PREPARE

This subcommand allows a filestore or bluestore setup. It is recommended to pre-provision a logical volume before using it with `ceph-volume lvm`.

Logical volumes are not altered except for adding extra metadata.

> **Note:** This is part of a two step process to deploy an OSD. If looking for a single-call way, please see create

To help identify volumes, the process of preparing a volume (or volumes) to work with Ceph, the tool will assign a few pieces of metadata information using LVM tags.

LVM tags makes volumes easy to discover later, and help identify them as part of a Ceph system, and what role they have (journal, filestore, bluestore, etc...)

Although initially filestore is supported (and supported by default) the back end can be specified with:

- –filestore
- –bluestore

FILESTORE

This is the OSD backend that allows preparation of logical volumes for a filestore objectstore OSD.

It can use a logical volume for the OSD data and a partitioned physical device or logical volume for the journal. No special preparation is needed for these volumes other than following the minimum size requirements for data and journal.

The API call looks like:

```
ceph-volume lvm prepare --filestore --data volume_group/lv_name --journal journal
```

There is flexibility to use a raw device or partition as well for `--data` that will be converted to a logical volume. This is not ideal in all situations since `ceph-volume` is just going to create a unique volume group and a logical volume from that device.

When using logical volumes for `--data`, the value *must* be a volume group name and a logical volume name separated by a `/`. Since logical volume names are not enforced for uniqueness, this prevents using the wrong volume. The `--journal` can be either a logical volume *or* a partition.

When using a partition, it *must* contain a `PARTUUID` discoverable by `blkid`, so that it can later be identified correctly regardless of the device name (or path).

When using a partition, this is how it would look for `/dev/sdc1`:

```
ceph-volume lvm prepare --filestore --data volume_group/lv_name --journal /dev/sdc1
```

For a logical volume, just like for `--data`, a volume group and logical volume name are required:

```
ceph-volume lvm prepare --filestore --data volume_group/lv_name --journal volume_group/journa
```

A generated uuid is used to ask the cluster for a new OSD. These two pieces are crucial for identifying an OSD and will later be used throughout the activate process.

The OSD data directory is created using the following convention:

```
/var/lib/ceph/osd/<cluster name>-<osd id>
```

At this point the data volume is mounted at this location, and the journal volume is linked:

```
ln -s /path/to/journal /var/lib/ceph/osd/<cluster_name>-<osd-id>/journal
```

The monmap is fetched using the bootstrap key from the OSD:

```
/usr/bin/ceph --cluster ceph --name client.bootstrap-osd
--keyring /var/lib/ceph/bootstrap-osd/ceph.keyring
mon getmap -o /var/lib/ceph/osd/<cluster name>-<osd id>/activate.monmap
```

`ceph-osd` will be called to populate the OSD directory, that is already mounted, re-using all the pieces of information from the initial steps:

```
ceph-osd --cluster ceph --mkfs --mkkey -i <osd id> \
--monmap /var/lib/ceph/osd/<cluster name>-<osd id>/activate.monmap --osd-data \
/var/lib/ceph/osd/<cluster name>-<osd id> --osd-journal /var/lib/ceph/osd/<cluster name>-<osd
--osd-uuid <osd uuid> --keyring /var/lib/ceph/osd/<cluster name>-<osd id>/keyring \
--setuser ceph --setgroup ceph
```

PARTITIONING

`ceph-volume lvm` does not currently create partitions from a whole device. If using device partitions the only requirement is that they contain the `PARTUUID` and that it is discoverable by `blkid`. Both `fdisk` and `parted` will create that automatically for a new partition.

For example, using a new, unformatted drive (`/dev/sdd` in this case) we can use `parted` to create a new partition. First we list the device information:

```
$ parted --script /dev/sdd print
Model: VBOX HARDDISK (scsi)
Disk /dev/sdd: 11.5GB
Sector size (logical/physical): 512B/512B
Disk Flags:
```

This device is not even labeled yet, so we can use `parted` to create a `gpt` label before we create a partition, and verify again with `parted print`:

```
$ parted --script /dev/sdd mklabel gpt
$ parted --script /dev/sdd print
Model: VBOX HARDDISK (scsi)
Disk /dev/sdd: 11.5GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
```

Now lets create a single partition, and verify later if `blkid` can find a `PARTUUID` that is needed by `ceph-volume`:

```
$ parted --script /dev/sdd mkpart primary 1 100%
$ blkid /dev/sdd1
/dev/sdd1: PARTLABEL="primary" PARTUUID="16399d72-1e1f-467d-96ee-6fe371a7d0d4"
```

EXISTING OSDS

For existing clusters that want to use this new system and have OSDs that are already running there are a few things to take into account:

Warning: this process will forcefully format the data device, destroying existing data, if any.

- OSD paths should follow this convention:

```
/var/lib/ceph/osd/<cluster name>-<osd id>
```

- Preferably, no other mechanisms to mount the volume should exist, and should be removed (like fstab mount points)

The one time process for an existing OSD, with an ID of 0 and using a "ceph" cluster name would look like (the following command will **destroy any data** in the OSD):

```
ceph-volume lvm prepare --filestore --osd-id 0 --osd-fsid E3D291C1-E7BF-4984-9794-B60D9FA139C
```

The command line tool will not contact the monitor to generate an OSD ID and will format the LVM device in addition to storing the metadata on it so that it can later be startednot contact the monitor to generate an OSD ID and will format the LVM device in addition to storing the metadata on it so that it can later be started (for detailed metadata description see Metadata).

BLUESTORE

The bluestore objectstore is the default for new OSDs. It offers a bit more flexibility for devices. Bluestore supports the following configurations:

- A block device, a block.wal, and a block.db device
- A block device and a block.wal device
- A block device and a block.db device
- A single block device

It can accept a whole device (or partition), or a logical volume for `block`. If a physical device is provided it will then be turned into a logical volume. This allows a simpler approach at using LVM but at the cost of flexibility: there are no options or configurations to change how the LV is created.

The `block` is specified with the `--data` flag, and in its simplest use case it looks like:

```
ceph-volume lvm prepare --bluestore --data vg/lv
```

A raw device can be specified in the same way:

```
ceph-volume lvm prepare --bluestore --data /path/to/device
```

If a `block.db` or a `block.wal` is needed (they are optional for bluestore) they can be specified with `--block.db` and `--block.wal` accordingly. These can be a physical device (they **must** be a partition) or a logical volume.

For both `block.db` and `block.wal` partitions aren't made logical volumes because they can be used as-is. Logical Volumes are also allowed.

While creating the OSD directory, the process will use a `tmpfs` mount to place all the files needed for the OSD. These files are initially created by `ceph-osd --mkfs` and are fully ephemeral.

A symlink is always created for the `block` device, and optionally for `block.db` and `block.wal`. For a cluster with a default name, and an OSD id of 0, the directory could look like:

```
# ls -l /var/lib/ceph/osd/ceph-0
lrwxrwxrwx. 1 ceph ceph 93 Oct 20 13:05 block -> /dev/ceph-be2b6fbd-bcf2-4c51-b35d-a35a162a02
lrwxrwxrwx. 1 ceph ceph 93 Oct 20 13:05 block.db -> /dev/sda1
lrwxrwxrwx. 1 ceph ceph 93 Oct 20 13:05 block.wal -> /dev/ceph/osd-wal-0
-rw-------. 1 ceph ceph 37 Oct 20 13:05 ceph_fsid
-rw-------. 1 ceph ceph 37 Oct 20 13:05 fsid
-rw-------. 1 ceph ceph 55 Oct 20 13:05 keyring
-rw-------. 1 ceph ceph  6 Oct 20 13:05 ready
-rw-------. 1 ceph ceph 10 Oct 20 13:05 type
-rw-------. 1 ceph ceph  2 Oct 20 13:05 whoami
```

In the above case, a device was used for `block` so `ceph-volume` create a volume group and a logical volume using the following convention:

- volume group name: `ceph-{cluster fsid}` or if the vg exists already `ceph-{random uuid}`
- logical volume name: `osd-block-{osd_fsid}`

CRUSH DEVICE CLASS

To set the crush device class for the OSD, use the `--crush-device-class` flag. This will work for both bluestore and filestore OSDs:

```
ceph-volume lvm prepare --bluestore --data vg/lv --crush-device-class foo
```

STORING METADATA

The following tags will get applied as part of the preparation process regardless of the type of volume (journal or data) or OSD objectstore:

- `cluster_fsid`
- `encrypted`
- `osd_fsid`
- `osd_id`
- `crush_device_class`

For filestore these tags will be added:

- `journal_device`
- `journal_uuid`

For bluestore these tags will be added:

- `block_device`
- `block_uuid`
- `db_device`
- `db_uuid`
- `wal_device`
- `wal_uuid`

Note: For the complete lvm tag conventions see Tag API

SUMMARY

To recap the `prepare` process for bluestore:

1. Accept a logical volume for block or a raw device (that will get converted to an lv)
2. Accept partitions or logical volumes for `block.wal` or `block.db`
3. Generate a UUID for the OSD
4. Ask the monitor get an OSD ID reusing the generated UUID
5. OSD data directory is created on a tmpfs mount.
6. `block`, `block.wal`, and `block.db` are symlinked if defined.
7. monmap is fetched for activation
8. Data directory is populated by `ceph-osd`
9. Logical Volumes are are assigned all the Ceph metadata using lvm tags

And the `prepare` process for filestore:

1. Accept only logical volumes for data and journal (both required)
2. Generate a UUID for the OSD
3. Ask the monitor get an OSD ID reusing the generated UUID
4. OSD data directory is created and data volume mounted
5. Journal is symlinked from data volume to journal location
6. monmap is fetched for activation
7. devices is mounted and data directory is populated by `ceph-osd`
8. data and journal volumes are assigned all the Ceph metadata using lvm tags