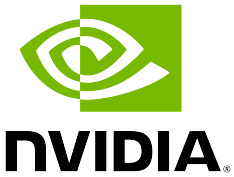

NVIDIA Base Command Manager 10

Upgrade Manual

Revision: 30bbd9219

Date: Wed Nov 12 2025

Table Of Contents

1 Introduction

This *Upgrade Manual* describes the procedure for upgrading a Bright cluster that is being managed by Bright Cluster Manager version 9.2, to a Base Command Manager cluster managed by Base Command Manager version 10.0. [1]

A clean installation of BCM is recommended instead of an upgrade because it is simpler. If an upgrade is to be carried out, then reasonable familiarity with Bright Cluster Manager is assumed, and the *Upgrade Manual* should be read carefully before the upgrade. A good idea is to look up any unfamiliar concepts described in this manual in the other Bright/BCM manuals.

The BCM Support team can be contacted using `https://enterprise-support.nvidia.com/s/create-case` if there are BCM-related upgrade issues for which the documentation is insufficient.

1.1 Supported Bright Versions And Linux Distributions

1.1.1 Upgrades From The Following Bright Versions Are Supported:

- Bright 9.2

1.1.2 The Following Linux Distributions Are Supported:

- RedHat Enterprise Linux 8 (update 8.8 and higher)
- RedHat Enterprise Linux 9 (update 9.2 and higher)
- Rocky Linux 8 (update 8.8 and higher)
- Rocky Linux 9 (update 9.2 and higher)
- SuSE Linux Enterprise Server 15 (SLES15 SP4) and higher
- Ubuntu 20.04
- Ubuntu 22.04

1.1.3 Upgrading The Underlying OS Is Not Supported

Upgrading the underlying OS, that is, upgrading from an earlier major release of the OS upon which BCM runs, to the next major release of the OS (for example, RHEL8 to RHEL9) while keeping BCM as it is, is not supported. If the OS is to be upgraded, then BCM must be reinstalled from scratch.

[1] Just to be clear, Base Command Manager (abbreviated to BCM) is the name given with release 10.0 of the erstwhile Bright Cluster Manager. While Bright Cluster Manager never formally used the abbreviation BCM, it is mildly convenient that it abbreviates to BCM too, since it is an earlier branch of Base Command Manager. The word "Bright" is however gradually being phased out, though it may still show up, for example in URLs, paths, and the manuals.

Exception: DGX OS Upgrade Is Supported, But Not Covered In This Upgrade Manual
A case that can be seen as an exception for upgrading the underlying OS, is upgrading DGX OS 5 to DGX OS 6, or DGX OS 6 to DGX OS 7. DGX OS is a custom OS for the DGX platform. It is based on BCM running on top of Ubuntu. DGX OS package and release upgrades are covered comprehensively in the dedicated DGX OS documentation at:

- `https://docs.nvidia.com/dgx/dgx-os-7-user-guide/upgrading-the-os.html` which covers upgrading an existing DGX OS 5, or 6.
- `https://docs.nvidia.com/dgx/dgx-os-7-user-guide/additional_software.html` which covers upgrading other software on DGX OS 5, or 6.

The latest supported DGX OS version at the time of writing (September 2025) is DGX OS 7.0.

1.2 Parallel Vs In-place Upgrades

There are two kinds of upgrades possible.

- A **parallel upgrade**: This is when a clone is made of the running head node. That clone is kept away from the cluster, and it is the clone that is upgraded. The original head node keeps managing the cluster, so that there is not such a severe pressure on getting the upgrade up before a deadline. This means that if there are unexpected issues that take time to resolve during upgrade, then the original cluster remains unaffected. The head node of the original cluster is replaced with the upgraded head node only after the administrator believes that the upgraded head node is ready.
- An **in-place upgrade**: This is when an upgrade is carried out using the existing head node or head nodes in a production cluster. The regular nodes need to be powered off when the upgrade is carried out.

Chapter 2 covers the steps and considerations for the upgrade procedure that are specific to the parallel upgrade.

Chapter 3 covers the steps and considerations for the upgrade procedure that are specific to the in-place upgrade.

2 Parallel Upgrades

2.1 Introduction

In contrast with the in-place upgrade (Chapter 3), for a parallel upgrade the production cluster remains operational while the upgrade takes place. This is possible with the new `cm-clone-install` and `cm-upgrade` functionality that allows the production (primary) head node to be cloned to a new node. The new node is upgraded (in parallel) outside of the cluster, and when the upgrade is completed, the old production primary head node is replaced with the (newly) upgraded head node. This last step to replace the head node will require a downtime.

Alternatively, the compute nodes can be gradually moved from the old production to the new parallel-upgrade setup, while the two setups co-exist.

2.2 Preparations For The Parallel Upgrade Setup

The following steps need to be carried out for parallel upgrades. The steps marked with HA (High Availability) need to be carried out only on clusters with (head node) HA:

- Have the Bright product key and the MAC address of the spare node (which is to be used for cloning the primary active head node) ready, since they will need to be provided while cloning the primary head node to the spare node.
- (HA) Have also the MAC address of the secondary head node ready.
- Have the new IPs for the cloned primary head node on the external network(s) ready.
- (HA) The primary head node must be made the active head, if it is not already.
- (HA) Login to the secondary passive head node and stop CMDaemon (systemctl stop cmd). Note: make sure to stop CMDaemon on the passive secondary head node, and not on the primary active head node.
- (HA) On the same secondary passive head node, edit /etc/fstab and comment out the entry for `/cm/shared`, so that on the next boot the secondary passive head node does not mount `/cm/shared` from the file server.
- (HA) Shut down the secondary passive head node. The node should remain shut down until later when plugged in to the parallel setup.
- Plug the spare node to the internal network of the production cluster and PXE boot it to the Bright rescue environment (select RESCUE in the PXE boot menu, in the same way as performing a clone for an HA setup)
- Log in to the rescue environment and run `cm-clone-install`, providing the product key and the MAC address of this spare node. In the case of HA, the MAC address of the secondary head node must also be provided. Thus, for the non-HA case:

```
# /cm/cm-clone-install --clone --upgrade --productkey PRODUCT-KEY \
 --primarymac PRIMARY-MAC
```

and for the HA case:

```
# /cm/cm-clone-install --clone --upgrade --productkey PRODUCT-KEY \
 --primarymac PRIMARY-MAC --secondarymac SECONDARY-MAC
```

and follow the instructions. At the end of the cloning process, confirm shutting down the spare node.

In the preceding the values used for the primary and secondary MAC addresses are specified as follows:
`PRIMARY_MAC`: the MAC address of the management interface of the primary head node (i.e the one that is on the management network)
`SECONDARY_MAC`: the MAC address of the management interface of the secondary head node (i.e the one that is on the management network)

- Unplug the spare node from the production setup/networks and plug it in the parallel setup.
- In the case of `/cm/shared` on a file server (i.e. usually on an HA setup), login to the file server and make a copy of the `/cm/shared` to a new directory. Alternatively, the copy can be made to a new file server. Have the directory name and the (possibly new) file server host name ready.
- Boot the spare (a.k.a. cloned primary head) node as it is plugged in the parallel (isolated) setup. The node will boot with all of its network interfaces remaining DOWN.
- Login on console (ignoring possible messages about not-found modules in `/cm/shared`), and run /root/cm/update-cloned-db.py, providing the new location of `/cm/shared`:

  ```
  # /root/cm/update-cloned-db.py --new-cm-shared NEW-CM-SHARED-DIRECTORY-NAME \
   --nas-server NAS-SERVER
  ```

  which will setup in CMDaemon to mount the copy of `/cm/shared` made in the preceding session from NAS-SERVER:NEW-CM-SHARED-DIRECTORY-NAME

  or, if `/cm/shared` is not on a NAS server (i.e. no copy of `/cm/shared` was made in the preceding session), then just run:

  ```
  # /root/cm/update-cloned-db.py
  ```

  The script finds values for MAC addresses, interfaces, and so on that came as part of the original primary head node, and converts those values to ones suitable for the clone.
- After /root/cm/update-cloned-db.py is completed, on the cloned primary head node still logged in on the console, by using ”cmsh -r localhost”, change the IPs of the primary head node (master) on the external network(s).
- In the case of HA: change also the shared IP (alias) of the secondary head node on the external network(s) to match the new shared IP(s) of the primary head node.
- If necessary, update also other settings such as power management settings or if desired fsmounts, to reflect that in this parallel setup the primary head node is a different physical machine, which is a clone of the original primary head node.

- Reboot, to allow for the settings to take effect. After the reboot, the (cloned) primary head node in the parallel setup will have the network interfaces UP.
- Ensure that this primary head node in the parallel setup has correctly mounted the copied `/cm/shared` from the file server, if a copy of `/cm/shared` was made in the preceding.

  If `/cm/shared` is not correctly mounted, then verify the fsmount settings for the primary head node, the secondary head node, as well as for the compute nodes (typically the compute nodes fsmount settings are set in the compute nodes categories) are correct and the head node is able to connect to the file server.
- (HA) Plug in the secondary head node to the parallel setup and boot it. After the secondary head node has booted, ensure it has mounted correctly the `/cm/shared` copy from the file server.
- (HA) On both the (cloned) primary head node and the secondary head node, ensure the active head node is the (cloned) primary head node, the failover is in a good state and the mysql replication is working by running (on both head nodes):

  ```
  # cmha status
  ```

  If any of the returned responses for `failoverping`, `mysql`, `ping`, or `status` do not report "OK", then ensure the network configuration and cabling allow the two head nodes in the parallel setup to communicate, and that they are isolated from the production setup. If this does not resolve the issue, then please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.
- Disable/remove the Bright Cluster Manager integrations, such as with Kubernetes or WLM for some versions of Bright, as described in the important notes and other prerequisites in the preceding.

The notes in sections 2.3–2.8 should now be considered, before continuing on with the upgrade as described in section 2.9.

2.3 Important: Considerations For The Parallel Upgrade Setup

2.3.1 Configuration Changes And Services On The Production Cluster Desynchronized

The parallel upgrade procedure involves creating a clone of the primary head node and its isolation from the production cluster. The administrator must be aware that this means that any further configuration changes on the production cluster, further collected monitoring data, other changes in settings or data, such as the running or new WLM jobs' state data, that is usually saved on the head node or in `/cm/shared` will not be mirrored or synced to the parallel setup after creating the clone of the primary head node, resp. the copy of the `/cm/shared` directory tree.

In just the same way, services running on the original primary head node will also be started by the operating system on the cloned head node, but they will not be able to communicate with the production cluster nodes. As an example, if the cluster is running with Ceph integration with only one monitor running on the head node, then the Ceph monitor on the production head node will continue to communicate with the Ceph OSDs nodes and maintain the Ceph cluster state. In the parallel setup, the Ceph monitor service on the cloned head node cannot communicate with any node from the production cluster and will not be aware of the Ceph cluster state changes.

Therefore some special services running on the parallel-setup head node may not be able to take over from the respective services running on the production head node when the old production head node is to be shut down and the upgraded parallel-setup head node is to become the

production head node. In these special cases, the administrator needs to take the necessary steps to migrate the services, or the administrator needs to perform an in-place upgrade.

2.3.2 Local Data On The Head Node Desynchronized

When the users' home directories (`/home`) or other data are not on a file server, then the administrator will most commonly need to take the necessary steps to sync the latest changes in the users' home, resp. other directories from the production setup to the parallel setup when the upgrade of the parallel setup is complete.

2.3.3 Dangling, Temporary, And Relocated Mounted Network File Systems

If the users' home directories are on a file server, then a manual sync of the data as pointed in the preceding is unnecessary, however the administrator must be aware that except for `/cm/shared` as noted below, the head node(s) in the parallel setup will retain their fsmount settings also during the upgrade. This means that also during the upgrade the head nodes from the parallel setup will attempt to mount the network file systems. If this behavior is not desired, the administrator can configure the parallel-upgrade network and cabling in such a way so that the file server(s) are also isolated from the parallel setup. Alternatively, the administrator can modify the fsmount settings after the primary head node is cloned when it is booted for the first time as described below, so that the head nodes in the parallel setup do not mount these network file systems.

On the other hand, if mounting the network file systems on the head nodes in the parallel setup is required, for example so that users can log in and test the upgraded setup while the production setup is still operational in parallel, then the network configuration/cabling should allow the head node to access the file servers. When the file servers are on an external network, usually no additional steps will be required since the head nodes in the parallel setup will have full access to the external networks.

If the file servers are on the internal network(s) however, because by default the cloned head node retains the same internal network IPs, and because the parallel setup's internal networks should be isolated from the production setup, by default the parallel-setup's head nodes cannot access the file servers or any other device on the production-cluster's internal network(s). Additional measures such as setting up intermediate servers or re-configuring the network settings will have to be taken to ensure the parallel-setup's nodes can access the data on the file servers when needed, or to make sure they have access to a copy of the data on different (possibly temporary) file servers. Such configuration changes are beyond the scope of this document, for further information please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.

A special case is `/cm/shared` mounted from a file server. The clone and upgrade scripts will detect that, and the administrator will be required to copy `/cm/shared` to a new location and provide the (new) path and file server host name. The new location can be a new directory on the same file server if the file server is on an external network. If the file server is on the internal network, the simplest solution is to set up a new or temporary file server to host the required for the parallel upgrade copy of `/cm/shared`.

2.3.4 Cluster Extension Upgrade Not Supported

Parallel upgrade of clusters extended to the cloud (Cluster Extension cloudbursting) is not supported. For these clusters only an in-place upgrade is possible, which involves the termination of all cloud nodes when the cluster extension is removed as part of the upgrade procedure.

2.3.5 Edge Setup Upgrade Not Supported

Parallel upgrade of clusters with an edge setup is not supported.

2.3.6 Prerequisites

- The cluster must have a previously installed and active subscription license.
  - Sites with hardware life licenses should contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.
  - Before starting with the upgrade, the subscription license should first be unlocked via the self-service customer portal at `https://customer.brightcomputing.com/Customer-Portal?p=unlock`
- A spare node is required for cloning the original (a.k.a production) primary head node. The spare node must have comparable to the current head node hardware, such as number of network cards, number of disks and size, memory, etc. After the completion of the upgrade, the spare node will become the new primary head node for the upgraded cluster.
- In the case of head node HA, the secondary head node will need to be taken out of the production cluster so that it can also be upgraded. This means that for the time needed to perform the parallel upgrade, until all compute nodes are migrated to the upgraded setup, the old production cluster will run with only one head node.
- If `/cm/shared` will be unmounted during the upgrade then please make sure that the contents of the local `/cm/shared` are in sync with the remote copy.
- In the case of `/cm/shared` on a file server (such as in the case of head node HA), the file server must have enough space for creating a copy of the `/cm/shared` directory tree. Alternatively, `/cm/shared` can be copied to a new file server.
- While performing the upgrade, the cloned primary and—in the case of HA—the secondary head node(s) need to have isolated internal network(s) which do not clash with the internal network(s) of the production cluster. The IP(s) of the upgraded head node(s) on the internal network(s) will remain the same. After the upgrade is completed, the old (not upgraded) primary head must be shut down and the upgraded head node(s) can be re-connected back in the cluster. Alternatively, compute nodes can be disconnected from the old (not upgraded) setup and then connected in the upgraded setup, which can also be performed gradually.
- There is no requirement to isolate the external network(s) for the head node(s) while performing the upgrade, which means the cloned head node can still have access to the Internet. However, the cloned primary head node will require a new set of IPs on the external network(s), which are different from the current IP(s) of the primary head node on the external network(s). It is also recommended to use static IPs, and not DHCP with dynamic DNS updates on the external network, because both the original production head node and its clone in the parallel setup will have the same hostname as defined in CMDaemon. The secondary head node does not require a new set of IPs because it is simply moved (and not cloned) from the production to the parallel setup.
- When upgrading the parallel setup, the head nodes require access to the new BCM package repositories. Alternatively, the BCM ISOs can be used as a source for the packages. Access to the base distro package repositories, using for example, the via internet or local repositories, is also required.

  Updates using the old Bright package repositories should be carried out only to bring the cluster up to date before running `cm-upgrade`.
- In the case of HA, the cloned primary and the secondary head nodes will require to be able to connect to the file server for their copied `/cm/shared` directory tree. This means that if the file server is on an internal network, then—because the internal networks must be isolated—a new file server must be setup and made available on the isolated internal network.

2.4 Important: Upgrading Particular Applications

This section has important notes on upgrading particular applications.

2.4.1 Workload Managers (WLM)

Torque, SGE and PBSPro CE workload managers are not supported in BCM 10, and hence the relevant packages will be removed as part of the upgrade.

Unversioned WLM packages (such as `pbspro`/`pbspro-client`/`pbspro-ce`/`pbspro-ce-client` or `slurm`/`slurm-client`) will be removed and the administrator can install a versioned WLM package of their choice (such as `pbspro2022`/`pbspro2022-client` or `slurm22.05`/`slurm22.05-client`).

On RHEL and SLES distributions, unsupported versioned WLM packages are not removed automatically. They must be removed manually and where relevant new versioned WLM packages must be installed.

Upgrading WLMs In General From Bright 9.2

Workload manager configurations must be disabled using `cm-wlm-setup` before proceeding with the upgrade.

After the upgrade has been completed, workload management must be re-deployed using `cm-wlm-setup`.

Slurm Upgrades

Slurm versions older than Slurm 23.02 are not supported in BCM 10. When upgrading from Bright 9.2, Slurm must be upgraded to at least Slurm 23.02 from the Bright 9.2 repositories before moving to BCM 10.

For more information on upgrading Slurm upgrades, please refer to the KB article `https://kb.brightcomputing.com/knowledge-base/upgrading-slurm/`.

Slurm DBD Performance Improvement

Starting with the BCM 10.24.05 release, some MySQL configuration settings have been changed to improve the performance of the slurmdbd service. The changes are not automatically applied during the cluster upgrade. The MySQL configuration must therefore be manually adjusted on the head nodes, after `cm-upgrade` has completed. The /etc/my.cnf on the head node, and also on the secondary head node in an HA-enabled cluster, should have its settings updated as follows:

```
innodb_buffer_pool_size=4096M
innodb_log_file_size=64M
innodb_lock_wait_timeout=900
```

PBS Pro Upgrades

PBS Pro versions older than PBS Pro 2022 are not supported in BCM 10. Below is an overview of the steps that can be taken to upgrade PBS Pro:

- Stop the `pbsserver` service on the head node

  ```
  cmsh% device
  cmsh% foreach -l pbsproserver ( services; stop pbsserver )
  ```

- Remove the old packages from the headnodes

  ```
  $ yum/zypper/apt remove pbspro<old version>*
  ```

- Remove the old packages from the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
[root@<software image> /]# yum/zypper/apt remove pbspro<old version>*
```

- Install the new packages in the headnode

```
$ yum/zypper/apt install pbspro<new version> pbspro<new version>-client
```

- Install the new packages in the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
...
[root@<software image> /]# yum/zypper/apt install pbspro<new version>-client
```

- Change the version of the WLM in the `wlm` object

```
cmsh% wlm use pbspro
cmsh% set version <new version>
cmsh% commit
```

- Start `pbsserver`

```
cmsh% device
cmsh% foreach -l pbsproserver ( services; start pbsserver )
```

2.4.2 Older OpenMPI versions 1.x Unsupported

The 1.x series of OpenMPI is not supported in BCM 10. These packages are removed during the upgrade. Administrators must install OpenMPI 4 packages manually, and rebuild the applications that use them after the upgrade has been completed.

2.4.3 Kubernetes Package Upgrades

Supported Kubernetes deployments must be removed before upgrading and re-deployed after the upgrade.

- Remove the old packages from the headnodes

```
$ yum/zypper/apt remove cm-kubernetes* cm-etcd cm-kube-diagnose cm-containerd
```

- Remove the old packages from the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
[root@<software image> /]# yum/zypper/apt remove cm-kubernetes* cm-etcd \
 cm-kube-diagnose cm-containerd
```

For more details please refer to chapter 4 of the BCM 10 Containerization Manual.

2.4.4 JupyterHub Database Upgrade

The JupyterHub database schema must be manually upgraded in each JupyterHub node after the main package upgrade has completed. The following commands can be run to upgrade the JupyterHub database:

```
module load jupyter
jupyterhub upgrade-db--config=/cm/local/apps/jupyter/conf/jupyterhub_config.py \
  --db sqlite:////cm/local/apps/jupyter/run/jupyterhub.sqlite
```

2.4.5 Docker Registry/Harbor Package Upgrades

When upgrading a cluster using the BCM ISO, if the Docker registry or Harbor packages were previously installed, then they do not get upgraded automatically on RPM-based distributions. On Ubuntu based clusters the packages get removed automatically.

The packages must be manually installed and upgraded using the package managers (`apt`/`zypper`/`yum`).

If Docker registry was installed, then the `cm-docker-registry` package must be upgraded/installed.

If Harbor was installed, then the packages to upgrade/install are: `cm-harbor cm-docker cm-docker-compose`

Also note that if they were deployed on a different node than the head node, then the packages must be upgraded/installed in the relevant software image(s).

Harbor versions older than 2.8 are not supported in BCM 10. When upgrading from Bright clusters running an older Harbor version, remove it with `cm-container-registry-setup` and re-deploy it after the upgrade.

2.4.6 OpenStack Integration

- Bright OpenStack deployments must be removed (using cm-openstack-setup) before upgrading. OpenStack integration is no longer supported in BCM 10, and cannot be re-deployed after the upgrade. All older Bright OpenStack packages and dependencies must be removed prior to starting the upgrade. Please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.

2.5 Important: Package Upgrade Dependency Issues

The upgrade process will not only upgrade CMDaemon and its dependencies, but it will also upgrade other packages. This means that old packages will not be available from the new BCM repositories.

In some cases, this will require recompiling the user applications to use the upgraded versions of the compilers and the libraries. Also, the configurations of the old packages will not be copied automatically to the new packages, which means that the administrator will have to adjust the configuration from the old packages to suit the new packages manually.

Extra base distribution packages may be installed by `yum`/`zypper`/`apt-get` in order to resolve dependencies that might arise as a result of the upgrade. Hence the base distribution repositories must be reachable. This means that the clusters that run the Enterprise Linux distributions (RHEL and SLES) must be subscribed to the appropriate software channels.

Packages in `/cm/shared` are upgraded, but the administrator should be aware of the following:

- If `/cm/shared` is installed in the local partition, then the packages are upgraded. This may not be desirable for users that wish to retain the old behavior.
- If `/cm/shared` is mounted from a separate partition, then unmounting it will prevent upgrades to the mounted partition, but will allow new packages to be installed in `/cm/shared` within

the local partition. This may be desirable for the administrator, who can later copy over updates from the local /cm/shared to the remote /cm/shared manually according to site-specific requirements. Since unmounting of mounted /cm/shared is carried out by default, a local /cm/shared will have files from any packages installed there upgraded. According to the yum database, the system is then upgraded even though the files are misplaced in the local partition. However, the newer packages can only be expected to work properly if their associated files are copied over from the local partition to the remote partition.

2.6 Important: Upgrading Ubuntu Clusters

After upgrading the cluster, the Ubuntu compute nodes must initially be re-provisioned with a FULL install. To achieve that, the following cmsh example command will update each Ubuntu compute node. In this example, 750 nodes are updated:

```
% device foreach -n node001..node750 (set nextinstallmode FULL); device commit
```

Important note: a FULL install will remove any local data on the compute nodes. Thus the administrator must perform the necessary steps to back up the data on the compute nodes before the upgrade and restore it afterward.

2.7 Known Issues

- On clusters with Jupyter deployed, after the upgrade, users might find themselves being asked to login twice. The first time when the user will be asked to login to JupyterHub, and a second time, where the user is asked about a "CMDaemon login". In such cases, the user should once again present the credentials. This extra login request is harmless, and after the credentials are given a second time, it is unlikely that users will be queried again. The cause of this issue is that the Jupyterhub component that interacts with CMDaemon can lose connection to CMDaemon, causing it to prompt for an extra login.
- On some base distributions, depending on the version of the installed packages, when starting cm-upgrade, a YAMLLoadWarning message may be printed when its configuration files are loaded. The warning message can be safely ignored.
- In some cases, in particular HA setup with a shared storage, it is possible that the slurmdbd.conf.template configuration file is overwritten after the upgrade where the package manager would not preserve a backup copy of the locally modified file. As a result, when Slurm WLM is later setup with cm-wlm-setup CMDaemon will not start slurmctld service because it will detect slurmdbd service is not able to connect to the mysql database. To resolve the issue, the administrator needs to run /cm/shared/apps/slurm/var/cm/cm-restore-db-password to set up a new MySQL password for the Slurm user, and should then restart the slurmdbd service.
- In some cases because /cm/shared may be temporarily not accessible from the cloned primary head node when it boots for the first time, if workload managers are setup on the cluster, then CMDaemon may create and leave module files called no-version. For example:
  slurm/slurm/no-version
  To resolve the issue, the leftover no-version module files under /cm/shared/modulefiles/ can be deleted.
- In some cases at the end of the head node or the software image(s) upgrade, the cm-upgrade script may report a list of packages that were not upgraded to their BCM 10 versions. This can occur in cases where the package manager was not able to resolve the list of packages to find a suitable candidate at the time of the upgrade of all packages.

An example here the OpenBLAS packages upgrades for BCM. In BCM prior to BCM 10, the package was called `openblas`, packaged by BCM developers. In BCM 10 the BCM package was replaced by `cm-openblas` (also packaged by BCM developers). The issue occurs since the base distro also can provide a (non-BCM) OpenBLAS package. If cm-upgrade reports that the BCM OpenBLAS package is not upgraded, then the administrator is advised to swap the OpenBLAS package manually with the `cm-openblas` package.

While cm-upgrade makes a best effort to swap or upgrade the packages to their BCM 10 versions, in some cases - notably on Ubuntu, it is also possible some packages such as `cmburn`, `hdf518`, or `hpl` are left in their 9.2 versions. The administrator is advised to manually swap the `cm9.2` packages (for example `cmburn-9.2-392_cm9.2`) with the `cm10.0` packages (for example `cmburn_10.0-391_cm10.0`).

- On clusters with cluster extension to AWS deployed, the cloud director and cloud nodes might fail to start after the upgrade. This can happen when there is an entry

  ```
  <blockdev mode="cloud">/dev/nvme1n1</blockdev>
  ```

  in the disksetup. The issue can be resolved by removing that entry from the disksetup for the cloud director and cloud nodes.

- On clusters that have PBSPro configured, and upgraded from 9.2 to BCM 10, new jobs submitted by users will get job ids that were already used. This is particularly relevant for users that use BCM monitoring data for billing purposes.

- Disabling Kubernetes as part of the parallel-upgrade procedure results in an apparent failure during the stage when etcd data is cleaned up, because the compute nodes are not reachable from the parallel setup. This is expected since the parallel setup is isolated from the production setup. To resolve the issue, the administrator should answer a prompt so that the failed stage is skipped. It is also recommended to re-provision the compute nodes of the upgraded parallel setup before attempting to set up Kubernetes or etcd again, so that any leftover data from before the upgrade is cleaned during the provisioning.

2.8 Upgrading Using A BCM DVD/ISO

When using a BCM DVD/ISO to perform the upgrade, it is important to use a DVD/ISO that is not older than **10.24.01**. The DVD/ISO version can be found (assuming that the DVD/ISO is mounted under `/mnt/cdrom`) with a find command such as:

```
# find /mnt/cdrom -type d -name '10.24.01'
/mnt/cdrom/data/packages/10.24.01
```

2.9 Enable Upgrade

2.9.1 Enable The Upgrade Repo, And Install The Upgrade Package

Enabling the upgrade repository and installing the upgrade package can be carried out for the various distributions as follows:

RHEL8 derivatives:

```
# yum-config-manager \
  --add-repo http://support.brightcomputing.com/upgrade/10.0/rhel/8/updates
```

```
# yum --nogpgcheck install cm-upgrade
```

RHEL9 derivatives:

```
# yum-config-manager \
--add-repo http://support.brightcomputing.com/upgrade/10.0/rhel/9/updates
# yum --nogpgcheck install cm-upgrade
```

SLES 15:

```
# zypper addrepo \
http://support.brightcomputing.com/upgrade/10.0/sles/15/updates cm-upgrade-10
# zypper install cm-upgrade
```

Ubuntu 20.04:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-10.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/10.0/ubuntu/2004/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

Ubuntu 22.04:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-10.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/10.0/ubuntu/2204/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

2.9.2 Load The Environment Module

```
# module load cm-upgrade/10.0
```

2.10 Perform Upgrade

The steps for carrying out the upgrade are as follows:

2.10.1 Apply Updates To Head Node

- RHEL derivatives:

  ```
  # yum update
  ```

- SLES derivatives:

  ```
  # zypper up
  ```

- Ubuntu:

```
# apt-get update
# apt-get upgrade --with-new-pkgs
```

2.10.2 Apply Updates To Software Images

Before updating and upgrading a software image, it may be worth creating an image backup by cloning the image (page 202 of the *Administrator Manual*). If the cluster after the upgrade has an image that has an issue, then comparing it against the backup can help troubleshoot the issue.

For each software image, updates can be applied as follows:

- RHEL derivatives:

```
# yum --installroot /cm/images/<software image> update
```

- SLES derivatives:

```
# zypper --root /cm/images/<software image> up
```

- Ubuntu:

```
# cm-chroot-sw-img /cm/images/<software image>
# apt-get update
# apt-get upgrade --with-new-pkgs
# exit
```

2.10.3 Apply Updates To The `node-installer` Image

- RHEL derivatives:

```
# yum --installroot /cm/node-installer update
```

- SLES derivatives:

```
# zypper --root /cm/node-installer up
```

- Ubuntu:

```
# cm-chroot-sw-img /cm/node-installer
# apt-get update
# apt-get upgrade --with-new-pkgs
# exit
```

2.10.4 Upgrade Head Nodes To BCM 10

Important: this must be run on both head nodes in a high availability setup. Recommended: Upgrade the active head node first and then the passive head node.

Depending on availability, the upgrade can be run either over the network, or using a DVD/ISO:

- Upgrade using repositories accessible over the network

```
# cm-upgrade
```

- Upgrade using a BCM DVD/ISO

```
# cm-upgrade -b /root/bcm-10.0-rocky8u8.iso
```

In an HA setup, after upgrading both the head nodes, resync the databases. Run the following from the active head node (it is very important to complete this step before moving to the next one):

```
# cmha dbreclone <secondary>
```

2.10.5 Reboot Head Node(s)

The head node(s) must be rebooted before proceeding to run the post-upgrade actions.

2.10.6 Post-upgrade Head Node Actions

Important: these must be run on both head nodes in an HA setup

```
# module load cm-upgrade/10

# cm-post-upgrade -m
```

2.10.7 Upgrade The Software Image(s) To BCM 10

Important: the upgrade to the software images must be done only on the active head node. Depending on availability, the upgrade can be run either over the network, or using a DVD/ISO:

- Upgrade using repositories accessible over the network

```
# cm-upgrade -i all
```

- Upgrade using a BCM DVD/ISO

```
# cm-upgrade -i all -b /root/bcm-10.0-rocky8u8.iso
```

If the software images are not under the standard location—`/cm/images/` on the head node—then the option `-a` should be used:

```
# cm-upgrade -a /apps/images -i <name of software image>
# cm-upgrade -a /apps/images -i <name of software image> -b /root/bcm-10.0-rocky8u8.iso
```

If the software images are located under multiple locations, the target locations can be specified as a comma-separated list:

```
# cm-upgrade -a '/apps/images,/opt/images' -i all
```

2.10.8 Upgrade /cm/node-installer To BCM 10

This needs to be performed only on the primary active head node, either over the network, or using a BCM DVD/ISO.

- Upgrade using repositories accessible over the network:

```
# cm-upgrade -x
```

- Upgrade using a BCM DVD/ISO:

```
# cm-upgrade -x -b /root/bcm-10.0-rocky8u8.iso
```

2.10.9 Post-upgrade Software Images Actions

Important: these must be run only on the active head node.

```
# cm-post-upgrade -i all
```

If the software images are not under the standard location, which is `/cm/images/` on the head node, then the option `-a` should be used:

```
# cm-post-upgrade -a /apps/images -i all
# cm-post-upgrade -a /apps/images -i <name of software image>
```

If the software images are located under multiple locations, then the target locations can be specified as a comma-separated list:

```
# cm-post-upgrade -a '/apps/images,/opt/images' -i all
```

2.10.10 (HA) Allow For The Provisioning Requests To The Secondary Head Node To Complete

Allow between 15 and 30 seconds for CMDaemon to schedule and start provisioning updates of `/tftpboot` and `/cm/node-installer` from the primary to the secondary head node. Then check and wait until the provisioning requests with the secondary head node as destination have completed using:

```
# cmsh -c 'softwareimage provisioningstatus -r'
```

If the provisioning requests with a destination node the secondary head node continue to be listed for a prolonged period of time and do not disappear from the list, then the `node-installer` and/or `tftpboot` on the secondary head node are out of sync from the primary head node, which can result in issues with provisioning of the compute nodes. Please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.

2.10.11 The Cluster Is Now Upgraded To BCM 10

It is recommended to provision the compute nodes at least once with BCM 10 before setting up any new integrations. In the case of Ubuntu base distro, it is recommended to initially perform a FULL install/provisioning of the compute nodes as noted in the preceding.

3 In-place Upgrades

3.1 Introduction

In contrast with a parallel upgrade (Chapter 2), with an in-place upgrade the production cluster is taken out of operation while the upgrade takes place.

This chapter covers the procedure for an in-place upgrade.

3.2 Important: Upgrading Particular Applications

This section has important notes on upgrading particular applications.

3.2.1 Workload Managers (WLM)

Torque, SGE and PBSPro CE workload managers are not supported in BCM 10, and hence the relevant packages will be removed as part of the upgrade.

Unversioned WLM packages (such as `pbspro`/`pbspro-client`/`pbspro-ce`/`pbspro-ce-client` or `slurm`/`slurm-client`) will be removed and the administrator can install a versioned WLM package of their choice (such as `pbspro2022`/`pbspro2022-client` or `slurm22.05`/`slurm22.05-client`).

On RHEL and SLES distributions, unsupported versioned WLM packages are not removed automatically. They must be removed manually and where relevant new versioned WLM packages must be installed.

Upgrading WLMs In General From Bright 9.2

Workload manager configurations must be disabled using `cm-wlm-setup` before proceeding with the upgrade.

After the upgrade has been completed, workload management must be re-deployed using `cm-wlm-setup`.

Slurm Upgrades

Slurm versions older than Slurm 23.02 are not supported in BCM 10. When upgrading from Bright 9.2, Slurm must be upgraded to at least Slurm 23.02 from the Bright 9.2 repositories before moving to BCM 10.

For more information on upgrading Slurm upgrades, please refer to the KB article `https://kb.brightcomputing.com/knowledge-base/upgrading-slurm/`.

Slurm DBD Performance Improvement

Starting with the BCM 10.24.05 release, some MySQL configuration settings have been changed to improve the performance of the slurmdbd service. The changes are not automatically applied during the cluster upgrade. The MySQL configuration must therefore be manually adjusted on the head nodes, after `cm-upgrade` has completed. The `/etc/my.cnf` on the head node, and also on the secondary head node in an HA-enabled cluster, should have its settings updated as follows:

```
innodb_buffer_pool_size=4096M
innodb_log_file_size=64M
innodb_lock_wait_timeout=900
```

PBS Pro Upgrades

PBS Pro versions older than PBS Pro 2022 are not supported in BCM 10. Below is an overview of the steps that can be taken to upgrade PBS Pro:

- Stop the `pbsserver` service on the head node

```
cmsh% device
cmsh% foreach -l pbsproserver ( services; stop pbsserver )
```

- Remove the old packages from the headnodes

```
$ yum/zypper/apt remove pbspro<old version>*
```

- Remove the old packages from the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
[root@<software image> /]# yum/zypper/apt remove pbspro<old version>*
```

- Install the new packages in the headnode

```
$ yum/zypper/apt install pbspro<new version> pbspro<new version>-client
```

- Install the new packages in the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
...
[root@<software image> /]# yum/zypper/apt install pbspro<new version>-client
```

- Change the version of the WLM in the `wlm` object

```
cmsh% wlm use pbspro
cmsh% set version <new version>
cmsh% commit
```

- Start `pbsserver`

```
cmsh% device
cmsh% foreach -l pbsproserver ( services; start pbsserver )
```

3.2.2 Older OpenMPI versions 1.x Unsupported

The 1.x series of OpenMPI is not supported in BCM 10. These packages are removed during the upgrade. Administrators must install OpenMPI 4 packages manually, and rebuild the applications that use them after the upgrade has been completed.

3.2.3 Kubernetes Package Upgrades

Supported Kubernetes deployments must be removed before upgrading and re-deployed after the upgrade.

- Remove the old packages from the headnodes

```
$ yum/zypper/apt remove cm-kubernetes* cm-etcd cm-kube-diagnose cm-containerd
```

- Remove the old packages from the software images

```
$ cm-chroot-sw-img /cm/images/<software image>
...
[root@<software image> /]# yum/zypper/apt remove cm-kubernetes* cm-etcd \
 cm-kube-diagnose cm-containerd
```

For more details please refer to chapter 4 of the BCM 10 Containerization Manual.

3.2.4 JupyterHub Database Upgrade

The JupyterHub database schema must be manually upgraded in each JupyterHub node after the main package upgrade has completed. The following commands can be run to upgrade the JupyterHub database:

```
module load jupyter
jupyterhub upgrade-db--config=/cm/local/apps/jupyter/conf/jupyterhub_config.py \
  --db sqlite:////cm/local/apps/jupyter/run/jupyterhub.sqlite
```

3.2.5 Docker Registry/Harbor Package Upgrades

When upgrading a cluster using the BCM ISO, if the Docker registry or Harbor packages were previously installed, then they do not get upgraded automatically on RPM-based distributions. On Ubuntu based clusters the packages get removed automatically.

The packages must be manually installed and upgraded using the package managers (`apt`/`zypper`/`yum`).

If Docker registry was installed, then the `cm-docker-registry` package must be upgraded/installed.

If Harbor was installed, then the packages to upgrade/install are: `cm-harbor cm-docker cm-docker-compose`

Also note that if they were deployed on a different node than the head node, then the packages must be upgraded/installed in the relevant software image(s).

Harbor versions older than 2.8 are not supported in BCM 10. When upgrading from Bright clusters running an older Harbor version, remove it with `cm-container-registry-setup` and re-deploy it after the upgrade.

3.2.6 OpenStack Integration

- Bright OpenStack deployments must be removed (using cm-openstack-setup) before upgrading. OpenStack integration is no longer supported in BCM 10, and cannot be re-deployed after the upgrade. All older Bright OpenStack packages and dependencies must be removed prior to starting the upgrade. Please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.

3.3 Important: Package Upgrade Dependency Issues

The upgrade process will not only upgrade CMDaemon and its dependencies, but it will also upgrade other packages. This means that old packages will not be available from the new BCM repositories.

In some cases, this will require recompiling the user applications to use the upgraded versions of the compilers and the libraries. Also, the configurations of the old packages will not be copied automatically to the new packages, which means that the administrator will have to adjust the configuration from the old packages to suit the new packages manually.

Extra base distribution packages may be installed by `yum/zypper/apt-get` in order to resolve dependencies that might arise as a result of the upgrade. Hence the base distribution repositories must be reachable. This means that the clusters that run the Enterprise Linux distributions (RHEL and SLES) must be subscribed to the appropriate software channels.

Packages in `/cm/shared` are upgraded, but the administrator should be aware of the following:

- If `/cm/shared` is installed in the local partition, then the packages are upgraded. This may not be desirable for users that wish to retain the old behavior.
- If `/cm/shared` is mounted from a separate partition, then unmounting it will prevent upgrades to the mounted partition, but will allow new packages to be installed in `/cm/shared` within the local partition. This may be desirable for the administrator, who can later copy over updates from the local `/cm/shared` to the remote `/cm/shared` manually according to site-specific requirements. Since unmounting of mounted `/cm/shared` is carried out by default, a local `/cm/shared` will have files from any packages installed there upgraded. According to the yum database, the system is then upgraded even though the files are misplaced in the local partition. However, the newer packages can only be expected to work properly if their associated files are copied over from the local partition to the remote partition.

3.4 Important: Upgrading Ubuntu Clusters

After upgrading the cluster, the Ubuntu compute nodes must initially be re-provisioned with a `FULL` install. To achieve that, the following `cmsh` example command will update each Ubuntu compute node. In this example, 750 nodes are updated:

```
% device foreach -n node001..node750 (set nextinstallmode FULL); device commit
```

Important note: a `FULL` install will remove any local data on the compute nodes. Thus the administrator must perform the necessary steps to back up the data on the compute nodes before the upgrade and restore it afterward.

3.5 Known Issues

- On clusters with Jupyter deployed, after the upgrade, users might find themselves being asked to login twice. The first time when the user will be asked to login to JupyterHub, and a second time, where the user is asked about a "CMDaemon login". In such cases, the user should once again present the credentials. This extra login request is harmless, and after the credentials are given a second time, it is unlikely that users will be queried again. The cause of this issue is that the Jupyterhub component that interacts with CMDaemon can lose connection to CMDaemon, causing it to prompt for an extra login.
- On some base distributions, depending on the version of the installed packages, when starting `cm-upgrade`, a YAMLLoadWarning message may be printed when its configuration files are loaded. The warning message can be safely ignored.

- In some cases, in particular HA setup with a shared storage, it is possible that the `slurmdbd.conf.template` configuration file is overwritten after the upgrade where the package manager would not preserve a backup copy of the locally modified file. As a result, when Slurm WLM is later setup with `cm-wlm-setup` CMDaemon will not start slurmctld service because it will detect slurmdbd service is not able to connect to the mysql database. To resolve the issue, the administrator needs to run `/cm/shared/apps/slurm/var/cm/cm-restore-db-password` to set up a new MySQL password for the Slurm user, and should then restart the `slurmdbd` service.

- In some cases because `/cm/shared` may be temporarily not accessible from the cloned primary head node when it boots for the first time, if workload managers are setup on the cluster, then CMDaemon may create and leave module files called `no-version`. For example:
  `slurm/slurm/no-version`
  To resolve the issue, the leftover `no-version` module files under `/cm/shared/modulefiles/` can be deleted.

- In some cases at the end of the head node or the software image(s) upgrade, the cm-upgrade script may report a list of packages that were not upgraded to their BCM 10 versions. This can occur in cases where the package manager was not able to resolve the list of packages to find a suitable candidate at the time of the upgrade of all packages.

  An example here the OpenBLAS packages upgrades for BCM. In BCM prior to BCM 10, the package was called `openblas`, packaged by BCM developers. In BCM 10 the BCM package was replaced by `cm-openblas` (also packaged by BCM developers). The issue occurs since the base distro also can provide a (non-BCM) OpenBLAS package. If cm-upgrade reports that the BCM OpenBLAS package is not upgraded, then the administrator is advised to swap the OpenBLAS package manually with the `cm-openblas` package.

  While cm-upgrade makes a best effort to swap or upgrade the packages to their BCM 10 versions, in some cases - notably on Ubuntu, it is also possible some packages such as `cmburn`, `hdf518`, or `hpl` are left in their 9.2 versions. The administrator is advised to manually swap the `cm9.2` packages (for example `cmburn-9.2-392_cm9.2`) with the `cm10.0` packages (for example `cmburn_10.0-391_cm10.0`).

- On clusters with cluster extension to AWS deployed, the cloud director and cloud nodes might fail to start after the upgrade. This can happen when there is an entry

  ```
  <blockdev mode="cloud">/dev/nvme1n1</blockdev>
  ```

  in the disksetup. The issue can be resolved by removing that entry from the disksetup for the cloud director and cloud nodes.

- On clusters that have PBSPro configured, and upgraded from 9.2 to BCM 10, new jobs submitted by users will get job ids that were already used. This is particularly relevant for users that use BCM monitoring data for billing purposes.

- Disabling Kubernetes as part of the parallel-upgrade procedure results in an apparent failure during the stage when etcd data is cleaned up, because the compute nodes are not reachable from the parallel setup. This is expected since the parallel setup is isolated from the production setup. To resolve the issue, the administrator should answer a prompt so that the failed stage is skipped. It is also recommended to re-provision the compute nodes of the upgraded parallel setup before attempting to set up Kubernetes or etcd again, so that any leftover data from before the upgrade is cleaned during the provisioning.

3.6 Upgrading Using A BCM DVD/ISO

When using a BCM DVD/ISO to perform the upgrade, it is important to use a DVD/ISO that is not older than **10.24.01**. The DVD/ISO version can be found (assuming that the DVD/ISO is mounted under `/mnt/cdrom`) with a `find` command such as:

```
# find /mnt/cdrom -type d -name '10.24.01'
/mnt/cdrom/data/packages/10.24.01
```

3.7 Enable Upgrade

3.7.1 Enable The Upgrade Repo, And Install The Upgrade Package

Enabling the upgrade repository and installing the upgrade package can be carried out for the various distributions as follows:

RHEL8 derivatives:

```
# yum-config-manager \
  --add-repo http://support.brightcomputing.com/upgrade/10.0/rhel/8/updates
# yum --nogpgcheck install cm-upgrade
```

RHEL9 derivatives:

```
# yum-config-manager \
--add-repo http://support.brightcomputing.com/upgrade/10.0/rhel/9/updates
# yum --nogpgcheck install cm-upgrade
```

SLES 15:

```
# zypper addrepo \
http://support.brightcomputing.com/upgrade/10.0/sles/15/updates cm-upgrade-10
# zypper install cm-upgrade
```

Ubuntu 20.04:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-10.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/10.0/ubuntu/2004/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

Ubuntu 22.04:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-10.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/10.0/ubuntu/2204/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

3.7.2 Load The Environment Module

```
# module load cm-upgrade/10.0
```

3.8 Perform Upgrade

The steps for carrying out the upgrade are as follows:

3.8.1 Power Off Nodes

- Power off regular nodes
- Terminate cloud nodes and cloud directors, and remove the cloud extension(s)

3.8.2 Apply Updates To Head Node

- RHEL derivatives:

```
# yum update
```

- SLES derivatives:

```
# zypper up
```

- Ubuntu:

```
# apt-get update
# apt-get upgrade --with-new-pkgs
```

3.8.3 Apply Updates To Software Images

Before updating and upgrading a software image, it may be worth creating an image backup by cloning the image (page 202 of the *Administrator Manual*). If the cluster after the upgrade has an image that has an issue, then comparing it against the backup can help troubleshoot the issue.

For each software image, updates can be applied as follows:

- RHEL derivatives:

```
# yum --installroot /cm/images/<software image> update
```

- SLES derivatives:

```
# zypper --root /cm/images/<software image> up
```

- Ubuntu:

```
# cm-chroot-sw-img /cm/images/<software image>
# apt-get update
# apt-get upgrade --with-new-pkgs
# exit
```

3.8.4 Apply Updates To The `node-installer` Image

- RHEL derivatives:

```
# yum --installroot /cm/node-installer update
```

- SLES derivatives:

```
# zypper --root /cm/node-installer up
```

- Ubuntu:

```
# cm-chroot-sw-img /cm/node-installer
# apt-get update
# apt-get upgrade --with-new-pkgs
# exit
```

3.8.5 Upgrade Head Nodes To BCM 10

Important: this must be run on both head nodes in a high availability setup. Recommended: Upgrade the active head node first and then the passive head node.

Depending on availability, the upgrade can be run either over the network, or using a DVD/ISO:

- Upgrade using repositories accessible over the network

```
# cm-upgrade
```

- Upgrade using a BCM DVD/ISO

```
# cm-upgrade -b /root/bcm-10.0-rocky8u8.iso
```

In an HA setup, after upgrading both the head nodes, resync the databases. Run the following from the active head node (it is very important to complete this step before moving to the next one):

```
# cmha dbreclone <secondary>
```

3.8.6 Reboot Head Node(s)

The head node(s) must be rebooted before proceeding to run the post-upgrade actions.

3.8.7 Post-upgrade Head Node Actions

Important: these must be run on both head nodes in an HA setup

```
# module load cm-upgrade/10

# cm-post-upgrade -m
```

3.8.8 Upgrade The Software Image(s) To BCM 10

Important: the upgrade to the software images must be done only on the active head node. Depending on availability, the upgrade can be run either over the network, or using a DVD/ISO:

- Upgrade using repositories accessible over the network

```
# cm-upgrade -i all
```

- Upgrade using a BCM DVD/ISO

```
# cm-upgrade -i all -b /root/bcm-10.0-rocky8u8.iso
```

If the software images are not under the standard location—`/cm/images/` on the head node—then the option `-a` should be used:

```
# cm-upgrade -a /apps/images -i <name of software image>
# cm-upgrade -a /apps/images -i <name of software image> -b /root/bcm-10.0-rocky8u8.iso
```

If the software images are located under multiple locations, the target locations can be specified as a comma-separated list:

```
# cm-upgrade -a '/apps/images,/opt/images' -i all
```

3.8.9 Upgrade /cm/node-installer To BCM 10

This needs to be performed only on the primary active head node, either over the network, or using a BCM DVD/ISO.

- Upgrade using repositories accessible over the network:

```
# cm-upgrade -x
```

- Upgrade using a BCM DVD/ISO:

```
# cm-upgrade -x -b /root/bcm-10.0-rocky8u8.iso
```

3.8.10 Post-upgrade Software Images Actions

Important: these must be run only on the active head node.

```
# cm-post-upgrade -i all
```

If the software images are not under the standard location, which is `/cm/images/` on the head node, then the option `-a` should be used:

```
# cm-post-upgrade -a /apps/images -i all
# cm-post-upgrade -a /apps/images -i <name of software image>
```

If the software images are located under multiple locations, then the target locations can be specified as a comma-separated list:

```
# cm-post-upgrade -a '/apps/images,/opt/images' -i all
```

3.8.11 (HA) Allow For The Provisioning Requests To The Secondary Head Node To Complete

Allow between 15 and 30 seconds for CMDaemon to schedule and start provisioning updates of `/tftpboot` and `/cm/node-installer` from the primary to the secondary head node. Then check and wait until the provisioning requests with the secondary head node as destination have completed using:

```
# cmsh -c 'softwareimage provisioningstatus -r'
```

If the provisioning requests with a destination node the secondary head node continue to be listed for a prolonged period of time and do not disappear from the list, then the `node-installer` and/or `tftpboot` on the secondary head node are out of sync from the primary head node, which can result in issues with provisioning of the compute nodes. Please contact support (`https://enterprise-support.nvidia.com/s/create-case`) for further assistance.

3.8.12 The Cluster Is Now Upgraded To BCM 10

It is recommended to provision the compute nodes at least once with BCM 10 before setting up any new integrations. In the case of Ubuntu base distro, it is recommended to initially perform a `FULL` install/provisioning of the compute nodes as noted in the preceding.

3.9 In-place Upgrade Of An Edge Director Or A Compute Node

The recommended way of upgrading the compute nodes is to upgrade the head node and software image(s) as described in section 3.8, and to allow the node-installer (when the compute nodes are booted) to re-provision the compute nodes with the upgraded software. For clusters with an edge setup, this usually means a new edge ISO needs to be generated on the head node, transferred to the edge site, and then the edge director node needs to be reconfigured to boot from the ISO, which will start the node-installer.

In some special cases, however, the administrator may elect to manually carry out an in-place upgrade of a compute node or edge director, instead of using the node installer to automatically upgrade the node. With the manual in-place upgrade, the administrator sets up the node to receive the updates directly from the software image (through `imageupdate`) without the involvement of the node installer.

Other than the cases where the node installer does not start on (re)boot by default, such as for the edge director, the in-place upgrade of a compute node/edge director does not offer any advantage over the standard approach to simply upgrade the head node and the software image(s).

3.9.1 Important Note About Services Running On The In-place Upgraded Compute Node/Edge Director

An in-place upgrade of a compute node will generally restart only the CMDaemon service. Any other service that may be affected by the upgrade of the packages, especially in the case of packages in `/cm/shared` (such as WLM), which are upgraded as part of the head node(s) upgrade, will need to be restarted manually by the administrator, or a reboot will be required.

It is therefore recommended to use the standard approach to upgrade the software image(s) on the head node, boot the nodes with the node installer, and allow the node installer to upgrade and configure the compute nodes as usual.

Nevertheless the administrator may find it more convenient in special cases such as edge sites, which have edge directors, to perform an in-place upgrade. The edge directors nodes are typically configured to boot from the local disk and by default do not start the node installer on (re)boot. An in-place upgrade of the edge directors will then allow the administrator to upgrade the software on the directors without transferring ISOs and reconfiguring the directors to boot from ISOs. To

address the issue with the running services, the edge directors can be rebooted at the end of the upgrade and while they will still not start the node installer, they will boot into the upgraded software.

3.9.2 In-place Edge Director Upgrade Procedure

Unless otherwise stated, all actions are performed locally on the edge director

Apply Updates To The Edge Director

- RHEL derivatives:

```
# yum update
```

- SLES derivatives:

```
# zypper up
```

- Ubuntu:

```
# apt-get update
# apt-get upgrade --with-new-pkgs
```

Back Up The File System

It is recommended to perform a file system backup of the edge director

Stop CMDaemon and `cm-nfs-checker` services

```
# systemctl stop cmd
# systemctl stop cm-nfs-checker
```

Upgrade The Head Node(s) As Described In The Preceding

Follow the steps to upgrade the head nodes, software image, and node installer as described in the preceding sections “Enable Upgrade” (section 3.7) and “Perform Upgrade” (section 3.8), and which are performed on the head node(s) of the cluster. Because CMDaemon on the edge director is stopped, the edge director does not need to be powered off while the upgrade of the head nodes takes place.

Upgrade CMDaemon To BCM 10

After the upgrade of the head nodes is completed, the package manager repositories files on the edge director must be manually updated to point to the BCM 10 repositories. This means at all occurrences of 9.2 in the repositories are replaced with with 10.0. Note: Leave the usernames such as in, for example, “`username=cm92user`”, unchanged.

Then, upgrade CMDaemon as follows:

- RHEL derivatives:

  Update the `/etc/yum.repos.d/cm.repo` file to point to the 10.0 repositories.

  On RHEL, typically `yum/dnf` can manage to resolve most of the dependencies, and the administrator can execute:

```
# yum clean all
# rpm -e --nodeps cm-config-cm
```

```
# yum install cmdaemon
```

- SLES derivatives:

  Update the /etc/zypp/repos.d/Cluster_Manager_Updates.repo file to point to the 10.0 repositories.

  Execute

  ```
  # zypper clean --all
  # zypper install cmdaemon
  ```

  A problem may be reported by zypper with upgrading cmdaemon to version 10.0, due to a requirement to install cm-config-cm version 10.0. Typically the first solution, Solution 1, offered by zypper can be selected, which is to deinstall a certain set of Bright packages (prefixed by cm9.2) in order to upgrade CMDaemon to version 10.0.

- Ubuntu:

  Update the /etc/apt/sources.list.d/cm.list file to point to the 10.0 repositories.

  Also update the /etc/apt/auth.conf.d/cm.conf file, to point to the 10.0 repositories.

  Create a new file /etc/apt/preferences.d/99-tmp-cmd-upgrade with content (without any space at the beginning of the lines):

  ```
  Package: *
  Pin: origin "updates.brightcomputing.com"
  Pin-Priority: 1001
  ```

  The administrator may also find it convenient to choose a geographically-close mirror in the cm.list file. But note that the local change will be overwritten to the settings in the software image, once imageupdate is run in a step later on.

  Execute

  ```
  # apt-get update
  # apt-get install cmdaemon
  ```

Start CMDaemon

After CMDaemon has been upgraded to BCM 10, execute:

```
# systemctl daemon-reload
# systemctl start cm-nfs-checker
# systemctl start cmd
```

Update The Edge Director Image With imageupdate

On the head node, with cmsh, verify that the edge director is UP. If the edge director is not UP, then the in-place upgrade has not been successful, and the director will need to be upgraded by booting into to the node-installer.

With cmsh on the head node, execute imageupdate for the edge director. The administrator is advised to execute a dry-run first, then to use use the synclog command to review the result and verify no unexpected local data directories or files will be affected, and then to perform the real imageupdate:

```
cmsh% device use edge-director
cmsh% imageupdate
cmsh% synclog
cmsh% imageupdate -w
```

Update The Provisioners And Recreate The Ramdisks

Execute on the head node

```
cmsh% softwareimage updateprovisioners;
cmsh% fspart; trigger /tftpboot; trigger /cm/node-installer; trigger /cm/shared
```

Allow for the provisioning requests with a destination node the in-place upgraded director to complete by monitoring the provisioning status:

```
cmsh% softwareimage provisioningstatus -r
```

For the images served by the edge director, recreate the ramdisk

```
cmsh% softwareimage; createramdisk -d <image-name>
```

If needed, the in-place upgraded edge director can now also be rebooted. It is then upgraded and can provision the edge compute nodes.