RBDMAP – MAP RBD DEVICES AT BOOT TIME

SYNOPSIS

rbdmap map
rbdmap unmap

DESCRIPTION

rbdmap is a shell script that automates `rbd map` and `rbd unmap` operations on one or more RBD (RADOS Block Device) images. While the script can be run manually by the system administrator at any time, the principal use case is automatic mapping/mounting of RBD images at boot time (and unmounting/unmapping at shutdown), as triggered by the init system (a systemd unit file, `rbdmap.service` is included with the ceph-common package for this purpose).

The script takes a single argument, which can be either “map” or “unmap”. In either case, the script parses a configuration file (defaults to `/etc/ceph/rbdmap`, but can be overridden via an environment variable `RBDMAPFILE`). Each line of the configuration file corresponds to an RBD image which is to be mapped, or unmapped.

The configuration file format is:

```
IMAGESPEC RBDOPTS
```

where `IMAGESPEC` should be specified as `POOLNAME/IMAGENAME` (the pool name, a forward slash, and the image name), or merely `IMAGENAME`, in which case the `POOLNAME` defaults to “rbd”. `RBDOPTS` is an optional list of parameters to be passed to the underlying `rbd map` command. These parameters and their values should be specified as a comma-separated string:

```
PARAM1=VAL1,PARAM2=VAL2,...,PARAMN=VALN
```

This will cause the script to issue an `rbd map` command like the following:

```
rbd map POOLNAME/IMAGENAME --PARAM1 VAL1 --PARAM2 VAL2
```

(See the `rbd` manpage for a full list of possible options.)

When run as `rbdmap map`, the script parses the configuration file, and for each RBD image specified attempts to first map the image (using the `rbd map` command) and, second, to mount the image.

When run as `rbdmap unmap`, images listed in the configuration file will be unmounted and unmapped.

`rbdmap unmap-all` attempts to unmount and subsequently unmap all currently mapped RBD images, regardless of whether or not they are listed in the configuration file.

If successful, the `rbd map` operation maps the image to a `/dev/rbdX` device, at which point a udev rule is triggered to create a friendly device name symlink, `/dev/rbd/POOLNAME/IMAGENAME`, pointing to the real mapped device.

In order for mounting/unmounting to succeed, the friendly device name must have a corresponding entry in `/etc/fstab`.

When writing `/etc/fstab` entries for RBD images, it’s a good idea to specify the “noauto” (or “nofail”) mount option. This prevents the init system from trying to mount the device too early - before the device in question even exists. (Since `rbdmap.service` executes a shell script, it is typically triggered quite late in the boot sequence.)

EXAMPLES

Example `/etc/ceph/rbdmap` for two RBD images called “bar1” and “bar2”, both in pool “foopool”:

```
foopool/bar1    id=admin,keyring=/etc/ceph/ceph.client.admin.keyring
foopool/bar2    id=admin,keyring=/etc/ceph/ceph.client.admin.keyring
```

Each line in the file contains two strings: the image spec and the options to be passed to `rbd map`. These two lines get transformed into the following commands:

```
rbd map foopool/bar1 --id admin --keyring /etc/ceph/ceph.client.admin.keyring
rbd map foopool/bar2 --id admin --keyring /etc/ceph/ceph.client.admin.keyring
```

If the images had XFS filesystems on them, the corresponding `/etc/fstab` entries might look like this:

```
/dev/rbd/foopool/bar1 /mnt/bar1 xfs noauto 0 0
/dev/rbd/foopool/bar2 /mnt/bar2 xfs noauto 0 0
```

After creating the images and populating the `/etc/ceph/rbdmap` file, making the images get automatically mapped and mounted at boot is just a matter of enabling that unit:

```
systemctl enable rbdmap.service
```

OPTIONS

None

AVAILABILITY

rbdmap is part of Ceph, a massively scalable, open-source, distributed storage system. Please refer to the Ceph documentation at http://ceph.com/docs for more information.

SEE ALSO

rbd(8),