****

**Ceph Ansible(***docker***)**

目录

[**一．部署** 5](#_Toc524351255)

[1.1 ansible安装 5](#_Toc524351256)

[1.2 下载ceph-ansible源码 5](#_Toc524351257)

[1.3 编辑hosts文件 5](#_Toc524351258)

[1.4 添加互信 5](#_Toc524351259)

[1.5 新增inventory文件 6](#_Toc524351260)

[1.6 重命名playbook 7](#_Toc524351261)

[1.7 编辑all变量文件 7](#_Toc524351262)

[1.8 编辑osd变量文件 8](#_Toc524351263)

[1.9 编辑mgr变量文件 10](#_Toc524351264)

[1.10 编辑mds变量文件 10](#_Toc524351265)

[1.11 编辑ceph.conf配置文件 10](#_Toc524351266)

[**二．集成Ceph-Container** 11](#_Toc524351267)

[2.1 添加组 11](#_Toc524351268)

[2.2 添加编译变量 11](#_Toc524351269)

[2.3 修改ceph-docker-common角色中的文件 11](#_Toc524351270)

[2.4 添加需要传输的Dockerfile文件 12](#_Toc524351271)

[2.5 docker registry构建过程 13](#_Toc524351272)

[**三．附录** 14](#_Toc524351273)

[3.1 问题集 14](#_Toc524351274)

[3.2 集成源码 16](#_Toc524351275)

[ ceph\_maker.yml 16](#_Toc524351276)

[ check\_system.yml 18](#_Toc524351277)

[ config\_docker.yml 19](#_Toc524351278)

[ pull\_image.yml 19](#_Toc524351279)

[ push\_image.yml 20](#_Toc524351280)

[ build\_registry.yml 20](#_Toc524351281)

[ check\_go.yml 21](#_Toc524351282)

[ install\_go.yml 22](#_Toc524351283)

[3.3 参考链接 22](#_Toc524351284)

***\* 版本修订记录 \****

|  |  |  |
| --- | --- | --- |
| ***版本号*** | ***修订时间*** | ***修订内容*** |
| *v1.0* | *2018-08-28* | * *初版修订* |
| *v1.1* | *2018-09-10* | * *添加docker registry构建过程* * *添加check\_go.yml、build\_registry.yml、install\_go.yml三个脚本* * *修改ceph\_maker.yml、push\_image.yml脚本* |
|  |  |  |
|  |  |  |
|  |  |  |

***\* Release Copyleft©free \****

**一．部署**

1.1 ansible安装

|  |
| --- |
| # 添加源  [yang @mon ~]$ yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm  # 安装ansible  [yang @mon ~]$ sudo yum install ansible  [yang @mon ~]$ ansible --version  ansible 2.6.2  config file = /etc/ansible/ansible.cfg  configured module search path = [u'/home/cpu/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']  ansible python module location = /usr/lib/python2.7/site-packages/ansible  executable location = /usr/bin/ansible  python version = 2.7.5 (default, Jul 13 2018, 13:21:48) [GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] |

1.2 下载ceph-ansible源码

|  |
| --- |
| # 下载源码  [yang @mon ~]$ git clone https://github.com/ceph/ceph-ansible.git  [yang @mon ~]$ cd ceph-ansible/  # 为了方便修改，切换到dev分支  [yang @mon ceph-ansible]$ git checkout -b dev  Switched to a new branch 'dev'  [yang @mon ceph-ansible]$ git branch  \* dev  master  [yang @mon ceph-ansible]$  # 安装pip  [yang@mon ~]$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py  [yang@mon ~]$ sudo python get-pip.py  # 下载ceph-ansible需要的依赖包  [yang @mon ceph-ansible]$ sudo pip install -r requirements.txt |

1.3 编辑hosts文件

在hosts文件添加相应需要安装ceph的主机解析；

1.4 添加互信

在ansible节点和各主机节点之间添加互信；

|  |
| --- |
| # 编辑~/.ssh/config  Host 别名  Hostname 主机名  Port 端口  User 用户名  # 生成密钥  [yang @mon ~]$ ssh-keygen -t rsa  一直回车即可  # 拷贝公钥  ssh-copy-id 别名 |

1.5 新增inventory文件

|  |
| --- |
| # 新建hosts文件  yang@CephD:/app/ceph-ansible/ceph-ansible$ touch hosts  # 编辑hosts文件，并在相应的组内添加主机  [mons]  [osds]  [mdss]  [mgrs]  [agents]  [rgws]  [nfss]  [restapis]  [rbdmirrors]  [clients]  [iscsigws]  [maker]  # ansible ping测试  [cpu@mon ceph-ansible]$ ansible -i hosts mon -m ping  mon | SUCCESS => {  "changed": false,  "ping": "pong"  } |

1.6 重命名playbook

|  |
| --- |
| [yang @mon ceph-ansible]$ mv site-docker.yml.sample site.yml  [yang @mon ceph-ansible]$ cd group\_vars/  [yang @mon group\_vars]$ mv mons.yml.sample mons.yml  [yang @mon group\_vars]$ mv all.yml.sample all.yml  [yang@admin group\_vars]$ mv osds.yml.sample osds.yml  [yang@admin group\_vars]$ mv mdss.yml.sample mdss.yml |

1.7 编辑all变量文件

|  |
| --- |
| # 编辑all.yml文件，取消下列变量的注释：  19 ceph\_release\_num:  20 dumpling: 0.67  21 emperor: 0.72  22 firefly: 0.80  23 giant: 0.87  24 hammer: 0.94  25 infernalis: 9  26 jewel: 10  27 kraken: 11  28 luminous: 12  29 mimic: 13  30 nautilus: 14  31 dev: 99  45 cluster: ceph # 设置集群的名称  48 mon\_group\_name: mons  49 osd\_group\_name: osds  50 rgw\_group\_name: rgws  51 mds\_group\_name: mdss  52 nfs\_group\_name: nfss  53 restapi\_group\_name: restapis  54 rbdmirror\_group\_name: rbdmirrors  55 client\_group\_name: clients  56 iscsi\_gw\_group\_name: iscsigws  57 mgr\_group\_name: mgrs.  168 ceph\_stable\_release: 'mimic' # 设置当前的稳定的ceph版本  300 cephx: true # 设置是否使用cephx  # 创建MDS服务时会自动创建下面的Pool和FS，可进行相应修改；  360 #cephfs: cephfs # 创建的文件系统名称  361 #cephfs\_data: cephfs\_data # 文件系统的数据池名称  362 #cephfs\_metadata: cephfs\_metadata # 文件系统的元数据名称  363  364 #cephfs\_pools:  365 # - { name: "{{ cephfs\_data }}", pgs: "{{ hostvars[groups[mon\_group\_name][0]]['osd\_pool\_default\_pg\_num'] }}" }  366 # - { name: "{{ cephfs\_metadata }}", pgs: "{{ hostvars[groups[mon\_group\_name][0]]['osd\_pool\_default\_pg\_num'] }}" }  376 osd\_objectstore: bluestore # 选择OSD的后端存储方式，filestore或bluestore可选  524 ceph\_mon\_docker\_interface: "enp0s3" # 设置monitor使用的网卡  525 ceph\_mon\_docker\_subnet: "192.168.10.0/24" # 设置monitor网卡的地址  530 containerized\_deployment: true # 设置使用docker进行ceph部署 |

1.8 编辑osd变量文件

|  |
| --- |
| # 编辑osds.yml文件  28 copy\_admin\_key: true # 拷贝client.admin的keyring  46 devices:  47 - /dev/sdb  63 # osd\_auto\_discovery: true # 启用磁盘自动检测方式  # 当该值为true时，ceph-ansible会忽略上面填写的devices变量，其使用ansible fact中的ansible\_devices变量中的设备列表传递给ceph-disk,ceph-disk会自动判断设备是否为空设备；  69 osd\_scenario: 'collocated' # 配置osd部署方式  # osd部署的方式分为以下三种：   * collocated:该场景下提供devices变量,使用ceph-disk进行osd磁盘格式化； * filestore: ceph data和ceph journal分区都会存储到相同的device上；   [yang@osd1 ~]$ sudo blkid /dev/sdb\*  /dev/sdb: PTTYPE="gpt"  /dev/sdb1: UUID="b0ecd7ef-44d9-4dc3-9065-e302d6568a9d" TYPE="xfs" PARTLABEL="ceph data" PARTUUID="71c88199-4d4b-4c16-a905-445681e6aeec"  /dev/sdb2: PARTLABEL="ceph journal" PARTUUID="d3e77722-5818-48c8-a433-fedbf371eca1"  [yang@osd1 ~]$ sudo lsblk  NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  sdb 8:16 0 8G 0 disk  ├─sdb1 8:17 0 3G 0 part  └─sdb2 8:18 0 5G 0 part  [yang@osd1 ~]$   * bluestore:将磁盘分为四个分区，一个分区用于data,称为ceph data;一个用于ceph block.db,ceph block.wal, ceph block称为 ceph block;   [yang@osd1 ~]$ sudo blkid /dev/sdb\*  /dev/sdb: PTTYPE="gpt"  /dev/sdb1: UUID="f9ea646a-a4e6-4e64-9e9a-ecd65075e0b2" TYPE="xfs" PARTLABEL="ceph data" PARTUUID="a544b966-6ebc-4a3a-b0b3-7108fd69afa0"  /dev/sdb2: PARTLABEL="ceph block" PARTUUID="0474381d-c750-48fb-896e-f5bd4cc44117"  /dev/sdb3: PARTLABEL="ceph block.db" PARTUUID="f1eac1a3-313e-4a6c-9f91-794fe6d1eadf"  /dev/sdb4: PARTLABEL="ceph block.wal" PARTUUID="6438f876-34e5-4467-9608-c0de742213be"  [yang@osd1 ~]$ sudo lsblk  NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  sdb 8:16 0 8G 0 disk  ├─sdb1 8:17 0 100M 0 part  ├─sdb2 8:18 0 6.3G 0 part  ├─sdb3 8:19 0 1G 0 part  └─sdb4 8:20 0 576M 0 part   * non-collocated:该场景下需要提供devices和dedicated\_devices变量，使用ceph-disk进行osd磁盘格式化； * filestore:ceph data存储在devices设备中，ceph journal存储在dedicated\_devices设备中；   [yang@osd1 ~]$ sudo blkid /dev/sd\*  /dev/sdb: PTTYPE="gpt"  /dev/sdb1: UUID="00b599ca-4fb8-4c88-a702-3a2e7bcc65c8" TYPE="xfs" PARTLABEL="ceph data" PARTUUID="96488c55-2d3d-4dce-8f74-2004c0dcbbc7"  /dev/sdc: PTTYPE="gpt"  /dev/sdc1: PARTLABEL="ceph journal" PARTUUID="0e492600-d434-44a1-8792-42129a42e42e"  [yang@osd1 ~]$ sudo lsblk  NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  sdb 8:16 0 8G 0 disk  └─sdb1 8:17 0 8G 0 part  sdc 8:32 0 8G 0 disk  └─sdc1 8:33 0 5G 0 part  sr0 11:0 1 1024M 0 rom   * bluestore:devices中的设备会有两个分区，一个用于block，一个用于data,data的大小只有100M，用于存储Ceph元数据，block将会用于存储所有的数据。dedicated\_devices中的设备将会使用一个分区用于RocksDB,称为block.db,一个分区用于RocksDB WAL,称为block.wal;   [yang@osd1 ~]$ sudo blkid /dev/sd\*  /dev/sdb: PTTYPE="gpt"  /dev/sdb1: UUID="3434b29a-e23e-4b1e-bb3d-58bcca43cb8c" TYPE="xfs" PARTLABEL="ceph data" PARTUUID="9b6ff389-8d1e-4651-b783-565ecff76474"  /dev/sdb2: PARTLABEL="ceph block" PARTUUID="3f23d9f0-c79b-49b3-93fa-17581e13b2a0"  /dev/sdc: PTTYPE="gpt"  /dev/sdc1: PARTLABEL="ceph block.db" PARTUUID="24d57709-926f-471a-ab0f-b54d719f2f65"  /dev/sdc2: PARTLABEL="ceph block.wal" PARTUUID="33666c89-accd-4ea5-89ac-d0fb56f9f73d"  [yang@osd1 ~]$ sudo lsblk  NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  sdb 8:16 0 8G 0 disk  ├─sdb1 8:17 0 100M 0 part  └─sdb2 8:18 0 7.9G 0 part  sdc 8:32 0 8G 0 disk  ├─sdc1 8:33 0 1G 0 part  └─sdc2 8:34 0 576M 0 part   * lvm:使用lvm\_volumes变量进行配置； * filestore:lvm\_volumes列表中的每个字典内必须包括data,journal和vg\_name这个key,value为名称而不是路径；data可以时一个逻辑卷、设备或分区，journal要么是一个lv，或是一个分区，data\_vg必须是data的lv的vg名称，journal\_vg必须是journal的lv的vg的名称； * bluestore:lvm\_volumes列表中的么个字典必须包含至少data这个键，如果有wal和db,就必须有其对应的卷组VG的名称；data, data\_vg, wal, wal\_vg, db, db\_vg；   117 # dedicated\_devices: 配置non-collocated场景下的日志磁盘  118 # - /dev/sdc |

1.9 编辑mgr变量文件

|  |
| --- |
| 17 copy\_admin\_key: true # 复制admin.keyring |

1.10 编辑mds变量文件

|  |
| --- |
| 20 copy\_admin\_key: true # 复制admin.keyring |

1.11 编辑ceph.conf配置文件

|  |
| --- |
| # 在下列模板文件中添加配置项即可  ./ceph-ansible/roles/ceph-config/templates/ceph.conf.j2  # 在下列变量中添加相应的配置  ceph\_conf\_overrides |

**二．集成Ceph-Container**

2.1 添加组

在inventroy文件中添加maker组，用于在该组的机器上安装本地镜像仓库，编译镜像并上传，其他机器从该机器上获取镜像。

|  |
| --- |
| [maker]  maker1 |

2.2 添加编译变量

|  |
| --- |
| # 编辑all.yml文本中的变量  # ############## #  # CEPH-CONTAINER #  # ############## #  # how to get ceph/daemon  # valid value: make,fetch  get\_image\_method: 'make' # 选择make的时候会使用下面ceph\_container\_repo变量提供的地址下载ceph-container的源码并自动编译ceph镜像，上传到自动设置的本地仓库后其他的主机从该本地仓库中拉去镜像；选择fetch的时候安装ceph-ansbiel原本使用docker安装的逻辑；  # ceph-container git address  ceph\_container\_repo: 'https://github.com/ceph/ceph-container.git'  # make parameters  ceph\_make\_params:  FLAVORS: 'mimic,centos,7'  target: 'build'  maker\_ip\_address: '192.168.10.39' |

2.3 修改ceph-docker-common角色中的文件

|  |
| --- |
| [yang@admin ceph-ansible]$ cd ./ceph-ansible/roles/ceph-docker-common/tasks/  # 编辑fetch\_image.yml  179 - name: "pulling {{ ceph\_docker\_registry}}/{{ ceph\_docker\_image }}:{{ ceph\_docker\_image\_tag }} image"  180 command: "timeout {{ docker\_pull\_timeout }} docker pull {{ ceph\_docker\_registry}}/{{ ceph\_docker\_image }}:{{ ceph\_docker\_image\_tag }}"  181 changed\_when: false  182 register: docker\_image  183 until: docker\_image.rc == 0  184 retries: "{{ docker\_pull\_retry }}"  185 delay: 10  186 when:  187 - (ceph\_docker\_dev\_image is undefined or not ceph\_docker\_dev\_image)  188 - get\_image\_method == 'fetch'  189  190 - name: include ceph\_maker.yml  191 include: ceph\_maker/ceph\_maker.yml  192 when:  193 - get\_image\_method == 'make'  194 - (inventory\_hostname in groups.get('maker'))  195  196 - name: include config\_docker.yml  197 include: ceph\_maker/config\_docker.yml  198 when:  199 - get\_image\_method == 'make'  200 - (inventory\_hostname not in groups.get('maker'))  201  202 - name: pull image from local registry  203 include: ceph\_maker/pull\_image.yml  204 when:  205 - get\_image\_method == 'make'  # 新增ceph\_maker文件夹后，在此目录添加附录中的源文件  [yang@admin tasks]$ mkdir ceph\_maker |

2.4 添加需要传输的Dockerfile文件

|  |
| --- |
| [cpu@mon files]$ pwd  /home/cpu/ceph-ansible/ceph-ansible-maker/roles/ceph-docker-common/files  [cpu@mon files]$ ls  Dockerfile  [cpu@mon files]$ cat Dockerfile  # Build a minimal distribution container  # FROM alpine  FROM centos:7  RUN set -ex \  && yum -y install ca-certificates apache2-utils  COPY ./registry/registry /bin/registry  COPY ./registry/config-example.yml /etc/docker/registry/config.yml  VOLUME ["/var/lib/registry"]  EXPOSE 5000  COPY docker-entrypoint.sh /entrypoint.sh  ENTRYPOINT ["/entrypoint.sh"]  CMD ["/etc/docker/registry/config.yml"] |

2.5 docker registry构建过程

*Note : 该步骤已使用Ansible脚本自动集成，下面仅作为构建过程的记录*

|  |
| --- |
| # 配置go环境  [gouser@mon ~]$ wget https://dl.google.com/go/go1.11.linux-arm64.tar.gz  [gouser@mon ~]$ tar -zxvf go1.11.linux-arm64.tar.gz  [gouser@mon bin]$ vim ~/.bash\_profile  export PATH=$PATH:/home/gouser/go/bin  export GOPATH=/home/gouser/golang  export GOBIN=$GOPATH/bin  export PATH=$GOBIN:$PATH  [gouser@mon bin]$ source ~/.bash\_profile  [gouser@mon bin]$ go version  go version go1.9.4 linux/arm64  [gouser@mon distribution]$ go get github.com/docker/distribution/registry  [gouser@mon distribution]$ go get github.com/opencontainers/go-digest  # 下载registry镜像构建源码  [cpu@mon images]$ git clone https://github.com/docker/distribution-library-image.git # 构建registry镜像使用的源码  [cpu@mon images]$ git clone https://github.com/docker/distribution.git # 下载registry源码包  # 构建registry二进制文件  [cpu@mon images]$ cd distribution  [cpu@mon distribution]$ make  + bin/registry  + bin/digest  + bin/registry-api-descriptor-template  + binaries  [cpu@mon distribution]$ ls bin/  digest registry registry-api-descriptor-template  [cpu@mon distribution]$  # 构建registry镜像  [cpu@mon distribution]$ cd ../distribution-library-image/  [cpu@mon distribution-library-image]$ vim Dockerfile # 修改基础镜像  # FROM alpine  FROM centos:7  [cpu@mon distribution-library-image]$ cd registry/  [cpu@mon registry]$ cp ../../distribution/bin/registry ./  [cpu@mon registry]$ ls  config-example.yml registry  [cpu@mon registry]$ cd ..  [cpu@mon distribution-library-image]$ sudo docker build .  [cpu@mon distribution-library-image]$ sudo docker images  REPOSITORY TAG IMAGE ID CREATED SIZE  registry latest 4de38d677628 3 days ago 311 MB  [cpu@mon distribution-library-image]$ |

**三．附录**

3.1 问题集

|  |
| --- |
| **问题1：**  Collecting requirements.txt  Could not find a version that satisfies the requirement requirements.txt (from versions: )  No matching distribution found for requirements.txt  You are using pip version 10.0.1, however version 18.0 is available.  You should consider upgrading via the 'pip install --upgrade pip' command.  解决：更新pip  cpu@mon ceph-ansible]$ sudo pip install --upgrade pip  **问题2：**  [WARNING]: log file at /var/log/ansible.log is not writeable and we cannot create it, aborting  解决：创建相应的文件  [cpu@mon ceph-ansible]$ sudo touch /var/log/ansible.log  [cpu@mon log]$ sudo chmod 666 /var/log/ansible.log  **问题3：**  [ERROR]: [mon] Validation failed for variable: ceph\_origin  [ERROR]: [mon] Given value for ceph\_origin: {{ 'repository' if ceph\_rhcs or ceph\_stable or ceph\_dev or ceph\_stable\_uca or ceph\_custom else 'dummy' }}  [ERROR]: [mon] Reason: ceph\_origin must be either 'repository', 'distro' or 'local'  解决：放开注释，选择其中一种安装方式，在docker安装的模式下，不需要该变量  valid\_ceph\_origins:  - repository  - distro  - local  **问题4：**  fatal: [mon]: FAILED! => {"changed": false, "msg": "interface does not exist on mon"}  解决：  all.yml文件中的ceph\_mon\_docker\_interface的变量有问题；  **问题5：**  "msg": "Error: docker-ce conflicts with 2:docker-1.13.1-74.git6e3bb8e.el7.centos.aarch64\n"  解决：有些服务器上已经安装了docker-ce和ansible安装的docker发生了冲突；卸载docker-ce即可  yum remove docker-ce  **问题6：**  fatal: [mon1]: FAILED! => {"attempts": 3, "changed": false, "cmd": ["timeout", "300s", "docker", "pull", "docker.io/ceph/daemon:latest"]  解决：由于网络问题，下载镜像的时候需要更长的时间；修改超时时间即可：  ./ceph-ansible/group\_vars/all.yml修改docker\_pull\_timeout: "300s" 为docker\_pull\_timeout: "3000s"  **问题7：**  fatal: [mon1]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'mon\_group\_name' is undefined  解决：变量mon\_group\_name未取消注释：  ./ceph-ansible/group\_vars/all.yml取消变量的mon\_group\_name: mons的注释；  **问题8：**  [ERROR]: [osd1] Validation failed for variable: item[0]  [ERROR]: [osd1] Reason: -> item[0] key did not match 'osd\_scenario' (required item in schema is missing: osd\_scenario)  解决：变量osd\_scenario未取消注释：  在文件./ceph-ansible/group\_vars/osds.yml中定义osd\_scenario变量；  **问题9：**  atal: [osd1]: FAILED! => {"msg": "The conditional check 'ceph\_release\_num[ceph\_release] < ceph\_release\_num.luminous' failed  解决：变量ceph\_release\_num未取消注释：  ./ ceph-ansible/group\_vars/all.yml取消ceph\_release\_num变量的注释；  **问题10：**  /entrypoint.sh: Timed out while trying to reach out to the Ceph Monitor(s).", "2018-08-23 21:30:10 /entrypoint.sh: Make sure your Ceph monitors are up and running in quorum.", "2018-08-23 21:30:10 /entrypoint.sh: Also verify the validity of client.bootstrap-osd keyring."]}  解决：注意查看是否是防火墙的问题；  [root@mon1 ~]# sudo systemctl status firewalled  [root@mon1 ~]# service firewalld stop  Redirecting to /bin/systemctl stop firewalld.service  [root@mon1 ~]# chkconfig firewalld off  Note: Forwarding request to 'systemctl disable firewalld.service'.  Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.  Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.  **问题11：**  ["Error ENOENT: all mgr daemons do not support module 'status', pass --force to force enablement"]  解决：防火墙导致Monitor连不上MGR，注意查看是否是防火墙的问题；  **问题12：**  fatal: [osd1]: FAILED! => {"changed": false, "cmd": ["docker", "inspect", "51b4a26d8290"], "delta": "0:00:00.059037", "end": "2018-08-26 22:29:05.211417", "msg": "non-zero return code", "rc": 1, "start": "2018-08-26 22:29:05.152380", "stderr": "Error: No such object: 51b4a26d8290", "stderr\_lines": ["Error: No such object: 51b4a26d8290"], "stdout": "[]", "stdout\_lines": ["[]"]}  解决：系统服务会一直重启doker，导致脚本查找出来的container id有问题，导致报错；  [yang@osd1 system]$ sudo systemctl stop ceph-osd@{device-name} |

3.2 集成源码

* ceph\_maker.yml

|  |
| --- |
| ---  - name: include check\_system.yml  include: check\_system.yml  - name: update repositories cache and install make on Debian os family  apt:  name: make  update\_cache: true  when: ansible\_os\_family == 'Debian'  - name: update repositories cache and install make on RedHat os family  yum:  name: make  update\_cache: true  when: ansible\_os\_family == 'RedHat'  - name: install git on Debian os family  apt:  name: git  when: ansible\_os\_family == 'Debian'  - name: install git on RedHat os family  yum:  name: git  when: ansible\_os\_family == 'RedHat'  - name: install python3 on Debian os family  apt:  name: python3  when: ansible\_os\_family == 'Debian'  - name: install python3 on RedHat os family  yum:  name: python36  when: ansible\_os\_family == 'RedHat'  - name: find location of python36 command  command: which python36  register: python3\_location  - name: print python36 location  debug:  msg: '{{ python3\_location }}'  - name: create a soft link for python36  file:  src: '{{ python3\_location.stdout }}'  dest: '/usr/bin/python3'  state: link  become: true  - name: create git container directory /tmp/ceph-container/  file:  path: '/tmp/ceph-container/'  state: directory  mode: 0777  when: not ceph\_container\_stat.stat.exists|bool  - name: download ceph-container with git  git:  repo: '{{ ceph\_container\_repo }}'  dest: '/tmp/ceph-container/'  update: false  - name: make the cepn/daemon image  make:  chdir: '/tmp/ceph-container/'  target: '{{ ceph\_make\_params.target }}'  params:  FLAVORS: '{{ ceph\_make\_params.FLAVORS }}'  - name: remove ceph/daemon:latest image  docker\_image:  state: absent  name: ceph/daemon  tag: latest  - name: get the ceph/daemon image id  shell: docker image ls ceph/daemon | awk 'NR>1' | awk '{printf $3}'  register: ceph\_image\_id  become: true  - name: retag the ceph/daemon image  shell: 'docker image tag {{ ceph\_image\_id.stdout\_lines[0] }} {{ ceph\_docker\_registry }}/ceph/daemon:latest'  become: true  - name: build registry image for aarch64  include: build\_registry.yml  when: ansible\_architecture == 'aarch64'  - name: include push\_image.yml  include: push\_image.yml |

* check\_system.yml

|  |
| --- |
| ---  - name: get /tmp directory free space  shell: df -m /tmp | awk 'NR>1' | awk '{printf $4}'  register: tmp\_free\_space  - name: check /tmp directory free space  fail:  msg: '/tmp free space is less than 2000m, now is {{ tmp\_free\_space }}'  when: tmp\_free\_space.stdout\_lines[0]|int < 2000  - name: get /opt directory free space  shell: df -m /opt | awk 'NR>1' | awk '{printf $4}'  register: opt\_free\_space  - name: check /opt directory free space  fail:  msg: '/opt free space is less than 2000m, now is {{ opt\_free\_space }}'  when: opt\_free\_space.stdout\_lines[0]|int < 2000  - name: check /tmp/ceph-container directory  stat:  path: '/tmp/ceph-container'  register: ceph\_container\_stat  - name: check Makefile stat  stat:  path: '/tmp/ceph-container/Makefile'  register: makefile\_stat  - name: check ceph image export tar stat  stat:  path: '/tmp/ceph-container/ceph\_daemon.tar'  register: ceph\_daemon\_tar\_stat  - name: check ceph/daemon image tag  shell: docker image ls ceph/daemon:latest | awk 'NR>1' | wc -l  become: true  register: ceph\_retagimage\_stat |

* config\_docker.yml

|  |
| --- |
| ---  - name: create docker local registry http config file  file:  path: '/etc/docker/daemon.json'  state: 'touch'  become: true  - name: config http for local registry(no tls)  shell: echo '{ "insecure-registries":["{{ maker\_ip\_address }}:5000"]}' > /etc/docker/daemon.json  - name: restart docker daemon  systemd:  state: 'restarted'  name: 'docker' |

* pull\_image.yml

|  |
| --- |
| ---  - name: repull ceph image  docker\_image:  name: '{{ maker\_ip\_address }}:5000/docker.io/ceph/daemon'  tag: 'latest'  repository: '{{ maker\_ip\_address }}:5000/docker.io/ceph/daemon'  - name: retag the ceph image  shell: docker tag {{ maker\_ip\_address }}:5000/docker.io/ceph/daemon:latest docker.io/ceph/daemon:latest |

* push\_image.yml

|  |
| --- |
| ---  - name: pull registry  docker\_image:  name: registry  tag: latest  when: ansible\_architecture != 'aarch64'  - name: include config\_docker.yml  include: config\_docker.yml  - name: start the local registry  docker\_container:  name: "registry"  image: "registry:latest"  published\_ports:  - "5000:5000"  volumes:  - "/opt/registry:/var/lig/registry:rw"  - name: tag and push to local registry  docker\_image:  name: '{{ ceph\_docker\_registry }}/ceph/daemon'  tag: 'latest'  repository: '{{ ansible\_default\_ipv4.address }}:5000/docker.io/ceph/daemon'  tls: no  push: yes  - name: repush ceph image to local registry  shell: 'docker push {{ ansible\_default\_ipv4.address }}:5000/docker.io/ceph/daemon'  - name: set\_fact maker\_ip\_address  set\_fact:  maker\_ip\_address: "{{ ansible\_default\_ipv4.address }}" |

* build\_registry.yml

|  |
| --- |
| ---  - name: include check\_go.yml  include: check\_go.yml  - name: create directory for distribution-library-image  file:  path: '/tmp/distribution-library-image'  state: directory  - name: create directory for ditribution  file:  path: '/tmp/distribution/'  state: directory  - name: download distribution-library-image from github  git:  repo: 'https://github.com/docker/distribution-library-image.git'  dest: /tmp/distribution-library-image  update: true  force: true  - name: download distribtion from github  git:  repo: 'https://github.com/docker/distribution.git'  dest: /tmp/distribution/  update: true  - name: build registry binary  make:  chdir: /tmp/distribution  - name: replace registry binary  command: cp /tmp/distribution/bin/registry /tmp/distribution-library-image/registry/  - name: replace distribution-library-image Dockerfile  copy:  src: Dockerfile  dest: /tmp/distribution-library-image/  force: true  - name: build registry image  docker\_image:  path: /tmp/distribution-library-image/  name: registry  tag: latest  push: no |

* check\_go.yml

|  |
| --- |
| ---  - name: check go version  shell: go version  register: goresult  - name: check go installed or not  include: install\_go.yml  when: goresult.rc| bool  - name: install registry code  shell: go get github.com/docker/distribution/registry  - name: install go-digest code  shell: go get github.com/opencontainers/go-digest |

* install\_go.yml

|  |
| --- |
| ---  - name: download go-version.tar.gz  get\_url:  url: https://dl.google.com/go/go1.11.linux-arm64.tar.gz  dest: $HOME/go1.11.linux-arm64.tar.gz  - name: untar the file  shell: tar -zxvf go1.11.linux-arm64.tar.gz  chdir: $HOME  - name: mkdir $HOME/golang as GOPATH  file:  path: $HOME/golang  state: directory  - name: set go environment  shell: |  echo 'export PATH=$PATH:$HOME/go/bin' >> ~/.bash\_profile  echo 'export GOPATH=$HOME/golang' >> ~/.bash\_profile  echo 'export GOBIN=$GOPATH/bin' >> ~/.bash\_profile  echo 'export PATH=$GOBIN:$PATH' >> ~/.bash\_profile  exit 0  - name: active go environment  shell: source ~/.bash\_profile |

3.3 参考链接

【1】*https://github.com/ceph/ceph-ansible*

【2】*http://docs.ceph.com/ceph-ansible/master/*