EDF HPC

HPC System: Puppet Configuration

Date: Pages: 22

Authors: CCN-HPC

Contents

Chapter 1

About this document

1.1 Purpose

This document contains a generic description of an HPC system in terms of its architectural views.

1.2 Typographic conventions

- Files or directories names are written in italics: */admin/restricted/config-puppet*.
- Hostnames are written in bold: **genbatch1**.
- Groups of hostnames are written using the nodeset syntax from clustershell. For example **genbatch[1-2]** refers to the servers **genbatch1** and **genbatch2**.
- Commands, configuration files contents or source code files are written in the format below:

```
$ cp /etc/default/rcS /tmp
```

1.3 License

1.4 Authors

In alphabetical order:

- Thomas HAMEL

Chapter 2

Dependencies

2.1 Debian

```
# apt-get install clustershell python-yaml
```

2.2 Puppet

This is a list of current puppet Module dependencies not provided.

- puppetlabs-stdlib (debian: puppet-module-puppetlabs-stdlib)
- puppetlabs-concat (debian: puppet-module-puppetlabs-concat)
- puppetlabs-apache (debian: puppet-module-puppetlabs-apache)

  stdlib is special, because for "historical" reasons, it is in the current git archive. Installing and using `hiera-eyaml` is not recomended:

```
# apt-get install hiera-eyaml
```

Chapter 3

Installation

Installing the configuration will depend of your cluster topology. This page only describes most simple.

3.1 Shared `/admin`

In this setup, a storage space is mounted on every nodes of the cluster and the configuration is applied directly from this storage space. By default this space is mounted on `/admin`, using another mount point should not be difficult.

On simple systems, it is possible to use an NFS server to make `/admin` available on all nodes. It is also possible to bootstrap the cluster with a `/admin` on the *Admin Server* exported by NFS and later move it to a more resilient location (HA NFS, CephFS or GPFS).

3.1.1 Directory layout

The layout setup should be done on the first node with `/admin` available. This is generally the *Admin Server*.

- `/admin`
- `restricted`
  - `puppet-hpc` (A git clone of the puppet-hpc repository)
  - `puppet-config`
  - `hieradata`
  - `hpc-privatedata` (Frequently another git repository)
  - `hieradata`
  - `files`
  - `hieradata`
  - `generic` (Symbolic link to `/admin/restricted/puppet-hpc/hieradata`)
  - `private` (Symbolic link to `/admin/restricted/hpc-privatedata/hieradata`)
  - `privatefiles` (Symbolic link to `/admin/restricted/hpc-privatedata/files`)
- `public`
  - `http`

3.1.2 Puppet

Puppet must be configured to search for the modules in the shared `/admin`. The following file can be used on debian and also search modules installed with debian packages:

```
[main]
logdir=/var/log/puppet
vardir=/var/lib/puppet
ssldir=/var/lib/puppet/ssl
rundir=/var/run/puppet
basemodulepath=$confdir/modules:/usr/share/puppet/modules:/admin/restricted/puppet-hpc/puppet
    -config/cluster:/admin/restricted/puppet-hpc/puppet-config/modules
prerun_command=/etc/puppet/etckeeper-commit-pre
```

```
postrun_command=/etc/puppet/etckeeper-commit-post
stringify_facts=false
hiera_config=/etc/puppet/hiera.yaml

[master]
# These are needed when the puppetmaster is run by passenger
# and can safely be removed if webrick is used.
ssl_client_header = SSL_CLIENT_S_DN
ssl_client_verify_header = SSL_CLIENT_VERIFY
```

3.1.3 Hiera-eyaml

It is recomended to use Hiera EYAML to store secret values. The keys must be created on the first node.

```
# mkdir /etc/puppet/secure
# cd /etc/puppet/secure/
# eyaml createkeys
[hiera-eyaml-core] Created key directory: ./keys
Keys created OK
# chown -R puppet:puppet /etc/puppet/secure/keys
# chmod -R 0500 /etc/puppet/secure/keys
# chmod 0400 /etc/puppet/secure/keys/*.pem
```

To configure `eyaml(1)` itself, the following file should be created: `/etc/eyaml/config.yaml`

```
---
pkcs7_private_key: '/etc/puppet/secure/keys/private_key.pkcs7.pem'
pkcs7_public_key: '/etc/puppet/secure/keys/public_key.pkcs7.pem'
```

Hiera is configured to search for values in the generic configuration repository, then in a few files for all nodes, then in files specific for each *role*.

```
:backends:
  - eyaml
:eyaml:
  :datadir:           /admin/restricted/hieradata
  :pkcs7_private_key: /etc/puppet/secure/keys/private_key.pkcs7.pem
  :pkcs7_public_key:  /etc/puppet/secure/keys/public_key.pkcs7.pem
  :extension:         'yaml'
:hierarchy:
  - private/default/roles/%{puppet_role}
  - generic/default/roles/%{puppet_role}
  - private/cluster
  - private/network
  - private/default
  - generic/common
  - generic/%{osfamily}/common
```

3.1.4 Node bootstraping

Setting up the directory layout can be done once, but you will still have to do some bootstraping on other newly installed nodes. Those steps will be handled by the bootsystem eventually.

The steps are:

- Distributing the puppet configuration
- Distributing the hiera configuration and keys
- Mounting `/admin`

Chapter 4

Cluster Definition

This cluster configuration is meant to be use with a standard cluster architecture, deviation from this architecture should be minimum. Some constraints are planned to be relaxed in future.

Here, we are going to describe this architecture and how it should be defined to be used by the configuration.

4.1 Architecture

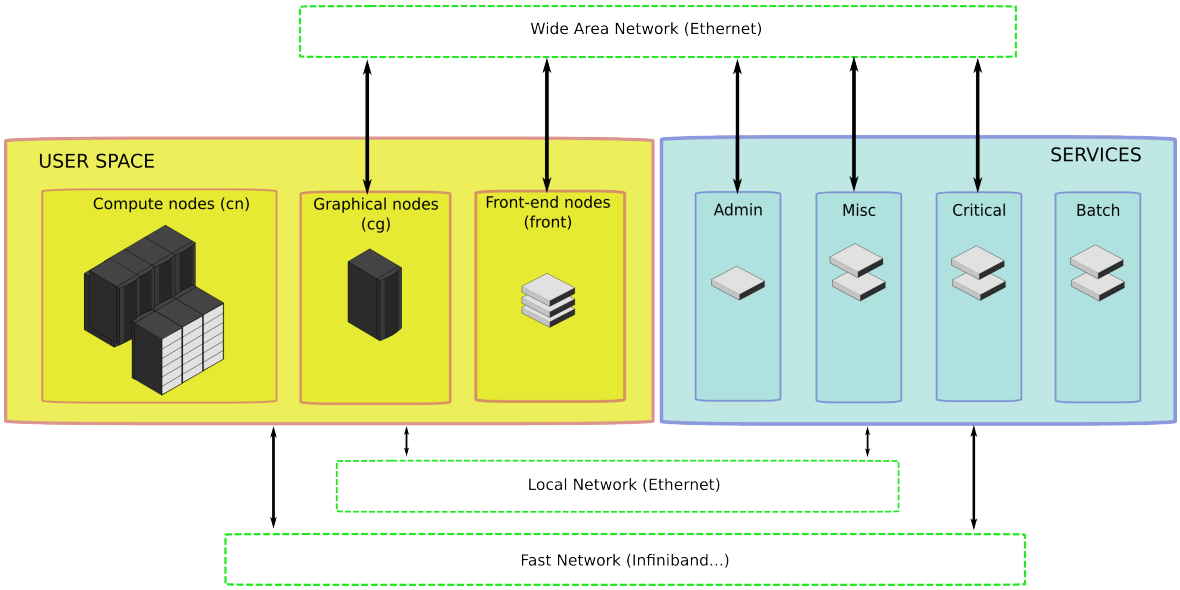

Figure 4.1:

4.2 Network definitions

4.2.1 Topology

Network topology is defined in the *cluster* level of the Hiera hierarchy. This means it is common to all nodes.

```
## Network topology of the cluster
net::allloc::ipnetwork: '172.16.0.0'
net::allloc::netmask: '255.255.0.0'
net::clusterloc::ipnetwork: '172.16.0.0'
net::clusterloc::netmask: '255.255.248.0'
net::clusterloc::prefix_length: '/21'
net::clusterloc::broadcast: '172.16.7.255'
net::clusterib::ipnetwork: '172.16.40.0'
net::clusterib::prefix_length: '/21'
```

```
net::clusteradm::ipnetwork: '172.16.80.0'
net::clusteradm::netmask: '255.255.240.0'
net::clusteradm::broadcast: '172.16.95.255'
net_topology:
    'wan':
        'name':           'WAN'
        'prefixes':       'wan'
        'ipnetwork':      '172.17.0.0.0'
        'netmask':        '255.255.255.0'
        'prefix_length':  '/24'
        'gateway':        '172.17.0.1'
        'broadcast':      '172.17.0.255'
        'ip_range_start': '172.17.0.1'
        'ip_range_end':   '172.17.0.254'
        'firewall_zone':  'wan'
    'allloc':
        'ipnetwork':      '172.16.0.0'
        'netmask':        '255.255.0.0'
    'clusterloc':
        'name':           'CLUSTER'
        'prefixes':       ''
        'ipnetwork':      '172.16.0.0'
        'netmask':        '255.255.248.0'
        'prefix_length':  '/21'
        'gateway':        '172.16.0.1'
        'broadcast':      '172.16.7.255'
        'ip_range_start': '172.16.0.1'
        'ip_range_end':   '172.16.7.254'
        'firewall_zone':  'clstr'
        'pool0':
            'ip_range_start':
                          '172.16.0.1'
            'ip_range_end':
                          '172.16.5.254'
        'pool1':          # IP reserved for the discovery process
            'ip_range_start':
                          '172.16.6.1'
            'ip_range_end':
                          '172.16.7.254'
    'clusterib':
        'name':           'IB'
        'prefixes':       'ib'
        'ipnetwork':      '172.16.40.0'
        'netmask':        '255.255.248.0'
        'prefix_length':  '/21'
        'gateway':         ''
        'broadcast':       '172.16.47.255'
        'ip_range_start':  '172.16.40.1'
        'ip_range_end':    '172.16.47.254'
        'firewall_zone':   'clstr'
    'clusteradm':
        'name':            'ADMIN'
        'prefixes':        'adm'
        'ipnetwork':       '172.16.80.0'
        'netmask':         '255.255.240.0'
        'prefix_length':   '/20'
        'gateway':         ''
        'broadcast':       '172.16.95.255'
        'ip_range_start':  '172.16.80.1'
        'ip_range_end':    '172.16.95.254'
        'firewall_zone':   'clstr'
    'mgmt':
        'name':            'MANAGEMENT'
        'prefixes':        'mgmt'
        'ipnetwork':       '172.16.80.0'
```

```
        'netmask':          '255.255.248.0'
        'prefix_length':    '/21'
        'gateway':          ''
        'broadcast':        '172.16.87.255'
        'ip_range_start':   '172.16.80.1'
        'ip_range_end':     '172.16.87.254'
        'firewall_zone':    'clstr'
```

4.2.2 Bonding

Some network interfaces are bonded together for load balancing and high availability. The bonding definition is done in hiera. If the bonding is uniform (same bond interface on same slaves interfaces) between nodes, this can be done at the *cluster* level. In case of differences between nodes, it must be done higher in the hierarchy (*role* or *node*).

```
bondcfg:
    'bond0':
      'slaves':
         - 'eth0'
         - 'eth1'
      'options':  'mode=active-backup primary=eth0'
    'bond1':
      'slaves':
         - 'eth2'
         - 'eth3'
      'options':  'mode=active-backup primary=eth2'
```

4.3 Node definitions

Nodes are defined in a hiera array called `master_network`. This structure is derived from an internal CSV file format. Each array "line" defines one node and its network configuration.

Each line consist of five comma separated lists of values, and three lists of associations between those values.

The value lists are:

- MAC addresses
- Interface devices
- Hostnames
- IPv4 Addresses
- IPv4 Net Masks

The associations reference each list with an index starting at 0. The associations are:

- DHCP Configuration, "MAC addresses" ←→"Hostnames" ←→"IPv4 Addresses"
- Node Configuration, "Interfaces devices" ←→"IPv4 Addresses" ←→"IPv4 Netmask"
- DNS/Hosts Configuration, "Hostnames" ←→"IPv4 Addresses"

Example:

```
master_network:
  #MAC_Addr;Interfaces;Hostnames;Addresses;NetMask;DHCP(Mac_Addr@Hostname@Address);Config(
    Interface@Address@Netmask);Hosts(Hostname@Address)
  - 52:54:00:ba:9d:ac,52:54:00:43:d9:45,52:54:00:8a:aa:30,52:54:00:8a:0b:d2;bond0,bond1;
    genmisc1,wangenmisc1;172.16.2.21,172.17.42.45;255.255.248.0,255.255.255.0;0@0@0;0@0@0,1
    @1@1;0@0,1@1
```

This example define one node (`genmisc1`) with the following configuration:

- DHCP
- `52:54:00:ba:9d:ac genmisc1 172.16.2.21`
- Network configuration on the node
- `bond0 172.16.2.21 255.255.248.0`
- `bond1 172.17.42.45 255.255.255.0`
- DNS and Hosts

- genmisc1 172.16.2.21
- wangenmisc1 172.17.42.45

All lists are optional, so it's possible to define element that just define a Hosts/DNS configuration (for virtual IP addresses for instance):

```
master_network:
  #MAC_Addr;Interfaces;Hostnames;Addresses;NetMask;DHCP(Mac_Addr@Hostname@Address);Config(
    Interface@Address@Netmask);Hosts(Hostname@Address)
  - ;;genmisc;172.16.2.20;;;;0@0
```

Chapter 5

Debugging

5.1 Facts

5.1.1 Listing facts

```
# puppet facts find $(hostname) --render-as=yaml
```

It's possible to use the -d (debug) flag with this command.

5.1.2 Debugging facts

This small script can be used to obtain a more precise error when a fact is failing:

```
#!/bin/bash

hpc_puppet_modules_dir=/admin/restricted/puppet-hpc/puppet-config/modules

for dir in /admin/restricted/puppet-hpc/puppet-config/modules/*
do
  FACTERLIB="$FACTERLIB:${dir}/lib/facter"
done

export FACTERLIB

facter "${@}"
```

All traditional facter flags are working:

```
# ./hpc_facter -p --trace
undefined method ‘empty?’ for nil:NilClass
/admin/restricted/puppet-hpc/puppet-config/modules/hpclib/lib/facter/network.rb:91:in ‘<top (
    required)>’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:130:in ‘load’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:130:in ‘kernel_load’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:115:in ‘load_file’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:49:in ‘block (2 levels) in load_all’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:47:in ‘each’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:47:in ‘block in load_all’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:45:in ‘each’
/usr/lib/ruby/vendor_ruby/facter/util/loader.rb:45:in ‘load_all’
/usr/lib/ruby/vendor_ruby/facter/util/collection.rb:104:in ‘load_all’
/usr/lib/ruby/vendor_ruby/facter.rb:126:in ‘to_hash’
/usr/lib/ruby/vendor_ruby/facter/application.rb:46:in ‘run’
/usr/bin/facter:9:in ‘<main>’
```

Chapter 6

Glossary

- **Admin Server**, Node the system administrators use to connect interactively to the cluster
- **Batch Server**, Server hosting services related to the job scheduling system
- **cg**, see *Graphical Node*
- **cn**, see *Compute Node*
- **Compute Node**, Node with a lot of CPU power and mempory to handle the actual jobs
- **Critical Server**, Server hosting critical services used by the nodes
- **Front-End Node**, Node where the users log in interactively to manage jobs, edit files, compile codes or do command line pre/post processing. This type of node is sometime called a *Login Node*.
- **front**, see *Front-End Node*
- **Graphical Node**, Node with a GPU to handle pre/post processing tasks or GPGPU jobs
- **Misc Server**, Server hosting miscellaneous services used by the nodes

Chapter 7

Operations - Clara

7.1 Init

7.1.1 Cluster Decrypt Password

A decrypt password is used by clara to decrypt files. Once you have generated this password, it should be in your hiera under this structure:

```
clara::password_options:
  ASUPASSWD:   "%{hiera('cluster_decrypt_password')}"
```

This password is used elsewhere in the hiera, generally under the name `decrypt_password` (for example: `opensshserver::decrypt_passwd`). So we define a top level variable (`cluster_decrypt_password`) to reuse it more easily.

7.1.2 Cluster keyring

The cluster must use a private cluster keyring. This keyring is used to sign packages generated locally and the local repositories.

You should generate it in your privatedata. You will be asked for a passphrase, this passphrase must be provided interactively when you call `clara repo add|del`. The following command can be pretty long to execute if you don't use a hardware Random Number Generator (RNG).

```
 # LANG=C gpg --no-default-keyring --keyring files/repo/cluster_keyring.gpg --secret-keyring
    files/repo/cluster_keyring.secret.gpg --gen-key                    gpg (GnuPG) 1.4.18;
     Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: keyring 'files/repo/cluster_keyring.secret.gpg' created
gpg: keyring 'files/repo/cluster_keyring.gpg' created
Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y
```

```
You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: HPC Team Example cluster
Email address: hpc@example.com
Comment:
You selected this USER-ID:
    "HPC Team Example cluster <hpc@example.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

passphrase not correctly repeated; try again.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
..+++++
..........+++++
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
+++++
.+++++
gpg: key 241FB865 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: public key of ultimately trusted key 1F2607DD not found
gpg: public key of ultimately trusted key 94DEFA86 not found
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   3  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 3u
pub   4096R/241FB865 2016-05-19
      Key fingerprint = D192 11C0 2EB6 BE80 A3BC  7928 1CB4 3266 241F B865
uid                  HPC Team Example cluster <hpc@example.com>
sub   4096R/C7027D3A 2016-05-19
```

Clara will use this key in its encrypted form, if you have a working `clara enc`, you can use `clara enc encode` directly. Otherwise you can use the following command:

```
$ openssl aes-256-cbc -in cluster_keyring.secret.gpg -out cluster_keyring.secret.gpg.enc -k <
    cluster decrypt password>
```

Chapter 8

Operations - MariaDB/Galera

8.1 Init/Start

You have to perform this operation anytime the cluster is completely down (first boot or full reboot).

```
# echo MYSQLD_ARGS=--wsrep-new-cluster > /etc/default/mysql
# systemctl start mysql
# rm /etc/default/mysql
```

Chapter 9

Operations - OpenLDAP

9.1 Replica

When you initialize your cluster, if you wish to use a local OpenLDAP replica, you have to execute the script `make_ldap_replica` on your replica node. This script will use an ldif file that you must provide.

9.2 Logging

9.2.1 Changing log level

To change the log level on a running server you must define a logging modification ldif file:

```
dn: cn=config
changetype: modify
replace: olcLogLevel
olcLogLevel: stats
```

The new level is applied with this command:

```
# ldapmodify -Q -Y EXTERNAL -H ldapi:/// -f /tmp/logging.ldif
```

Chapter 10

Operations - SlurmDBD

10.1 Init

After the batch server have been installed, you must create the cluster in slurmdbd:

```
# sacctmgr add cluster <cluster_name>
```

Chapter 11

Profiles - Clush

11.1 Groups

By default this profile will declare a new group source: `hpc_roles`. This source will give a group for each role in the configuration. This source is the default.

To define other groups, it's possible to define custom groups in hiera:

```
#### Clustershell
profiles::clustershell::groups:
  'cn':      '@hpc_roles:cn'
  'bm':      '@hpc_roles:bm'
  'cg':      '@hpc_roles:cg'
  'crit':    '@hpc_roles:critical'
  'misc':    '@hpc_roles:misc'
  'front':   '@hpc_roles:front'
  'admin':   '@hpc_roles:admin'
  'service': '@misc,@crit'
  'compute': '@cn,@cg,@bm'
  'all':     '@admin,@service,@compute'
```

Chapter 12

Profiles - DB

12.1 Hiera Configuration

THe MariaDB/Galera cluster should be configured in your hiera to have a working configuration:

```
###### Slurm DBD database ####
galera_base_name:             "%{hiera('cluster_prefix')}%{::my_db_server}"
mariadb::galera_conf_options:
  mysqld:
    binlog_format:            'ROW'
    default-storage-engine:   'innodb'
    innodb_autoinc_lock_mode: '2'
    query_cache_size:         '0'
    query_cache_type:         '0'
    bind-address:             '0.0.0.0'
    wsrep_provider:           '/usr/lib/galera/libgalera_smm.so'
    wsrep_cluster_name:       '"galera_cluster"'
    wsrep_cluster_address:    "\"gcomm://%{hiera('galera_base_name')}1,%{hiera('
    galera_base_name')}2\""
    wsrep_sst_method:         'rsync'
mariadb::mysql_root_pwd:      'GALERA_ROOT_PASSWORD_OVERRIDEME_IN_EYAML'
```

Chapter 13

Profiles - HA

13.1 role

This profile describes HA configuration that are applied to all machine of the same role. This is done by configuring two hiera hashes:

- for the virtual IP addresses (failover):

```
#### High-Availability Virtual IP addresses ######
profiles::ha::role_vips:
  clusterloc_misc:
    prefix:      "%{hiera('cluster_prefix')}"
    net_id:      'clusterloc'
    router_id:   '121'
    ip_address:  '10.100.2.20'
    auth_secret: 'SECRET'
  wan_misc:
    prefix:      "%{hiera('cluster_prefix')}"
    net_id:      'wan'
    router_id:   '122'
    ip_address:  '192.168.42.55'
    auth_secret: 'SECRET'
```

- One for the virtual servers (load balancing):

```
profiles::ha::role_vservs:
  wan_front_ssh:
    ip_address:        '192.168.42.57'
    port:              '22'
    vip_name:          'wan_front'
    real_server_hosts:
      - 'extgenfront1'
```

Chapter 14

Profiles - Jobsched

14.1 Hiera Configuration

A generic configuration is defined in “puppet-hpc/hieradata/common.yaml", in your own hiera files you could just redefine, the following values:

```
slurm_primary_server:         "%{hiera('cluster_prefix')}%{::my_jobsched_server}1"
slurm_secondary_server:       "%{hiera('cluster_prefix')}%{::my_jobsched_server}2"

slurmcommons::partitions_conf:
  - "Nodename=%{hiera('cluster_prefix')}cn01 CPUs=4 RealMemory=512 State=UNKNOWN"
  - "PartitionName=cn Nodes=%{hiera('cluster_prefix')}cn01 Default=YES MaxTime=INFINITE State
    =UP"

# FIXME: When we have reverse DNS, it should be
# StorageHost:       "%{hiera('galera_base_name')}"
slurmdbd_slurm_db_password: 'SLURM_PASSWORD_OVERRIDEME_IN_EYAML'
slurmdbd_slurmro_db_password: 'SLURMRO_PASSWORD_OVERRIDEME_IN_EYAML'

slurmdbd::main_conf_options:
  DbdHost:            "%{hiera('slurm_primary_server')}"
  DbdBackupHost:      "%{hiera('slurm_secondary_server')}"
  DbdPort:            '6819'
  SlurmUser:          "%{hiera('slurm_user')}"
  DebugLevel:         '3'
  AuthType:           'auth/munge'
  AuthInfo:           '/var/run/munge/munge.socket.2'
  LogFile:            "/var/log/slurm-llnl/slurmdbd.log"
  PidFile:            '/var/run/slurm-llnl/slurmdbd.pid'
  StorageType:        'accounting_storage/mysql'
  StorageHost:        'localhost'
  StorageUser:        'slurm'
  StoragePass:        "%{hiera('slurmdbd_slurm_db_password')}"

slurmdbd::dbd_conf_options:
  db:
    hosts:        'localhost'
    user:         'debian-sys-maint'
    password:     "%{hiera('mariadb::mysql_root_pwd')}"
  passwords:
    slurm:        "%{hiera('slurmdbd_slurm_db_password')}"
    slurmro:      "%{hiera('slurmdbd_slurmro_db_password')}"
  hosts:
    controllers: "%{hiera('slurm_primary_server')},%{hiera('slurm_secondary_server')}"
    admins:       "%{hiera('cluster_prefix')}admin1"
```

Last updated 2016-06-29 14:53:02 BST