'postgresql+psycopg2://postgres:post@localhost/paddles',  …  }  paddles@admin:/app/paddles/paddles$ source ./virtualenv/bin/activate  (virtualenv) paddles@admin:/app/paddles/paddles$  (virtualenv) paddles@admin:/app/paddles/paddles$ pip install -r requirements.txt  (virtualenv) paddles@admin:/app/paddles/paddles$ python setup.py develop # 初始化环境  (virtualenv) paddles@admin:/app/paddles/paddles$ pecan populate config.py # 数据表迁移  ==> LOADING ENVIRONMENT  ==> BUILDING SCHEMA  ==> STARTING A TRANSACTION...  No handlers could be found for logger "sqlalchemy.engine.base.Engine"  ==> COMMITING...  (virtualenv) paddles@admin:/app/paddles/paddles$ cp alembic.ini.in alembic.ini # 配置数据库迁移工具  (virtualenv) paddles@admin:/app/paddles/paddles$ vim alembic.ini  sqlalchemy.url = postgresql+psycopg2://postgres:post@localhost/paddles # 配置数据库库迁移工具  (virtualenv) paddles@admin:/app/paddles/paddles$ alembic stamp head  INFO [alembic.runtime.migration] Context impl PostgresqlImpl.  INFO [alembic.runtime.migration] Will assume transactional DDL.  INFO [alembic.runtime.migration] Running stamp\_revision -> 11e2594da07b # 触发迁移生效  # 启动测试  virtualenv) paddles@admin:/app/paddles/paddles$ pecan serve config.py  Starting server in PID 26348  serving on http://10.65.42.200:8080  No handlers could be found for logger "sqlalchemy.engine.base.Engine  # 启动后，浏览器输入<http://10.65.42.200:8080>，输出相应的json字符串则表示安装成功 |

2.4 配置pulpito

|  |
| --- |
| root@admin:~# useradd -m -s /bin/bash -g root -G root pulpito  root@admin:~# echo pulpito:yang | chpasswd  root@admin:~# mkdir /app/pulpito  root@admin:~# chmod 775 /app/pulpito/  root@admin:~# su - pulpito  pulpito@admin:~$ cd /app/pulpito/  pulpito@admin:/app/pulpito$ git clone https://github.com/ceph/pulpito.git  Cloning into 'pulpito'...  remote: Enumerating objects: 1725, done.  remote: Total 1725 (delta 0), reused 0 (delta 0), pack-reused 1725  Receiving objects: 100% (1725/1725), 1014.87 KiB | 446.00 KiB/s, done.  Resolving deltas: 100% (1238/1238), done.  pulpito@admin:/app/pulpito$ cd pulpito/  pulpito@admin:/app/pulpito/pulpito$  pulpito@admin:/app/pulpito/pulpito$ virtualenv ./virtualenv  Running virtualenv with interpreter /usr/bin/python2  New python executable in /app/pulpito/pulpito/virtualenv/bin/python2  Also creating executable in /app/pulpito/pulpito/virtualenv/bin/python  Installing setuptools, pkg\_resources, pip, wheel...done.  pulpito@admin:/app/pulpito/pulpito$ cp config.py.in prod.py  …  server = {  'port': 8081,  'host': '10.65.42.200'  }  paddles\_address = 'http://10.65.42.200:8080'  …  'debug': False # 关掉debug  …  pulpito@admin:/app/pulpito/pulpito$ source virtualenv/bin/activate  (virtualenv) pulpito@admin:/app/pulpito/pulpito$ pip install -r requirements.txt  # 测试  # 先启动运行paddles  # 启动pulpito  (virtualenv) pulpito@admin:/app/pulpito/pulpito$ python run.py  # 浏览器输入http://10.65.42.200:8081 |

2.5 配置supervisor

|  |
| --- |
| root@admin:~# pip2 install supervisor  root@admin:~# cd /etc/supervisor/conf.d/  root@admin:/etc/supervisor/conf.d# cat paddles.conf  [program:paddles]  user=paddles  environment=HOME="/home/paddles",USER="paddles"  directory=/app/paddles/paddles  command=/app/paddles/paddles/virtualenv/bin/gunicorn\_pecan -c gunicorn\_config.py config.py  autostart=true  autorestart=true  redirect\_stderr=true  stdout\_logfile = /app/paddles/paddles/paddles.out.log  stderr\_logfile = /app/paddles/paddles/paddles.err.log  root@admin:/etc/supervisor/conf.d# cat pulpito.conf  [program:pulpito]  user=pulpito  directory=/app/pulpito/pulpito/  command=/app/pulpito/pulpito/virtualenv/bin/python run.py  environment=HOME="/home/pulpito",USER="pulpito"  autostart=true  autorestart=true  redirect\_stderr=true  stdout\_logfile = /app/pulpito/pulpito/pulpito.out.log  stderr\_logfile = /app/pulpito/pulpito/pulpito.err.log  # 启动服务  root@admin:/etc/supervisor# systemctl reload supervisor  root@admin:/etc/supervisor# systemctl restart supervisor  root@admin:/etc/supervisor# supervisorctl start all  root@admin:/etc/supervisor# supervisorctl status  paddles RUNNING pid 30055, uptime 0:00:12  pulpito RUNNING pid 30056, uptime 0:00:12 |

2.6 配置gitbuilder

|  |
| --- |
| root@admin:~# mkdir /app/gitbuilder  root@admin:~# cd /app/gitbuilder/  root@admin:/app/gitbuilder# git clone https://github.com/ceph/gitbuilder.git  Cloning into 'gitbuilder'...  remote: Enumerating objects: 622, done.  remote: Total 622 (delta 0), reused 0 (delta 0), pack-reused 622  Receiving objects: 100% (622/622), 124.72 KiB | 265.00 KiB/s, done.  Resolving deltas: 100% (339/339), done.  root@admin:/app/gitbuilder# cd gitbuilder/  root@admin:/app/gitbuilder/gitbuilder# cp build.sh.example build.sh |

2.7 配置gitbuilder web

|  |
| --- |
| # 编译安装httpd，可参考2.12节的安装步骤  # 配置httd.conf  root@admin:/app/httpd/httpd/conf# vim httpd.conf  …  LoadModule cgid\_module modules/mod\_cgid.so # 取消注释  ServerName 10.65.42.200:80  Alias /gitbuilder/ /app/gitbuilder/gitbuilder/out/  <Directory "/app/gitbuilder/gitbuilder/out/">  Options +FollowSymLinks +ExecCGI  AllowOverride All  # Order Allow,Deny  # Allow from all  Require all granted  DirectoryIndex index.cgi  AddHandler cgi-script .cgi  </Directory>  …  # 启动httpd服务器  root@admin:/app/httpd/httpd/bin# ./apachectl -k start  # 浏览器输入http://10.65.42.200/gitbuilder/ |

2.8 ntp服务器配置

|  |
| --- |
| root@admin1:~# apt install ntp ntpdate  root@admin1:~# vim /etc/ntp.conf  …  restrict 10.65.42.0 mask 255.255.254.0 nomodify  …  server 127.127.1.0 # ATOM(PPS)  fudge 127.127.1.0 stratum 10 # enable PPS API  …  root@admin1:~# systemctl restart ntdp  root@admin1:~# ntpdate 0.ubuntu.pool.n  root@admin1:~# hwclock -w |

c2.9 teuthology任务执行节点配置

|  |
| --- |
| root@exec1:~# vim /etc/ssh/sshd\_config  …  PermitRootLogin yes  …  root@exec1:~# systemctl restart sshd  root@exec1:~# apt install ansible  # root@exec1:~# apt install ntp  root@exec1:~# useradd -m -g root -G root -s /bin/bash ubuntu  root@exec1:~# echo ubuntu:yang | chpasswd  root@exec1:~# vim /etc/sudoers.d/Ubuntu  ubuntu ALL=(ALL) NOPASSWD:ALL  root@exec1:~# apt install ntp  root@exec1:~# vim /etc/ntp.conf  # 如果只想使用内网ntp服务器，把pool也注释掉  …  server 10.65.42.204  …  root@exec1:~# systemctl restart ntp  root@exec1:~# ntpq -p  remote refid st t when poll reach delay offset jitter  ==============================================================================  \*10.65.42.204 203.107.6.88 3 u 48 64 337 0.234 3.882 2.667  root@exec1:~# ntpdate -u 10.65.42.204  20 Nov 01:38:45 ntpdate[2249]: adjust time server 10.65.42.204 offset -0.038386 sec  root@exec1:~# hwclock -w |

2.10 teuthology管理节点配置

|  |
| --- |
| root@admin1:~# apt-get -y install git python-dev python-pip python-virtualenv libevent-dev python-libvirt beanstalkd  [root@mon3 ~]# systemctl enable beanstalkd  [root@mon3 ~]# systemctl start beanstalkd  [root@mon3 ~]# systemctl status beanstalkd  ● beanstalkd.service - Beanstalkd Fast Workqueue Service  Loaded: loaded (/usr/lib/systemd/system/beanstalkd.service; enabled; vendor preset: disabled)  Active: active (running) since Tue 2018-12-04 13:44:18 CST; 5s ago  [root@mon3 ~]# netstat -na | grep 11300  tcp 0 0 0.0.0.0:11300 0.0.0.0:\* LISTEN  root@admin1:~# useradd -m -g root -G root -s /bin/bash teuthology  root@admin1:~# echo teuthology:yang|chpasswd  root@admin1:~# useradd -m -g root -G root -s /bin/bash teuthworker  root@admin1:~# echo teuthworker:yang|chpasswd  root@admin1:~# vim /etc/sudoers  # 追加  teuthology ALL=(ALL) NOPASSWD:ALL  teuthworker ALL=(ALL) NOPASSWD:ALL  # 创建配置文件  root@admin1:~# vim /etc/teuthology.yaml  root@admin1:~# cat /etc/teuthology.yaml  # paddles所在服务器  lock\_server: 'http://10.65.42.200:8080'  # paddles所在服务器  results\_server: 'http://10.65.42.200:8080'  # 域名，创建slave节点时有用到  lab\_domain: ‘myteuth.com'  # beanstalkd队列服务器，第一步安装的，就在我们本地，默认端口是11300  queue\_host: 127.0.0.1  queue\_port: 11300  # 本地归档，直接放在执行者的家目录下  archive\_base: /home/teuthworker/archive  verify\_host\_keys: false  ceph\_git\_base\_url: http://github.com/ceph/  gitbuilder\_host: '10.65.42.200'  reserve\_machines: 1  # 归档目录，直接写本机的地址加/teuthology即可  archive\_server: http://10.65.42.203/teuthology/  max\_job\_time: 86400  root@admin1:~# apt-get -y install git python-dev python-pip python-virtualenv libevent-dev python-libvirt beanstalkd  root@admin1:~# apt-get -y install libssl-dev libmysqlclient-dev libffi-dev libyaml-dev  root@admin1:~# sudo apt-get -y install qemu-utils libev-dev libvirt-dev  root@admin1:~# su - teuthology  teuthology@admin1:~$ mkdir src  teuthology@admin1:~$ git clone https://github.com/ceph/teuthology.git src/teuthology  Cloning into 'src/teuthology'...  remote: Enumerating objects: 2, done.  remote: Counting objects: 100% (2/2), done.  remote: Compressing objects: 100% (2/2), done.  remote: Total 26353 (delta 0), reused 0 (delta 0), pack-reused 26351  Receiving objects: 100% (26353/26353), 8.33 MiB | 347.00 KiB/s, done.  Resolving deltas: 100% (18472/18472), done.  Checking connectivity... done.  teuthology@admin1:~$ cd src/teuthology/  teuthology@admin1:~/src/teuthology$ ./bootstrap  teuthology@admin1:~/src/teuthology$ wget https://raw.githubusercontent.com/ceph/teuthology/master/docs/\_static/create\_nodes.py  teuthology@admin1:~/src/teuthology$ vim create\_nodes.py  …  paddles\_url = 'http://10.65.42.200:8080/'  machine\_type = 'plana'  lab\_domain = ‘myteuth.com'  # Don't change the user. It won't work at this time.  user = 'ubuntu'  # We are populating 'typica003' -> 'typica192'  machine\_index\_range = range(3,4)  …  teuthology@admin1:~/src/teuthology$ vim /etc/hosts  # 添加  10.65.42.203 plana003.myteuth.com  10.65.42.205 plana004.myteuth.com  10.65.42.206 plana005.myteuth.com  # 免密配置  teuthology@admin1:~/src/teuthology$ ssh-keygen  teuthology@admin1:~/src/teuthology$ ssh-copy-id ubuntu@plana003.myteuth.com  teuthology@admin1:~/src/teuthology$ source ./virtualenv/bin/activate  (virtualenv) teuthology@admin1:~/src/teuthology$ python create\_nodes.py  root@admin1:~# su - teuthworker  teuthworker@admin1:~$ mkdir src  teuthworker@admin1:~$ git clone https://github.com/ceph/teuthology.git src/teuthology  Cloning into 'src/teuthology'...  remote: Enumerating objects: 2, done.  remote: Counting objects: 100% (2/2), done.  remote: Compressing objects: 100% (2/2), done.  remote: Total 26353 (delta 0), reused 0 (delta 0), pack-reused 26351  Receiving objects: 100% (26353/26353), 8.33 MiB | 351.00 KiB/s, done.  Resolving deltas: 100% (18472/18472), done.  Checking connectivity... done.  teuthworker@admin1:~$ cd src/teuthology/  teuthworker@admin1:~/src/teuthology$ ./bootstrap  teuthworker@admin1:~/src/teuthology$ source virtualenv/bin/activate  (virtualenv) teuthworker@admin1:~/src/teuthology$ cd  (virtualenv) teuthworker@admin1:~$ mkdir archive/worker\_logs  (virtualenv) teuthworker@admin1:~$ mkdir -p archive/worker\_logs  (virtualenv) teuthworker@admin1:~$ mkdir bin  (virtualenv) teuthworker@admin1:~$ wget -O ~/bin/worker\_start https://raw.githubusercontent.com/ceph/teuthology/master/docs/\_static/worker\_start.sh  (virtualenv) teuthworker@admin1:~$ echo 'PATH="$HOME/src/teuthology\_master/virtualenv/bin:$PATH"' >> ~/.profile  (virtualenv) teuthworker@admin1:~$ source .profile  (virtualenv) teuthworker@admin1:~$ |

2.11 日志代理配置

|  |
| --- |
| # 使用nginx作为日志代理  # 下载pcre  wget ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/pcre-8.39.tar.gz  tar -zxvf pcre-8.39.tar.gz  # 下载编译zlib  wget http://zlib.net/zlib-1.2.11.tar.gz  tar -zxvf zlib-1.2.11.tar.gz  cd zlib-1.2.11/  ./configure  sudo make && sudo make install  # 下载openssl  wget https://www.openssl.org/source/openssl-1.0.1t.tar.gz  tar -zxvf nginx-1.14.1.tar.gz  # 下载编译nginx  wget http://nginx.org/download/nginx-1.14.1.tar.gz  tar -zxvf nginx-1.14.1.tar.gz  ./configure --prefix=/your-dir/nginx --with-pcre=/your-dir/pcre-8.39 --with-openssl=/your-dir/openssl-1.0.1t  make && make install  # 配置nginx.conf文件  user root;  …  location / {  allow all;  autoindex on;  root /home/teuthworker/archive;  default\_type text/plain;  }  …  # 启动nginx  ./nginx  # 更新paddles配置文件config.py  paddles@admin:/app/paddles/paddles$ vim config.py  …  job\_log\_href\_templ = 'http://10.65.42.204/worker\_logs'  …. |

2.12 Git服务器搭建

|  |
| --- |
| root@admin1:~# apt -y install git git-core git-daemon-sysinit  root@admin1:~# vim /etc/default/git-daemon  root@admin1:~# cat /etc/defaultuse/git-daemon  …  GIT\_DAEMON\_ENABLE=true  GIT\_DAEMON\_USER=gitdaemon  GIT\_DAEMON\_BASE\_PATH=/  GIT\_DAEMON\_DIRECTORY=/git/  # Additional options that are passed to the Daemon.  GIT\_DAEMON\_OPTIONS="--export-all --enable=upload-pack --enable=upload-archive --enalbe=receive-pack --informative-errors"  …  root@admin1:~# systemctl restart git-daemon  root@admin1:/# mkdir git  root@admin1:/# cd git  root@admin1:/git# ls  # ceph下载需要很长时间，这里是一个类似案例  root@admin1:/git# git init --bare sample.git  root@admin1:/git# git clone https://github.com/EarthWind/RDMA\_Know.git RDMA\_Know.git  root@admin1:/git# cd RDMA\_Know.git/  root@admin1:/git/RDMA\_Know.git# vim config  root@admin1:/git/RDMA\_Know.git# cat config  [core]  repositoryformatversion = 0  filemode = true  bare = true  [remote "origin"]  url = https://github.com/EarthWind/RDMA\_Know.git  fetch = +refs/\*:refs/\*  mirror = true  [daemon]  uploadpack = true  uploadarch = true  receivepack = true  allowunreachable = true  root@admin1:~# apt install -y gitweb  root@admin1:~# vim /etc/gitweb.conf  …  $projectroot = "/git/";  …  # 安装httpd  root@admin1:/apache# wget http://mirrors.tuna.tsinghua.edu.cn/apache//apr/apr-1.6.5.tar.gz  root@admin1:/apache# tar -zxvf apr-1.6.5.tar.gz  root@admin1:/apache# mkdir apr  root@admin1:/apache# cd apr-1.6.5/  root@admin1:/apache/apr-1.6.5# ./configure --prefix=/apache/apr  root@admin1:/apache/apr-1.6.5# make && make install  root@admin1:/apache# wget http://mirrors.tuna.tsinghua.edu.cn/apache//apr/apr-util-1.6.1.tar.gz  root@admin1:/apache# tar -zxvf apr-util-1.6.1.tar.gz  root@admin1:/apache# mkdir apr-util  root@admin1:/apache# cd apr-util-1.6.1/  root@admin1:/apache/apr-util-1.6.1# ./configure --prefix=/apache/apr-util --with-apr=/apache/apr  root@admin1:/apache/apr-util-1.6.1# make && make install  root@admin1:/apache# wget https://sourceforge.net/projects/pcre/files/pcre/8.38/pcre-8.38.tar.gz  root@admin1:/apache# tar -zxvf pcre-8.38.tar.gz  root@admin1:/apache# mkdir pcre  root@admin1:/apache# cd pcre-8.38/  root@admin1:/apache/pcre-8.38# ./configure --prefix=/apache/pcre  root@admin1:/apache/pcre-8.38# make && make install  root@admin1:/apache# wget http://mirrors.shu.edu.cn/apache//httpd/httpd-2.4.37.tar.gz  root@admin1:/apache# tar -zxvf httpd-2.4.37.tar.gz  root@admin1:/apache# mkdir httpd  root@admin1:/apache# cd httpd-2.4.37/  root@admin1:/apache/httpd-2.4.37# ./configure --prefix=/apache/httpd --with-apr=/apache/apr --with-apr-util=/apache/apr-util --with-pcre=/apache/pcre --enable-module=shared  root@admin1:/apache/httpd-2.4.37# make && make install  # 编辑httpd.conf文件  root@admin1:/apache/httpd/conf# vim httpd.conf  …  LoadModule cgid\_module modules/mod\_cgid.so # 取消注释  Listen 8000  ServerName 10.65.42.204:8000  Include conf/extra/gitweb.conf # 增加该行  ….  root@admin1:/apache/httpd/conf# cp /etc/apache2/conf-available/gitweb.conf ./extra/  root@admin1:/apache/httpd/conf# cd extra/  root@admin1:/apache/httpd/conf/extra# cat gitweb.conf  <IfModule mod\_alias.c>  <IfModule mod\_mime.c>  <IfModule mod\_cgi.c>  Define ENABLE\_GITWEB  </IfModule>  <IfModule mod\_cgid.c>  Define ENABLE\_GITWEB  </IfModule>  </IfModule>  </IfModule>  <IfDefine ENABLE\_GITWEB>  Alias /gitweb /usr/share/gitweb/  <Directory /usr/share/gitweb/>  Options +FollowSymLinks +ExecCGI  AllowOverride All  Require all granted  DirectoryIndex index.cgi  AddHandler cgi-script .cgi  </Directory>  </IfDefine>  root@admin1:/apache/httpd/conf/extra# cd ../../bin/  root@admin1:/apache/httpd/bin# ./apachectl -k start  # 浏览器输入http://10.65.42.204:8000/gitweb/    ############# 新UI GitWeb ##############  root@admin1:/apache# git clone https://github.com/kogakure/gitweb-theme.git  Cloning into 'gitweb-theme'...  remote: Enumerating objects: 311, done.  remote: Total 311 (delta 0), reused 0 (delta 0), pack-reused 311  Receiving objects: 100% (311/311), 709.06 KiB | 367.00 KiB/s, done.  Resolving deltas: 100% (114/114), done.  root@admin1:/apache# cd gitweb-theme/  root@admin1:/apache/gitweb-theme# ./setup -vi --install  ./setup: [NOTICE] - Target not set, using default path: '/usr/share/gitweb'  ./setup:  ./setup: Is this ok? [y] | [n] : y  ./setup:  ./setup: Backing up original files, continue? [y] | [n] : y  ./setup:  ./setup: Backing up...  ./setup:  '/usr/share/gitweb/static/gitweb.css' -> '/usr/share/gitweb/static/gitweb.css.bak'  '/usr/share/gitweb/static/git-favicon.png' -> '/usr/share/gitweb/static/git-favicon.png.bak'  '/usr/share/gitweb/static/git-logo.png' -> '/usr/share/gitweb/static/git-logo.png.bak'  ./setup:  ./setup: ...done  ./setup:  ./setup: Linking theme files, continue? [y] | [n] : y  ./setup:  ./setup: Linking...  ./setup:  '/usr/share/gitweb/static/gitweb.css' -> '/apache/gitweb-theme/gitweb.css'  '/usr/share/gitweb/static/git-favicon.png' -> '/apache/gitweb-theme/git-favicon.png'  '/usr/share/gitweb/static/git-logo.png' -> '/apache/gitweb-theme/git-logo.png'  ./setup:  ./setup: ...done  ./setup:  ./setup: [NOTICE] - Installation complete!  # 重新输入http://10.65.42.204:8000/gitweb/ |

2.13 Jenkins搭建

|  |
| --- |
|  |

2.14 ceph-build运行

|  |
| --- |
|  |

**三．配置和运行**

3.1 命令列表和teuthology.yaml配置项

|  |
| --- |
| teuthology: 运行单个的jobs 可以指定多个任务配置文件，每个文件的配置相互叠加。运行该命令使需要在配置文件中手动指定job\_id和name才会生成相应的日志文件；  teuthology-coverage: 使用lcov进行代码覆盖率测试  teuthology-kill: Kill掉运行的jobs  teuthology-lock: 锁定，解锁和更新机器的状态  teuthology-ls: 列出一个归档目录的测试结果  teuthology-openstack: 使用Openstack作为后端  teuthology-nuke: 恢复机器到原始状态  teuthology-queue: 查询或删除队列中的jobs  teuthology-report: 提交测试结果到Web服务器(paddles)  teuthology-results: 邮件发送测试结果  teuthology-schedule: 调度单个job  teuthology-suit: 调度suite  teuthology-updatekeys: 更新服务器的ssh-key  teuthology-worker: 从beanstalk的TUBE中读取job  'archive\_base': '/home/teuthworker/archive',  'archive\_upload': None,  'archive\_upload\_key': None,  'archive\_upload\_url': None,  'automated\_scheduling': False,  'reserve\_machines': 5,  'ceph\_git\_base\_url': 'https://github.com/ceph/',  'ceph\_git\_url': None,  'ceph\_qa\_suite\_git\_url': None,  'use\_conserver': False,  'conserver\_master': 'conserver.front.sepia.ceph.com',  'conserver\_port': 3109,  'gitbuilder\_host': 'gitbuilder.ceph.com',  'githelper\_base\_url': 'http://git.ceph.com:8080',  'check\_package\_signatures': True,  'lab\_domain': 'front.sepia.ceph.com',  'lock\_server': 'http://paddles.front.sepia.ceph.com/',  'max\_job\_time': 259200, # 3 days  'nsupdate\_url': 'http://nsupdate.front.sepia.ceph.com/update',  'results\_server': 'http://paddles.front.sepia.ceph.com/',  'results\_ui\_server': 'http://pulpito.ceph.com/',  'results\_sending\_email': 'teuthology',  'results\_timeout': 43200,  'src\_base\_path': os.path.expanduser('~/src'),  'verify\_host\_keys': True,  'watchdog\_interval': 120,  'kojihub\_url': 'http://koji.fedoraproject.org/kojihub',  'kojiroot\_url': 'http://kojipkgs.fedoraproject.org/packages',  'koji\_task\_url': 'https://kojipkgs.fedoraproject.org/work/',  'baseurl\_template': 'http://{host}/{proj}-{pkg\_type}-{dist}-{arch}-{flavor}/{uri}',  'use\_shaman': True,  'shaman\_host': 'shaman.ceph.com',  'teuthology\_path': None,  'suite\_verify\_ceph\_hash': True,  'suite\_allow\_missing\_packages': False,  'openstack': {  'clone': 'git clone http://github.com/ceph/teuthology',  'user-data': 'teuthology/openstack/openstack-{os\_type}-{os\_version}-user-data.txt',  'ip': '1.1.1.1',  'machine': {  'disk': 20,  'ram': 8000,  'cpus': 1,  },  'volumes': {  'count': 0,  'size': 1,  },  }, |

3.2 测试脚本

|  |
| --- |
| roles:  - [mon.0, mds.0, osd.0]  - [mon.1, osd.1]  - [mon.2, client.0]  targets:  ubuntu@plana003.myteuth.com: ssh-rsa rsa-value  ubuntu@plana004.myteuth.com: ssh-rsa rsa-value  ubuntu@plana005.myteuth.com: ssh-rsa rsa-value  tasks:  - install:  - ceph:  - ceph-fuse: [client.0]  - interactive:  -overrides:  mon:  ceph:    运行脚本编写：   1. 每个脚本包含三个部分，roles,targets和tasks 2. targets定义运行任务所在的主机，roles定义每个主机的上安装的ceph进程类型，数组长度必须和targets数量一致，tasks定义需要运行的任务； 3. rsa-value的值为相应主机上文件/ etc/ssh/ssh\_host\_rsa\_key.pub的key值 4. install任务要先于其他所有任务 5. targets为需要可以解析的主机名，如果没有DNS服务器可以将解析放入/etc/hosts中，本机必须能够无秘登入远端主机，并且登入的用户可以无秘进行sudo 。 6. 支持的task有如下这些：install,ceph,ansible,args,clock,iscsi等 |

3.3 运行测试脚本

|  |
| --- |
| # 任务运行时需要指定相应的机器并锁定，有两种方式对服务器进行锁定：   * 在配置文件中targets部分指定运行任务的服务器，使用teuthology-lock对服务器进行锁定 * 在配置文件中指定lock\_server: 'http://10.65.42.200:8080/lock'配置项，不指定targets,运行任务时添加--lock参数，teuthology会自动锁定空闲服务器。   # source相应的virtualenv, 运行脚本example.yaml, 多个脚本时，配置项会叠加  (virtualenv) teuthology@admin1:~/src/teuthology/roles$ teuthology example.yaml |

**四．问题集**

|  |
| --- |
| **Q:**  creating pip-egg-info/psycopg2.egg-info  writing pip-egg-info/psycopg2.egg-info/PKG-INFO  writing top-level names to pip-egg-info/psycopg2.egg-info/top\_level.txt  writing dependency\_links to pip-egg-info/psycopg2.egg-info/dependency\_links.txt  writing manifest file 'pip-egg-info/psycopg2.egg-info/SOURCES.txt'  Error: could not determine PostgreSQL version from '10.5'  **A:**  编辑requirements.txt修改pysycopg2为psycopg2==2.7.3，参考https://stackoverflow.com/questions/49291428/error-could-not-determine-postgresql-version-from-10-3-django-on-heroku  **Q:**  symbol \_\_res\_maybe\_init version GLIBC\_PRIVATE not defined in file libc.so.6 with link time reference  **A:**  (virtualenv) paddles@admin:/app/paddles/paddles$ pip uninstall psycopg2  (virtualenv) paddles@admin:/app/paddles/paddles$ sudo apt install gcc  (virtualenv) paddles@admin:/app/paddles/paddles$ pip install --no-binary :all: psycopg2  **Q:**  Can't locate CGI.pm in @INC (you may need to install the CGI module)  **A:**  root@admin:/app/gitbuilder/gitbuilder# perl -e shell -MCPAN  cpan[1]> install CGI  **Q:**  Can't locate LockFile/Simple.pm  **A:**  root@admin:/app/gitbuilder/gitbuilder# cpan install LockFile::Simple  **Q:**  libtoolT': No such file or directory  **A:**  configure文件，查找 $RM "$cfgfile" 这个地方，用#注释掉.  确保libtool已经安装  **Q:**  git-daemon: unable to open supervise/ok: file does not exist  **A:** |

**五．附录**

5.1 命令参数详解

* teuthology example.yaml

|  |
| --- |
| -h, --help 输出帮助信息  -v, --verbose 输出详细运行日志  --version 获取teuthology的版本  -a DIR, --archive DIR 指定运行日志的生成目录  --description DESCRIPTION 指定job描述  --owner OWNER 指定job的所属者  --lock 自动锁定服务器，配置文件中不配置—targets  --machine-type MACHINE\_TYPE 指定运行任务的机器类别  --os-type OS\_TYPE 使任务在指定的操作系统的机器上运行  --os-version OS\_VERSION 使任务在指定操作系统版本的机器上运行  --block 和--lock参数一起使用，在未获取机器之前进程一致阻塞  --name NAME 指定任务名称  --suite-path SUITE\_PATH 指定测试suite目录，如果未指定，则会从github上拉去(如果指定该参数，这会把该目录加入sys.path，并导入tasks(import tasks)，也就是说该目录下需要存在tasks目录，否则就算指定了也会从github上拉去，如果没有指定该参数则分以下步骤：   * 获取example.yaml中的branch键的值设置为拉取的分支，如果没有则设置为master。获取example.yaml中的suite\_repo键的值设置为拉取的地址，并将teuthology键ceph\_qa\_suite\_git\_url的值设置为该值。获取example.yaml中的键为suite\_branch的值，如果有，则该值为最后拉去的分支值，没有则赋予branch的值。 * 如果teuthology的ceph\_qa\_suite\_git\_url值存在，则用该值作为拉去地址，默认为None,如果该值不存在，则是使用teuthology的ceph\_git\_url作为拉去地址，默认为None,如果该值不存在，则使用ceph\_git\_base\_url+ 'ceph-ci.git'作为拉去地址，ceph\_git\_base\_url的值默认为https://github.com/ceph/ * 使用获取的拉取地址和分支来下载源码到teuthology的src\_base\_path，默认为~/src，不存在则会自动创建该目录， 目录名称为url\_to\_dirname(url)\_branch。 * 将下载目录加入和exmaple.yaml中suite\_relpath的值合并加入sys.path，如果没有该值为空   '--archive-upload': None,  '--archive-upload-url': None,  '--ceph': 'master', # 测试的ceph分支  '--ceph-repo': 'https://github.com/ceph/ceph.git',  '--distro': 'ubuntu',  '--distro-version': None,  '--dry-run': False,  '--email': None,  '--filter': None,  '--filter-out': None,  '--flavor': 'basic',  '--help': False,  '--kernel': 'distro',  '--limit': '0',  '--machine-type': 'plana',  '--newest': '0',  '--num': '1',  '--owner': None,  '--priority': '1000',  '--rerun': None,  '--rerun-status.': False,  '--rerun-statuses': 'fail,dead',  '--seed': '-1',  '--sha1': None,  '--subset': None,  '--suite': 'rbd', # 调度的测试suite  '--suite-branch': None,  '--suite-dir': None,  '--suite-relpath': 'qa',  '--suite-repo': 'https://github.com/ceph/ceph.git',  '--teuthology-branch': 'master', # 需要拉取teuthology的版本  '--throttle': None,  '--timeout': '43200',  '--verbose': 0,  '--wait': False, # 是否阻塞知道suite测试完成 |

* teuthology-ls <archive\_dir>

|  |
| --- |
| -h, --hlep 打印该命令的帮助信息  -v, --verbose 输出测试日志的详细信息  <archive\_dir> 为测试日志的目录，指定的目录下有以job\_id命名的目录 |

* teuthology-lock

|  |
| --- |
| -h, --help 打印该命令的帮助信息  -v, --verbose 输出详细日志  --list 输出被’当前用户名@机器名’锁定的所有机器的详细信息，也可以指定机器，可以和--owner,--status和--locked一起使用  --brief 输出被’当前用户名@机器名’锁定的所有机器的简略信息  --lock 锁定特定机器，指定需要锁定的机器名  --unlock 解锁机器，指定需要解锁的机器名  --lock-many NUM\_TO\_LOCK 在-m参数类型的机器中锁定NUM\_TO\_LOCK个机器  --update 更新指定机器的描述或状态，和--desc或--status一起使用  --summary 统计被’当前用户名@机器名’锁定的机器的数量信息和状态信息  -a,--all 输出所有机器，和--list,--list-targets,--brief一起使用  --owner 指定机器的锁定用户，和--lock一起使用  -f 在第一个错误出现的时候程序不退出，继续锁定或解锁其他机器  --desc 指定锁定描述  --desc-pattern DESC\_PATTERN 指定锁定描述  -m MACHINE\_TYPE, --machine-type MACHINE\_TYPE  指定操作的的机器类别，当前机器类别有mira | plana | burnupi | vps | saya | tala  --status {up,down} 用于过滤或更新机器的状态  --locked {true,false} 用于过滤机器的锁定状态  --os-type OS\_TYPE 指定机器的操作系统类别  --os-version OS\_VERSION 指定机器的操作系统版本  --arch ARCH 指定系统架构x86\_64, i386, armv7, arm64  --json-query JSON\_QUERY --list和--brief的json请求格式 |

* teuthology-schedule

|  |
| --- |
| -h, --help 输出命令的帮助信息  -v, --verbose 打印详细日志  -n <name>, --name <name> 该job所属的suite名称  -d <desc>, --description <desc> job描述  -o <owner>, --owner <owner> job所属者  -w <worker>, --worker <worker> 任务运行使用的机器类别，也为beanstalk中tube名称，默认为plana  -p <priority>, --priority <priority> job调度的优先级，越小优先级越高  -N <num>, --num <num> 该job调度的次数，默认为1  --first-in-suite 是否为suite中的第一个job,默认为False  --last-in-suite 是否为suite中的最后一个job，默认为False  --email <email> 指定测试结果邮件发送地址，和--last-in-suiet一起使用  --timeout <timeout> 指定等待job完成的时间，到时时发送邮件，和—last-in-suite一起使用  --seed <seed> job重新运行的随机种子，和--first-in-suite一起使用  --subset <subset> 传递给teuthology-suite，和--first-in-suite一起使用  --dry-run 仅输出配置文件信息，不运行测试  <example.yaml> 指定其他的配置文件 |

* teuthology-queue

|  |
| --- |
| # 查询beanstalk队列中的job  -h, --help 打印命令的帮助信息  -D, --delete PATTERN 删除队列  -d, --description 输出job描述  -r, --runs 仅输出运行时的job名称  -f, --full 输出job的所有配置信息  -s, --status 输出队列状态  -p, --pause SECONDS 将队列暂停指定的时间 |

* teuthology-suite

|  |
| --- |
| -h, --help 打印命令的帮助信息  -v, --verbose 输出debug信息  --dry-run 仅输运行的配置信息，不运行测试  -s <suite>, --suite <suite> 需要调度的suite  -c <ceph>, --ceph <ceph> 指定ceph的版本  -S <sha1>, --sha1 <sha1> 指定ceph提交hash值，覆盖-c参数  -n <newest>, --newest <newest>  -k <kernel>, --kernel <kernel>  -f <flavor>, --flavor <flavor>  -t <branch>, --teuthology-branch <branch>  teuthology的版本，默认为master  -m <type>, --machine-type <type>  运行该suite的机器类型  -d <distro>, --distro <distro> 指定运行任务的操作系统类型  -D <distroversion>, --distro-version <distroversion>  指定操作系统版本  --ceph-repo <ceph\_repo> 指定ceph源码拉取地址，默认值为https://github.com/ceph/ceph-ci.git  --suite-repo <suite\_repo> 指定suite源码地址，默认为https://github.com/ceph/ceph-ci.git  --suite-relpath <suite\_relpath> 指定suite的相对目录，如下载的ceph代码的suites目录为./qa/suites，该值可些微qa/suites，默认值为空  --suite-branch <suite\_branch> 替换ceph branch的值，指定ceph源码分支  --suite-dir <suite\_dir> 如果suite目录已经下载，可以用该参数指定， |

* teuthology-report

|  |
| --- |
| # 将测试结果发送到服务器  -h, --help 打印命令的帮助信息  -a ARCHIVE, --archive ARCHIVE 测试时指定的日志归档目录  -r [RUN ...], --run [RUN ...] 提交一个RUN的结果  -j [JOB ...], --job [JOB ...] 提交一个JOB的结果  --all-runs 提交ARCHIVE中所有RUN的结果  -R, --refresh 更新paddles中所有已经提交的RUN  -s SERVER, --server SERVER 指定paddles地址  -n, --no-save  -D, --dead 标记指定的jobs或所有job的状态为dead，和--refresh一起使用  -v, --verbose 输出debug信息 |

* teuthology-worker

|  |
| --- |
| # 从beanstalk队列中获取任务并运行集成测试  -h, --help 打印命令的帮助信息  -v, --verbose 打印debug信息  --archive-dir DIR 指定测试日志归档目录  -l LOG\_DIR, --log-dir LOG\_DIR 指定运行日志目录  -t TUBE, --tube TUBE 指定从beanstalk的那个tube中获取任务 |

* teuthology-results

|  |
| --- |
| # 发送集成测试结果日志  -h, --help 打印命令帮助信息  -v, --verbose 输出debug信息  --dry-run 仅打印发送内容，不尽兴邮件的发送  --email EMAIL 指定接收邮件地址  --timeout TIMEOUT 指定所有测试完成的等待时间  --archive-dir DIR 指定集成测试的日志归档目录  --name NAME 指定测试的suite名称  --subset SUBSET 指定传递到teuthology-suite的subset参数  --seed SEED 指定teuthology-suite的随机seed |

5.2 tasks和测试案例添加

teuthology的所有task都位于文件ceph/qa/tasks目录和teuthology/teuthology/task下的目录对应的\*.py文件，相应的配置参数可以参看文件内的task函数

* install

|  |
| --- |
| tasks:  - install:  project: ceph # 没有指定时默认为ceph  branch: bar # 指定相应的分支  tag: v18.01  sha1: 1234  debuginfo: true # 为false时过滤掉安装包列表中以-dbg和-debuginfo结尾的安装包，不包括extra\_packages中的，  flavor: basic  extra\_packages:  extra\_system\_packages:  exclude\_packages:  extras: # 该参数存在时，不下载packages中的包列表，而是下载固定的包debs = ['ceph-test', 'ceph-fuse','librados2', 'librbd1','python-ceph']，rpms = ['ceph-fuse', 'librbd1', 'librados2', 'ceph-test', 'python-ceph']，通常用在使用ceph-deploy安装ceph或测试ceph-deploy的场景  wait\_for\_package: False  packages:  rhbild: # 有该参数时，会结合playbook,其他的参数没有效果  playbook: # 不指定则为cephlab.yml  # 安装包列表qa/packages  # 例  - install:  project: samba  branch: foo  extra\_packages: ['samba']  - install:  extra\_packages:  deb: ['librados-dev', 'libradosstriper-dev']  rpm: ['librados-devel', 'libradosstriper-devel']  extra\_system\_packages:  deb: ['libboost-system-dev']  rpm: ['boost-devel']  - install:  rhbuild: 1.3.0  playbook: downstream\_setup.yml  vars:  yum\_repos:  - url: "http://location.repo"  name: "ceph\_repo" |

* ceph

|  |
| --- |
| tash:  - ceph:  branch: foo  tag: v0.42.13  sha1: 1376a5ab0c89780eab39ffbbe436f6a6092314ed  path: /home/sage/ceph  overage: true  fs: xfs  mkfs\_options: [-b,size=65536,-l,logdev=/dev/sdc1]  mount\_options: [nobarrier, inode64]  valgrind:  mds.1: --tool=memcheck  osd.1: [--tool=memcheck, --leak-check=no]  conf:  section:  key: value  log-whitelist: ['foo.\*bar', 'bad message']  cluster: test  - ceph:  cluster: ceph  - ceph:  cluster: backup |

5.4 参考资料

【1】Ceph Gitbuilder: http://gitbuilder.ceph.com/

【2】http://pulpito.ceph.com/

【3】https://github.com/ceph/paddles

【4】https://github.com/ceph/pulpito

【5】Gitweb安装于配置：https://www.cnblogs.com/gtarcoder/p/5309229.html

【6】Apache目录访问权限：https://blog.csdn.net/ddv1999/article/details/78434324/

【7】Python + Apache + CGI配置：https://www.cnblogs.com/feifeidxl/p/5556696.html

【8】Gitweb学习笔记：https://blog.csdn.net/fromcaolei/article/details/78530059?locationNum=4&fps=1

【9】ntp服务器配置：https://www.cnblogs.com/jczhu/p/5851268.html

【10】Git服务器：https://www.cnblogs.com/badwood316/p/4589407.html

【11】Ceph Jenkins: https://jenkins.ceph.com/

【12】Ceph build: https://github.com/ceph/ceph-build

【13】https://blog.csdn.net/cyh201201/article/details/51889594

【14】Jenkins下载地址：https://pkg.jenkins.io/redhat-stable/