Bright 9.2 Upgrade Manual

This document describes the procedure for upgrading Bright clusters to version 9.2. It should be read very carefully before carrying out an upgrade. The Bright Support team can be contacted (https://www.brightcomputing.com/support) if there are Bright upgrade issues for which the documentation is insufficient.

Supported Bright Versions and Linux Distributions

Upgrades from the following Bright versions are supported:

- Bright 9.1
- Bright 9.0
- Bright 8.2

The following Linux distributions are supported:

- RedHat Enterprise Linux 7 (update 7.8 and higher)
- CentOS Linux 7 (update 7.8 and higher)
- RedHat Enterprise Linux 8 (update 8.6 and higher)
- Rocky Linux 8 (update 8.6 and higher)
- SuSE Linux Enterprise Server 12 (SLES12 SP5) and higher
- SuSE Linux Enterprise Server 15 (SLES15 SP3) and higher
- Ubuntu 18.04
- Ubuntu 20.04

Parallel upgrades

Bright now offers the possibility to perform the so-called parallel upgrade, as opposed to the in-place upgrades, where 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.

Important note about parallel upgrades and configuration changes or services on the production cluster

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 shutdown 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.

Important note about parallel upgrades and local data on the head node

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.

Important note about parallel upgrades and mounted network file systems

If the users' home directories are on a file server, then a manual sync of the data as pointed above will not be necessary, 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 Bright Support.

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 setup a new or temporary file server to host the required for the parallel upgrade copy of /cm/shared.

Important note about parallel upgrades and cluster extensions

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.

Important note about parallel upgrades and edge setup

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

Prerequisites for parallel upgrades

- The cluster must have a previously installed and active subscription license. Sites with hardware life licenses should contact Bright Support. Before starting with the upgrade, the subscription licenses 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.
- 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 shutdown 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 still require access to

the Bright package repositories. Alternatively - the Bright ISOs can be used as a source for the Bright packages. Access to the base distro package repositories, e.g. via Internet or local repos, is also required.

- 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.

Important note about workload managers (WLM)

Torque, SGE and PBSPro CE workload managers are not supported in Bright 9.2, and hence the relevant packages will be removed as part of the upgrade.

Unversioned WLM packages (such as pbspro/pbspro-client/pbsoro-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.

Important information when upgrading from Bright 8.2

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

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

Important information about Slurm upgrades

Slurm versions older than Slurm 21.08 are not supported in Bright 9.2. When upgrading from Bright 9.1 or 9.0, Slurm must be upgraded to Slurm 21.8 before moving to Bright 9.2.

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

Important information about PBS Pro upgrades

PBS Pro versions older than PBS Pro 2021 are not supported in Bright 9.2. When upgrading from Bright 9.1 or 9.0, PBS Pro must be upgraded to PBS Pro 2021 or higher before moving to Bright 9.2. Below is an overview of the steps that must be taken for upgrading 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 )
  ```

Important information about Kubernetes package upgrades

Unsupported versions of Kubernetes packages must be removed manually after the main upgrade has completed.

- Remove the old packages from the headnodes

```
$ yum/zypper/apt remove cm-kubernetes*
```

- 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*
```

For more details please refer chapter 4 of the Bright 9.2 Containerization Manual.

Important information about Juypterhub database upgrade

The jupyterhub database schema must be manually upgraded in each jupyterhub node after the main package upgrade has completed. Below are the commands needed to 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
```

Important information about Juypter kernels after upgrade

Jupyter kernel templates in Bright 9.2 are based on python 3.9. Kernel templates in Bright 9.1 based in python 3.7 are no longer available. Jupyter kernels created with python 3.7 must be recreated with python 3.9 via the jupyter kernel creator extension after the main package upgrades have been completed. For more details please refer section 18.5 of the Bright 9.2 Administrator Manual.

Important note about package upgrades

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 repositories of the latest version of Bright (in this case 9.2 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.

Important note about upgrading Ubuntu clusters

After upgrading the cluster, the Ubuntu compute nodes must initially be re-provisioned with 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: 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.

Important note about OpenStack integration

- Bright OpenStack deployments must be removed (using cm-openstack-setup) before upgrading. OpenStack integration is no longer supported in Bright 9.2, 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 Bright Support for further assistance.

Other prerequisites

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 SLES11) 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.

- If the /cm/shared will be unmounted during the upgrade (i,e if an in-place upgrade is not being performed), then please make sure that the contents of the local /cm/shared are in sync with the remote copy.
- Kubernetes deployments from older Bright versions must be removed before upgrading and re-deployed after the upgrade.
- Configurations of cluster extension to the cloud must be removed before upgrading and re-deployed after the upgrade.

Known issues

- Upgrading Ubuntu 18.04 Bright clusters, using the Bright DVD/ISO as the source of the Bright packages, can result in some of the WLM packages being removed instead of being upgraded to the Bright 9.2 version. In particular, slurm19 and pbspro19 WLM packages are affected. The administrator is advised to either use the Bright Internet package repositories or, after the upgrade of the cluster is completed, to download the relevant Bright 9.2 packages from the Bright Internet package repositories and install them manually. The issue where packages can be removed occurs since the Bright DVD/ISO has only a limited set of packages and some of the packages can be found only in the Bright Internet repositories (or a mirror).
- On some base distributions, depending on the version of the installed packages, when starting cm-upgrade may print a YAMLLoadWarning message when loading its configuration files. The warning message can be safely ignored.
- Disabling Kubernetes as part of the parallel-upgrade procedure results in an apparent failure in the stage for cleaning of the etcd data 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 to skip the failed stage. 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 left-over data from before the upgrade is cleaned during the provisioning.
- In some cases, in particular HA setup with a shared storage, it is possible 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 slur-

mdbd 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 setup 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 "no-version" module files (such as slurm/slurm/no-version module file). To resolve the issue, the left-over "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 Bright 9.2 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 is the openblas CM package, which in Bright 9.2 is replaced by the cm-openblas CM package. The issue occurs since the base distro also can provide a (non-CM) openblas package. If cm-upgrade reports the Bright CM openblas package is not upgraded, 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 Bright 9.2 versions, in some cases - notably on Ubuntu, it is also possible some packages such as cmburn, hdf518, or hpl are left in their 9.0 versions. The administrator is advised to manually swap the cm9.0 packages (for example cmburn_8.0-357-cm9.0) with the cm9.2 packages (for example cmburn_8.0-356-cm9.2).

Upgrading using a Bright DVD/ISO

When using a Bright DVD/ISO to perform the upgrade, it is important to use a DVD/ISO that is not older than **9.2-8**. 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 '9.2-*'

/mnt/cdrom/data/packages/9.2-8
```

Preparing the parallel upgrade setup

These steps need to be performed only for parallel upgrades and should be skipped for in-place upgrades. The steps marked with (HA) need to be performed 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) Shutdown 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, the MAC address of this spare node, and in the case of HA - the MAC address of the secondary head node, i.e. for the non-HA case:

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

or 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.

- 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 above-made copy of /cm/shared 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 above), just run:

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

- 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 above.

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 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 Bright Support 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 above

The parallel setup is now ready for an upgrade

Enable upgrade

Enable the upgrade repo and install the upgrade package

- RHEL7 derivatives:

  ```
  # yum-config-manager \
        --add-repo http://support.brightcomputing.com/upgrade/9.2/rhel/7/updates
  # yum --nogpgcheck install cm-upgrade-9.2
  ```

- RHEL8 derivatives:

  ```
  # yum-config-manager \
        --add-repo http://support.brightcomputing.com/upgrade/9.2/rhel/8/updates
  # yum --nogpgcheck install cm-upgrade-9.2
  ```

- SLES12:

  ```
  # zypper addrepo \
        http://support.brightcomputing.com/upgrade/9.2/sles/12/updates cm-upgrade-9.2
  # zypper install cm-upgrade-9.2
  ```

- SLES15:

  ```
  # zypper addrepo \
        http://support.brightcomputing.com/upgrade/9.2/sles/15/updates cm-upgrade-9.2
  # zypper install cm-upgrade-9.2
  ```

- Ubuntu1804:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-92.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/9.2/ubuntu/1804/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

- Ubuntu2004:

```
# cat <<EOF > /etc/apt/sources.list.d/cm-upgrade-92.list
deb [trusted=yes] https://support.brightcomputing.com/upgrade/9.2/ubuntu/2004/ ./
EOF
# apt-get update
# apt-get install cm-upgrade
```

Load the environment module

```
# module load cm-upgrade/9.2
```

Perform upgrade

Power off nodes

This step needs to be performed only for in-place upgrades

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

Apply updates to head node

- RHEL derivatives:

```
# yum update
```

- SLES derivatives:

```
# zypper up
```

- Ubuntu:

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

Apply updates to software images

For each software image, do the following:

- 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
  ```

On Bright 9.0 and higher, apply updates to the node-installer image

- RHEL derivatives:

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

  A known issue on RHEL8/Centos8 is that and update of the packages in the node-installer image can result in the installation of the pgi compilers from the Bright repositories as the package manager is trying to resolve the package dependencies. To work around this, on RHEL8 the administrator can first install the libibverbs package (using "yum –installroot /cm/node-installer install libibverbs"), and then upgrade the packages as shown with the above example yum command. If the Bright pgi package has already been installed during a previous update of the packages, the administrator can install libibverbs and then safely remove the pgi package. There is no known side effect of the installation of the pgi package in the node-installer image, other than that it enlarges the size of the image.

- 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
  ```

Upgrade head nodes to Bright 9.2

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.

- Upgrade using repositories accessible over the network

  ```
  # cm-upgrade
  ```

- Upgrade using a Bright DVD/ISO

  ```
  # cm-upgrade -b /root/bright9.2-centos7u9.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>
```

Reboot head node(s)

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

Post upgrade head node

Important: this must be run on both head nodes in a high availability setup

```
# module load cm-upgrade/9.2

# cm-post-upgrade -m
```

Upgrade the software image(s) to Bright 9.2

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

- Upgrade using repositories accessible over the network

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

- Upgrade using a Bright DVD/ISO

  ```
  # cm-upgrade -i all -b /root/bright9.2-centos7u9.iso
  ```

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.

```
Examples:
# cm-upgrade -a /apps/images -i <name of software image>
# cm-upgrade -a /apps/images -i <name of software image> -b /root/bright9.2-centos7u9.iso
```

Upgrade /cm/node-installer to Bright 9.2

This needs to be performed only on the primary active head node

- Upgrade using repositories accessible over the network

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

- Upgrade using a Bright DVD/ISO

  ```
  # cm-upgrade -x -b /root/bright9.2-centos7u9.iso
  ```

Post upgrade software images

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

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

(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 destination node the secondary head node 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 Bright Support for further assistance.

The cluster is now upgraded to Bright 9.2

It is recommended to provision the compute nodes at least once with Bright 9.2 before setting up any new integrations. In the case of Ubuntu base distro, it is recommended to perform initially a FULL install / provisioning of the compute nodes as noted above.

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 above, 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 instead of using the node installer to automatically upgrade the node, to perform manually an in-place upgrade of a compute node or edge director, where 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).

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, resp. 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.

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 above

Follow the steps to upgrade the head nodes and the software image as described in the above sections "Enable upgrade" and "Perform upgrade", which is 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 Bright 9.2

After the upgrade of the head nodes is completed, on the edge director update manually the package manager repositories files to point to the Bright 9.2 repositories: replace all occurrences of "8.2" or "9.0", resp. when upgrading from Bright 8.2 or 9.0, with "9.2". Note to leave untouched the usernames, which may have "82" or "90" such as in "username=cm90user". Then upgrade cmdaemon:

- RHEL derivatives:

  Update the /etc/yum.repos.d/cm.repo file to point to the 9.2 repositories

On RHEL8, typically yum/dnf can manage with resolving the dependencies and the administrator can simply execute:

```
# yum clean all
# yum install cmdaemon
```

On RHEL7, usually the package manager requires additional hints how to resolve the dependencies. "yum install cmdaemon" will report some packages could not be upgraded. In this case, the administrator can work around the issue by using the yum distro-sync command and providing on the command line the list of all Bright packages that were reported by yum with Errors (typically these are cm-luajit and mysql++)

```
# yum clean all
# yum distro-sync cmdaemon cm-luajit mysql++
```

- SLES derivatives:

  Update the /etc/zypp/repos.d/Cluster_Manager_Updates.repo file to point to the 9.2 repositories

  Execute

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

  zypper may report a problem with upgrading cmdaemon to version 9.2 due to a requirement to install cm-config-cm version 9.2. Typically the first Solution 1 offered by zypper to deinstall a certain set of Bright (cm8.2 or cm9.0) packages in order to upgrade cmdaemon to version 9.2 can be selected.

  A known issue on Bright 8.2 SLES12 is a possible conflict between cm-dhcp and dhcp packages. It is safe to answer "yes" to replace the conflicting dhcp files with files from cm-dhcp.

- Ubuntu:

  Update the /etc/apt/sources.list.d/cm.list file to point to the 9.2 repositories

  Update also the /etc/apt/auth.conf.d/cm.conf file to point to the 9.2 repositories

  Create a new file /etc/apt/preferences.d/99-tmp-cmd-upgrade with content (without any space in the begining 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 executed below.

Execute

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

Start cmdaemon

After cmdaemon has been upgraded to Bright 9.2, execute

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

Image update the edge director

On the head node, with cmsh verify 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 first a dry-run, then 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.