CEPH-MEDIC – FIND COMMON ISSUES IN CEPH CLUSTERS

ceph-medic is a very simple tool to run against a Ceph cluster to detect common issues that might prevent correct functionality. It requires non-interactive SSH access to accounts that can sudo without a password prompt.

INSTALLATION

ceph-medic supports a couple different installation methods:

OFFICIAL UPSTREAM REPOS

Download official releases of ceph-medic at https://download.ceph.com/ceph-medic

Currently, only rpm repos built for centos 7 are supported.

ceph-medic has dependencies on packages found in EPEL, so EPEL will need to be enabled.

Follow these steps to install a centos 7 repo from download.ceph.com:

- Install the latest rpm repo from download.ceph.com:

```
wget http://download.ceph.com/ceph-medic/latest/rpm/el7/ceph-medic.repo -O /et
```

- Install epel-release:

```
yum install epel-release
```

- Install the gpg key for ceph-medic:

```
wget https://download.ceph.com/keys/release.asc
rpm --import release.asc
```

- Install ceph-medic:

```
yum install ceph-medic
```

- Verify your install:

```
ceph-medic --help
```

SHAMAN REPOS

Every branch pushed to ceph-medic.git gets a rpm repo created and stored in shaman.ceph.com. Currently, only rpm repos built for centos 7 are supported.

Browse https://shaman.ceph.com/repos/ceph-medic to find the available repos.

Note: Shaman repos are only available for 2 weeks before they are automatically deleted. However, there should always be a repo available for the master branch of ceph-medic.

ceph-medic has dependencies on packages found in EPEL, so EPEL will need to be enabled.

Follow these steps to install a centos 7 repo from shaman.ceph.com:

- Install the latest master shaman repo:

```
wget https://shaman.ceph.com/api/repos/ceph-medic/master/latest/centos/7/repo
```

- Install `epel-release`:

```
yum install epel-release
```

- Install `ceph-medic`:

```
yum install ceph-medic
```

- Verify your install:

```
ceph-medic --help
```

GITHUB

You can install directly from the source on github by following these steps:

- Clone the repository:

```
git clone https://github.com/ceph/ceph-medic.git
```

- Change to the `ceph-medic` directory:

```
cd ceph-medic
```

- Create and activate a python virtual environment:

```
virtualenv venv
source venv/bin/activate
```

- Install ceph-medic into the virtual environment:

```
python setup.py install
```

`ceph-medic` should now be installed and available in the virtualenv you just created. Check your installation by running: `ceph-medic --help`

USAGE

The basic usage of `ceph-medic` is to perform checks against a ceph cluster to identify potential issues with it's installation or configuration. To do this you'd run the following command:

```
ceph-medic --inventory /path/to/hosts --ssh-config /path/to/ssh_config check
```

INVENTORY

`ceph-medic` needs to know the nodes that exist in your ceph cluster before it can perform checks. The inventory (or `hosts` file) is a typical Ansible inventory file and will be used to inform `ceph-medic` of the nodes in your cluster and their respective roles. The following standard host groups are supported by `ceph-medic`: `mons`, `osds`, `rgws`, `mdss`, `mgrs` and `clients`. An example `hosts` file would look like:

```
[mons]
mon0
mon1

[osds]
osd0

[mgrs]
mgr0
```

The location of the `hosts` file can be passed into `ceph-medic` by using the `--inventory` cli option. e.g `ceph-medic --inventory /path/to/hosts`

If the `--inventory` option is not defined `ceph-medic` will first look in the current working directory for a file named `hosts`. If that file does not exist it will look for `/etc/ansible/hosts` to be used as the inventory.

SSH CONFIG

All nodes in your `hosts` file must be configured to provide non-interactive SSH access to accounts that can `sudo` without a password prompt.

Note: This is the same ssh config required by ansible. If you've used `ceph-ansible` to deploy your cluster then your nodes are most likely already configured for this type of ssh access. If that is the case, using the same user that was performed the initial deployment would be easiest.

To provide your ssh config you must use the `--ssh-config` flag and give it a path to a file that defines your ssh configuration. For example, a file like this is used to connect with a cluster comprised of vagrant vms:

```
Host mon0
  HostName 127.0.0.1
  User vagrant
  Port 2200
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  PasswordAuthentication no
  IdentityFile /Users/andrewschoen/.vagrant.d/insecure_private_key
  IdentitiesOnly yes
  LogLevel FATAL

Host osd0
  HostName 127.0.0.1
  User vagrant
  Port 2201
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  PasswordAuthentication no
  IdentityFile /Users/andrewschoen/.vagrant.d/insecure_private_key
  IdentitiesOnly yes
  LogLevel FATAL
```

LOGGING

By default `ceph-medic` sends complete logs to the current working directory. This log file is more verbose than the output you see on the terminal. To change where these logs are created modify the default value for `--log-path` in `~/.cephmedic.conf`.

RUNNING CHECKS

To perform checks against your cluster use the check subcommand. This will perform a series of general checks as well as checks specific to each daemon. Sample output from this command will look like:

```
ceph-medic --ssh-config vagrant_ssh_config check
Host: mgr0                  connection: [connected  ]
Host: mon0                  connection: [connected  ]
Host: osd0                  connection: [connected  ]
```

```
Collection completed!

=======================  Starting remote check session  =======================
Version: 0.0.1    Cluster Name: "test"
Total hosts: [3]
OSDs:    1    MONs:    1     Clients:    0
MDSs:    0    RGWs:    0     MGRs:       1

================================================================================

---------- managers ----------
 mgr0

------------ osds ------------
 osd0

------------ mons ------------
 mon0

17 passed, 0 errors, on 4 hosts
```