****

**Ceph Container**

目录

[**一．编译启动** 4](#_Toc523819095)

[1.1 安装docker 4](#_Toc523819096)

[1.2 编译 4](#_Toc523819097)

[1.3 启动monitor 5](#_Toc523819098)

[1.4 启动osd 5](#_Toc523819099)

[1.5 启动mgr 6](#_Toc523819100)

[1.6 启动mds 6](#_Toc523819101)

[1.7 启动rgw 7](#_Toc523819102)

[1.8 启动restapi 7](#_Toc523819103)

[1.9 启动rbd 8](#_Toc523819104)

[1.10 设置kv\_store etcd 8](#_Toc523819105)

[**二．配置项** 8](#_Toc523819106)

[2.1 ceph.conf配置 8](#_Toc523819107)

[**三．rpm包构建** 9](#_Toc523819108)

[3.1 tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm构建 9](#_Toc523819109)

[3.2 nfs-ganesha rpm构建 11](#_Toc523819110)

[**四．附录** 13](#_Toc523819111)

[4.1 问题集 13](#_Toc523819112)

[4.2 相关链接 16](#_Toc523819113)

***\* 版本修订记录 \****

|  |  |  |
| --- | --- | --- |
| ***版本号*** | ***修订时间*** | ***修订内容*** |
| *v1.0* | *2018-08-22* | *初版修订* |
| *v1.1* | *2018-09-04* | *增加aarch64 rpm包构建过程* |
|  |  |  |
|  |  |  |
|  |  |  |

***\* Release Copyleft©free \****

**一．编译启动**

1.1 安装docker

|  |
| --- |
| # 卸载旧版本的docker组件  $ sudo yum remove docker \  docker-client \  docker-client-latest \  docker-common \  docker-latest \  docker-latest-logrotate \  docker-logrotate \  docker-selinux \  docker-engine-selinux \  docker-engine  # 安装依赖包  $ sudo yum install -y yum-utils \  device-mapper-persistent-data \  lvm2  # 添加镜像  $ sudo yum-config-manager \  --add-repo \  https://download.docker.com/linux/centos/docker-ce.repo  # 安装docker  $ sudo yum install docker-ce  #启动docker  $ sudo systemctl start docker |

1.2 编译

|  |
| --- |
| # 编译需要python3环境  [cpu@mon ~]$ sudo yum –y install python36  [cpu@mon ~]$ which python3  # 没有python3时执行  [cpu@mon ~]$ ln –s /usr/bin/python36 /usr/bin/python3  # 下载ceph-container源码  [cpu@mon ~]$ git clone https://github.com/ceph/ceph-container  [cpu@mon ~]$ cd ceph-container  # 去掉arm平台没有的rpm包   * ./ceph-releases/ALL/centos/daemon-base/ \_\_ISCSI\_PACKAGES\_\_文件中去掉tcmu-runner包 * ./src/daemon-base/ \_\_GANESHA\_PACKAGES\_\_文件去掉nfs-ganesha-ceph、nfs-ganesha-rgw两个包   # 编译镜像  [cpu@mon ceph-container]$ make FLAVORS=mimic,centos,7 build  # 查询编译结果  [cpu@mon ceph-container]$ sudo docker images  REPOSITORY TAG IMAGE ID CREATED SIZE  ceph/daemon master-mimic-centos-7-aarch64 1377c0f1c493 22 hours ago 716MB  ceph/daemon-base master-mimic-centos-7-aarch64 c5640eb584de 22 hours ago 584MB  centos 7 a8bed901118d 13 days ago 235MB  [cpu@mon ceph-container]$ |

1.3 启动monitor

|  |
| --- |
| # 启动：  [cpu@mon ~]$ sudo docker run -d --net=host --name mon -v /etc/ceph:/etc/ceph -v /var/lib/ceph/:/var/lib/ceph/ -e MON\_IP=192.168.0.20 -e CEPH\_PUBLIC\_NETWORK=192.168.0.0/24 ceph/daemon mon  # 可设置的环境变量有:   * MON\_NAME: Monitor的ID，默认为hostname * CEPH\_PUBLIC\_NETWORK:配置集群公有网络的IP地址 * CEPH\_CLUSTER\_NETWORK:配置集群网络的IP地址 * MON\_IP: 运行docker容器的主机IP地址 * NETWORK\_AUTO\_DETECT:在没有--net=host的情况下，是否自动获取IP地址； * 0:表示不自动检测IP地址，这是默认值(MON\_IP,CEPH\_PUBLIC\_NETWORK) * 1:自动检测IPv6地址，没有IPv6地址的情况下自动检测IPv4地址 * 4:只检测IPv4地址 * 6:只检测IPv6地址 |
|  |

1.4 启动osd

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  b.使用ceph auth get client.bootstrap-osd –o $OSD\_BOOTSTRAP\_KEYRING导出keyring到相应的/var/lib/ceph/osd/{ CLUSTER }-{ OSD-ID }目录下  # 启动  [cpu@mon ~]$ sudo docker run -d --net=host \  --pid=host \  --privileged=true \  -v /etc/ceph:/etc/ceph \  -v /var/lib/ceph/:/var/lib/ceph/ \  -v /dev/:/dev/ \  -e OSD\_DEVICE=/dev/sdb \  -e OSD\_BLUESTORE=1 \  ceph/daemon osd  # 可设置的环境变量有:   * JOURNAL\_DIR:设置日志目录，最终日志目录为：${JOURNAL\_DIR}/journal.${OSD\_ID}，没有该变量时查找JOURNAL变量； * JOURNAL:设置日志路径，最终日志目录为：${JOURNAL}；为设置该变量时日志目录为：/var/lib/ceph/osd/{ CLUSTER }-{ OSD-ID}/journal目录； * OSD\_DEVICE:指定OSD的数据设备目录 * OSD\_JOURNAL:仅在OSD\_TYPE为disk和和activate的时候使用； * OSD\_TYPE: * directory:将 /var/lib/ceph/osd作为osd的数据目录创建OSD，默认OSD磁盘挂载在目录上，可通过-v <host\_osd\_data\_dir>:/var/lib/ceph/osd进行设置；无需使用环境变量OSD\_DEVICE，并且在OSD\_TYPE不设置的时候禁止提供该变量，会被作为自动判断OSD\_TYPE的标识； * directory\_single:判断/var/lib/ceph/osd/{ CLUSTER }-{ OSD-ID}/fsid没有被锁的情况下启动OSD * disk:使用OSD\_DEVICE指定的设备prepare并activate， * prepare:prepare OSD\_DEVICE * activate:activate OSD\_DEVICE * devices:使用-e OSD\_DISKS=0:sdd 1:sde 2:sdf设置的环境变量来提供已经prepare和activate的OSD设备； * activate\_journal:为环境变量OSD\_JOURNAL的设备激活日志 * 没有设置该变量:如果没有设置OSD\_DEVICE变量，这时directory;如果在OSD\_DEVICE中发现了ceph data，这只需要activate即可，其他形式为disk类型； * OSD\_BLUESTORE:和OSD\_FILESTORE互为相反，1：表示安装BLUESTORE,同时会自动设置FILESTORE为0，默认值为1 * OSD\_FILESTORE:和OSD\_BLUESTORE互为相反，1：表示安装FILESTORE,同时会自动设置BLUESTORE为0，默认值为0 * OSD\_DMCRYPT:是否对OSD数据目录使用dm-crypt算法进行加密，1：加密，0：不加密；默认为0 |

1.5 启动mgr

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  #启动  docker run -d --net=host \  -v /etc/ceph:/etc/ceph \  -v /var/lib/ceph/:/var/lib/ceph/ \  ceph/daemon mgr |

1.6 启动mds

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  #启动  docker run -d --net=host \  -v /var/lib/ceph/:/var/lib/ceph/ \  -v /etc/ceph:/etc/ceph \  -e CEPHFS\_CREATE=1 \  ceph/daemon mds  # 可设置的环境变量：   * MDS\_NAME:指定不同的MDS服务器的名称，在同一个容器部署多个MDS时，可以使用该变量为不同的服务指定不同的名称； * CEPHFS\_CREATE:是否创建文件系统，默认为0，不创建； * CEPHFS\_NAME:创建的文件系统名称，默认为cephfs; * CEPHFS\_DATA\_POOL:文件系统的数据池，不存在则创建，默认为{ CEPHFS\_NAME }\_data; * CEPHFS\_DATA\_POOL\_PG:设置文件系统数据池的PG数量，默认为8； * CEPHFS\_METADATA\_POOL:设置文件系统元数据池，不存在则创建，默认为{ CEPHFS\_NAME }\_metadata; * CEPHFS\_METADATA\_POOL\_PG:设置元数据池PG的数量，默认为8； |

1.7 启动rgw

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  #启动  docker run -d --net=host \  -v /var/lib/ceph/:/var/lib/ceph/ \  -v /etc/ceph:/etc/ceph \  ceph/daemon rgw  # 可设置的环境变量：   * RGW\_CIVETWEB\_PORT:设置civetwe监听端口，默认8080 * RGW\_NAME:设置RGW的client用户名称，默认为HOSTNAME * RGW\_ZONEGROUP:默认为空值 * RGW\_ZONE:默认为空值 |

1.8 启动restapi

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  b.导出ceph.client.admin.keyring  #启动  docker run -d --net=host \  ceph/daemon restapi  # 可设置的环境变量：   * RESTAPI\_IP:默认监听的IP地址，默认0.0.0.0 * RESTAPI\_PORT:默认监听的端口，默认5000 * RESTAPI\_BASE\_URL:URL地址上下文，默认/api/v0.1 * RESTAPI\_LOG\_LEVEL:日志级别，默认warning * RESTAPI\_LOG\_FILE:日志文件设置，默认/var/log/ceph/ceph-restapi.log |

1.9 启动rbd

|  |
| --- |
| #导出配置文件  a.从monitor集群导出ceph.conf文件并复制到osd /etc/ceph/目录下  #启动  docker run -d --net=host \  ceph/daemon rbd\_mirro |

1.10 设置kv\_store etcd

|  |
| --- |
| # 使用ceph.defaults文件设置键值对，可以将相应的参数放到该文件中，该文件位于./ceph-container/src/daemon/目录下  docker run -d --net=host \  -e KV\_TYPE=etcd \  -e KV\_IP=127.0.0.1 \  -e KV\_PORT=2379 \  ceph/daemon populate\_kvstore  # 如果配置需要使用etcd，则可以在所有的启动命令中添加以下两个环境变量:  -e KV\_TYPE=etcd  -e KV\_IP=192.168.0.20 |

**二．配置项**

2.1 ceph.conf配置

|  |
| --- |
| # 不使用etcd  编辑ceph-container/src/daemon/config.static.sh  # 使用etcd  编辑ceph-container/src/daemon/ceph.defaults |

**三．rpm包构建**

3.1 tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm构建

|  |
| --- |
| # 安装cmake  [cpu@mon tcmu-runner-rpm]$ sudo yum -y install cmake3  [cpu@mon tcmu-runner]$ sudo ln -s /usr/bin/cmake3 /usr/bin/cmake  # 下载依赖包  [cpu@mon tcmu-runner-rpm]$ sudo yum -y install libnl3 libnl3-devel zlib zlib-devel glib2-devel kmod kmod-devel glusterfs-api-devel librbd1 librbd1-devel  # 下载源码  [cpu@mon tcmu-runner-rpm]$ git clone git://www.github.com/open-iscsi/tcmu-runner.git  # 编译  [cpu@mon tcmu-runner-rpm]$ cd tcmu-runner/  [cpu@mon tcmu-runner]$ cmake .  -- The C compiler identification is GNU 4.8.5  -- Check for working C compiler: /usr/bin/cc  -- Check for working C compiler: /usr/bin/cc -- works  -- Detecting C compiler ABI info  -- Detecting C compiler ABI info - done  -- Detecting C compile features  -- Detecting C compile features - done  -- Found PkgConfig: /usr/bin/pkg-config (found version "0.27.1")  -- Checking for module 'gio-unix-2.0'  -- Found gio-unix-2.0, version 2.54.2  -- Checking for module 'libkmod'  -- Found libkmod, version 20  -- Found ZLIB: /usr/lib64/libz.so (found version "1.2.7")  -- Looking for linux/falloc.h  -- Looking for linux/falloc.h - found  -- Configuring done  -- Generating done  -- Build files have been written to: /home/cpu/tcmu-runner-rpm/tcmu-runner  [cpu@mon tcmu-runner]$ make all  [cpu@mon tcmu-runner]$ ll \*.so  -rwxrwxr-x 1 cpu cpu 74944 Aug 30 10:14 handler\_file\_optical.so  -rwxrwxr-x 1 cpu cpu 71744 Aug 30 10:14 handler\_file.so  -rwxrwxr-x 1 cpu cpu 75480 Aug 30 10:14 handler\_file\_zbc.so  -rwxrwxr-x 1 cpu cpu 74208 Aug 30 10:14 handler\_glfs.so  -rwxrwxr-x 1 cpu cpu 75872 Aug 30 10:14 handler\_qcow.so  -rwxrwxr-x 1 cpu cpu 76432 Aug 30 10:14 handler\_rbd.so  lrwxrwxrwx 1 cpu cpu 12 Aug 30 10:14 libtcmu.so -> libtcmu.so.2  # 安装rpmbuild工具  [cpu@mon ~]$ sudo yum -y install rpm-build  [cpu@mon ~]$ sudo yum -y install rpmdevtools  [cpu@mon ~]$ sudo yum -y install rpmlint  # 编辑rpmbuild配置文件，使用普通用户构建rpm包，禁止使用root用户  [cpu@mon ~]$ vim .rpmmacros  %\_topdir /home/cpu/tcmu-runner-rpm/aarch64 # 添加rpm包编译的路径  [cpu@mon aarch64]$ rpmbuild --showrc | grep topdir #查看添加的编译目录  -14: \_builddir %{\_topdir}/BUILD  -14: \_buildrootdir %{\_topdir}/BUILDROOT  -14: \_rpmdir %{\_topdir}/RPMS  -14: \_sourcedir %{\_topdir}/SOURCES  -14: \_specdir %{\_topdir}/SPECS  -14: \_srcrpmdir %{\_topdir}/SRPMS  -14: \_topdir /home/cpu/tcmu-runner-rpm/aarch64  [cpu@mon ~]$ cd /home/cpu/tcmu-runner-rpm/aarch64  [cpu@mon aarch64]$ rpmdev-setuptree  [cpu@mon aarch64]$ ls #使用rpmdev-setuptree命令时不创建BUILDROOT目录  BUILD RPMS SOURCES SPECS SRPMS  [cpu@mon aarch64]$ cd SOURCES  [cpu@mon SOURCES]$ git clone git://www.github.com/open-iscsi/tcmu-runner.git # 下载源码到SOURCE文件夹  [cpu@mon SOURCES]$ tar -zcvf tcmu-runner.tar.gz tcmu-runner/  [cpu@mon SOURCES]$ rm -rf tcmu-runner/  [cpu@mon SOURCES]$ cd ../SPECS/  [cpu@mon SPECS]$ rpmdev-newspec tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm.spec # 新增spce文件  [cpu@mon SPECS]$ ls  tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm.spec  [cpu@mon SPECS]$ vim tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm.spec # 编辑文件  Name: tcmu-runner  Version: 1.4.0  Release: 0.1.51.geef5115%{?dist}  Summary: tcmu-runner for aarch64  License: LGPLv2.1 or Apache License 2.0  URL: https://github.com/open-iscsi/tcmu-runner  Source0: /home/cpu/tcmu-runner-rpm/aarch64/SOURCES/tcmu-runner.tar.gz  %description  tcmu-runner is a daemon that handles the userspace side of the LIO TCM-User backstore.  %prep  %setup -c  %build  %cmake .  %make\_build  %install  %make\_install  mkdir -p $RPM\_BUILD\_ROOT/usr/lib/systemd/system  cp tcmu-runner.service $RPM\_BUILD\_ROOT/usr/lib/systemd/system  %files  %doc README.md LICENSE.LGPLv2.1 LICENSE.Apache2  %{\_sysconfdir}/dbus-1/system.d/tcmu-runner.conf  %{\_sysconfdir}/tcmu/tcmu.conf  %{\_bindir}/tcmu-runner  %{\_prefix}/lib/systemd/system/tcmu-runner.service  %{\_libdir}/libtcmu.so.2  %{\_libdir}/tcmu-runner/handler\_file\_zbc.so  %{\_libdir}/tcmu-runner/handler\_glfs.so  %{\_libdir}/tcmu-runner/handler\_qcow.so  %{\_libdir}/tcmu-runner/handler\_rbd.so  %{\_datarootdir}/dbus-1/system-services/org.kernel.TCMUService1.service  %{\_mandir}/man8/tcmu-runner.8.gz  %changelog  # 检查spec文件是否有误  [cpu@mon SPECS]$ rpmlint tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm.spec  0 packages and 1 specfiles checked; 3 errors, 0 warnings. # 注意E: hardcoded-library-path in可以忽略  # 构建rpm包  [cpu@mon SPECS]$ rpmbuild -ba tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm.spec  # 查看构建完成的rpm包  [cpu@mon SPECS]$ cd /home/cpu/tcmu-runner-rpm/aarch64/RPMS/aarch64  [cpu@mon aarch64]$ ls  tcmu-runner-1.4.0-0.1.51.geef5115.el7.aarch64.rpm tcmu-runner-debuginfo-1.4.0-0.1.51.geef5115.el7.aarch64.rpm  [cpu@mon aarch64]$ |

3.2 nfs-ganesha rpm构建

|  |
| --- |
| # [cpu@mon nfs-ganesha-rpm]$ git clone https://github.com/nfs-ganesha/nfs-ganesha.git  # 安装依赖包  [cpu@mon src]$ sudo yum -y install krb5 krb5-devel bison bison-devel flex flex-devel xfsprogs-devel  # 下载x86的ceph版本为mimic的rpm包  [cpu@mon mimic]$ wget http://download.ceph.com/nfs-ganesha/rpm-V2.6-stable/mimic/SRPMS/nfs-ganesha-2.6.2-0.1.el7.src.rpm  [cpu@mon mimic]$ ls  nfs-ganesha-2.6.2-0.1.el7.src.rpm  # 将rpm包使用cpio解压  [cpu@mon mimic]$ rpm2cpio nfs-ganesha-2.6.2-0.1.el7.src.rpm | cpio -idv  nfs-ganesha-2.6.2.tar.gz  nfs-ganesha.spec  4026 blocks  [cpu@mon mimic]$ ls  nfs-ganesha-2.6.2-0.1.el7.src.rpm nfs-ganesha-2.6.2.tar.gz nfs-ganesha.spec  # 新建aarch64的工作目录  [cpu@mon nfs-ganesha-rpm]$ mkdir -p aarch64/mimic/  [cpu@mon nfs-ganesha-rpm]$ cd aarch64/mimic/  [cpu@mon mimic]$ ls  [cpu@mon mimic]$ pwd  /home/cpu/nfs-ganesha-rpm/aarch64/mimic  # 添加rpm构建的topdir  [cpu@mon mimic]$ vim ~/.rpmmacros  %\_topdir /home/cpu/nfs-ganesha-rpm/aarch64/mimic # 文件开始处添加该行  *NOTE*:存在多个\_topdir时以最后一个为准  # 查看是否添加成功  cpu@mon mimic]$ rpmbuild --showrc | grep topdir  -14: \_builddir %{\_topdir}/BUILD  -14: \_buildrootdir %{\_topdir}/BUILDROOT  -14: \_rpmdir %{\_topdir}/RPMS  -14: \_sourcedir %{\_topdir}/SOURCES  -14: \_specdir %{\_topdir}/SPECS  -14: \_srcrpmdir %{\_topdir}/SRPMS  -14: \_topdir /home/cpu/nfs-ganesha-rpm/aarch64/mimic  # 初始化目录  [cpu@mon mimic]$ rpmdev-setuptree  [cpu@mon mimic]$ ls  BUILD RPMS SOURCES SPECS SRPMS  # 复制x86的文件到相应的文件下  [cpu@mon mimic]$ cp ~/nfs-ganesha-rpm/x86\_64/mimic/nfs-ganesha.spec SPECS/  [cpu@mon mimic]$ cp ~/nfs-ganesha-rpm/x86\_64/mimic/nfs-ganesha-2.6.2.tar.gz SOURCES/  [cpu@mon mimic]$ tree ./  ./  ├── BUILD  ├── RPMS  ├── SOURCES  │   └── nfs-ganesha-2.6.2.tar.gz  ├── SPECS  │   └── nfs-ganesha.spec  └── SRPMS  5 directories, 2 files  # 构建rpm包  [cpu@mon mimic]$ cd SPECS/  [cpu@mon SPECS]$ ls  nfs-ganesha.spec  [cpu@mon SPECS]$ rpmbuild -ba nfs-ganesha.spec  # 查看编译结果  [cpu@mon SPECS]$ cd ..  [cpu@mon mimic]$ ll RPMS/aarch64/  total 4368  -rw-rw-r-- 1 cpu cpu 107012 Sep 4 09:29 libntirpc-1.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 83404 Sep 4 09:29 libntirpc-devel-1.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 553044 Sep 4 09:29 nfs-ganesha-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 26672 Sep 4 09:29 nfs-ganesha-ceph-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 3445136 Sep 4 09:29 nfs-ganesha-debuginfo-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 49076 Sep 4 09:29 nfs-ganesha-gpfs-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 22816 Sep 4 09:29 nfs-ganesha-mem-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 5332 Sep 4 09:29 nfs-ganesha-mount-9P-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 12360 Sep 4 09:29 nfs-ganesha-nullfs-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 38884 Sep 4 09:29 nfs-ganesha-proxy-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 19824 Sep 4 09:29 nfs-ganesha-rgw-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 38676 Sep 4 09:29 nfs-ganesha-vfs-2.6.2-0.1.el7.aarch64.rpm  -rw-rw-r-- 1 cpu cpu 38036 Sep 4 09:29 nfs-ganesha-xfs-2.6.2-0.1.el7.aarch64.rpm  # 以同样的方式构建luminous版本的rpm包 |

**四．附录**

4.1 问题集

|  |
| --- |
| **问题1：**  [cpu@mon ceph-container]$ make FLAVORS=mimic,centos,7 build  /usr/bin/env: python3: No such file or directory  make: \*\*\* [stage.aarch64,mimic,centos,7] Error 127  解决：建立软连接(python3->python36)  [cpu@mon ceph-container]$ sudo ln -s /usr/bin/python36 /usr/bin/python3 |
| **问题2：**  make[1]: Entering directory `/home/cpu/ceph-container/staging/mimic-centos-7-aarch64/daemon-base'  === docker build ceph/daemon-base:master-mimic-centos-7-aarch64  ERRO[0000] failed to dial gRPC: cannot connect to the Docker daemon. Is 'docker daemon' running on this host?: dial unix /var/run/docker.sock: connect: permission denied  解决：命令加sudo  **问题3：**  The command '/bin/sh -c bash -c ' if [ -n "nfs-ganesha nfs-ganesha-ceph nfs-ganesha-rgw" ]; then echo "[ganesha]" > /etc/yum.repos.d/ganesha.repo ; echo "name=ganesha" >> /etc/yum.repos.d/ganesha.repo ; echo "baseurl=http://download.ceph.com/nfs-ganesha/rpm-V2.6-stable/$CEPH\_VERSION/\$basearch/" >> /etc/yum.repos.d/ganesha.repo ; echo "gpgcheck=0" >> /etc/yum.repos.d/ganesha.repo ; echo "enabled=1" >> /etc/yum.repos.d/ganesha.repo ; fi && if [ -n "tcmu-runner ceph-iscsi-config ceph-iscsi-cli python-rtslib" ]; then for repo in ceph-iscsi-config ceph-iscsi-cli tcmu-runner python-rtslib; do curl -L https://shaman.ceph.com/api/repos/$repo/master/latest/centos/7/repo > /etc/yum.repos.d/$repo.repo ; done ; fi' && yum update -y && rpm --import 'https://download.ceph.com/keys/release.asc' && rpm -Uvh http://download.ceph.com/rpm-${CEPH\_VERSION}/el7/noarch/ceph-release-1-1.el7.noarch.rpm && yum install -y epel-release && yum install -y ca-certificates e2fsprogs ceph-common ceph-mon ceph-osd ceph-mds rbd-mirror ceph-mgr kmod lvm2 ceph-radosgw libradosstriper1 nfs-ganesha nfs-ganesha-ceph nfs-ganesha-rgw tcmu-runner ceph-iscsi-config ceph-iscsi-cli python-rtslib && INITIAL\_SIZE="$(bash -c 'sz="$(du -sm --exclude=/proc /)" ; echo "${sz%\*/}"')" && echo 'Postinstall cleanup' && ( rm -rf "/usr/bin/hyperkube /usr/bin/etcd /usr/bin/systemd-analyze /usr/share/hwdata/{iab.txt,oui.txt}" && yum clean all && rpmdb --rebuilddb && rpm -q ca-certificates e2fsprogs ceph-common ceph-mon ceph-osd ceph-mds rbd-mirror ceph-mgr kmod lvm2 ceph-radosgw libradosstriper1 nfs-ganesha nfs-ganesha-ceph nfs-ganesha-rgw tcmu-runner ceph-iscsi-config ceph-iscsi-cli python-rtslib) && sed -i -e 's/udev\_rules = 1/udev\_rules = 0/' -e 's/udev\_sync = 1/udev\_sync = 0/' /etc/lvm/lvm.conf && grep -sqo "udev\_sync = 0" /etc/lvm/lvm.conf && grep -sqo "udev\_rules = 0" /etc/lvm/lvm.conf && rm -rf /etc/{selinux,systemd,udev} /lib/{lsb,udev} /tmp/\* /usr/lib{,64}/{locale,systemd,udev,dracut} /usr/share/{doc,info,locale,man} /usr/share/{bash-completion,pkgconfig/bash-completion.pc} /var/log/\* /var/tmp/\* && find / -xdev -name "\*.pyc" -o -name "\*.pyo" -exec rm -f {} \; && if [ -f /usr/bin/ceph-dencoder ]; then gzip -9 /usr/bin/ceph-dencoder; fi && bash -c ' function ifstrip () { if compgen -g "$1"; then strip -s "$1"; fi } && ifstrip /usr/lib{,64}/ceph/erasure-code/\* && ifstrip /usr/lib{,64}/rados-classes/\* && ifstrip /usr/lib{,64}/python\*/{dist,site}-packages/{rados,rbd,rgw}.\*.so && ifstrip /usr/bin/{crushtool,monmaptool,osdmaptool}' && rm -f /usr/lib/ceph/mgr/dashboard/static/AdminLTE-\*/plugins/datatables/extensions/TableTools/images/psd/\* && find /var/log/ -type f -exec truncate -s 0 {} \; && FINAL\_SIZE="$(bash -c 'sz="$(du -sm --exclude=/proc /)" ; echo "${sz%\*/}"')" && REMOVED\_SIZE=$((INITIAL\_SIZE - FINAL\_SIZE)) && echo "Cleaning process removed ${REMOVED\_SIZE}MB" && echo "Dropped container size from ${INITIAL\_SIZE}MB to ${FINAL\_SIZE}MB" && rpm -q ca-certificates e2fsprogs ceph-common ceph-mon ceph-osd ceph-mds rbd-mirror ceph-mgr kmod lvm2 ceph-radosgw libradosstriper1 nfs-ganesha nfs-ganesha-ceph nfs-ganesha-rgw tcmu-runner ceph-iscsi-config ceph-iscsi-cli python-rtslib && echo 'Packages verified successfully'' returned a non-zero code: 3  make[1]: \*\*\* [build] Error 3  make[1]: Leaving directory `/home/cpu/ceph-container/staging/mimic-centos-7-aarch64/daemon-base'  make: \*\*\* [daemon-base.aarch64,mimic,centos,7] Error 2  解决：  tcmu-runner、nfs-ganesha-ceph、nfs-ganesha-rgw无法安装；  方案一：去掉不能安装的rpm包  ./ceph-releases/ALL/centos/daemon-base/ \_\_ISCSI\_PACKAGES\_\_文件中去掉tcmu-runner包  ./src/daemon-base/ \_\_GANESHA\_PACKAGES\_\_文件去掉nfs-ganesha-ceph、nfs-ganesha-rgw两个包  方案二：打包生成符合arm架构的rpm包  **tcmu**: https://github.com/open-iscsi/tcmu-runner  **nfs\_ganesha**: https://github.com/nfs-ganesha/nfs-ganesha  **问题4：(不影响)**  Delta RPMs disabled because /usr/bin/applydeltarpm not installed  **问题5:（不影响）**  Public key for centos-release-7-5.1804.4.el7.centos.a.aarch64.rpm is not installed  **问题6：（不影响）**  Warning: RPMDB altered outside of yum.  **问题7：（不影响）**  https://mirrors.tongji.edu.cn/epel/7/aarch64/repodata/32aa505750af1bfcc036ef54b98377859ce93a881724c103b3dbe41b6622ede9-updateinfo.xml.bz2: [Errno 14] HTTPS Error 404 - Not Found  **问题8：**  CMake Error: The following variables are used in this project, but they are set to NOTFOUND.  Please set them or make sure they are set and tested correctly in the CMake files:  /home/cpu/tcmu-runner-rpm/tcmu-runner/LIBNL\_INCLUDE\_DIR  used as include directory in directory /home/cpu/tcmu-runner-rpm/tcmu-runner  used as include directory in directory /home/cpu/tcmu-runner-rpm/tcmu-runner  LIBNL\_GENL\_LIB  linked by target "tcmu" in directory /home/cpu/tcmu-runner-rpm/tcmu-runner  LIBNL\_LIB  linked by target "tcmu" in directory /home/cpu/tcmu-runner-rpm/tcmu-runner  -- Configuring incomplete, errors occurred!  See also "/home/cpu/tcmu-runner-rpm/tcmu-runner/CMakeFiles/CMakeOutput.log".  解决：安装libnl-devel包  [cpu@mon ~]$ sudo yum -y install libnl3-devel  **问题9：**  CMake Error at CMakeLists.txt:492 (message):  Cannot find kerberos libraries  解决：  [cpu@mon src]$ sudo yum -y install krb5-devel |

4.2 相关链接

*【1】https://hub.docker.com/r/ceph/daemon/*

*【2】https://github.com/ceph/ceph-container*