

Pivotal HD Enterprise

Version 2.0

Installation and Administrator Guide

Rev: 01

Table of Contents

| | | |
|--------|--|----|
| 1 | Overview of PHD | 7 |
| 1.1 | Components | 7 |
| 1.2 | About Supported Pivotal HD Services | 9 |
| 1.2.1 | HDFS | 10 |
| 1.2.2 | YARN | 10 |
| 1.2.3 | ZooKeeper | 11 |
| 1.2.4 | HBase | 11 |
| 1.2.5 | Hive | 11 |
| 1.2.6 | HAWQ | 12 |
| 1.2.7 | PXF | 12 |
| 1.2.8 | Pig | 12 |
| 1.2.9 | Mahout | 12 |
| 1.2.10 | Flume | 13 |
| 1.2.11 | Sqoop | 13 |
| 1.2.12 | Oozie | 13 |
| 1.2.13 | Hamster | 14 |
| 1.2.14 | GraphLab | 14 |
| 2 | Installing PHD Using the CLI | 15 |
| 2.1 | Command Line Installation Features | 15 |
| 2.2 | Planning your Pivotal HD Cluster Deployment | 15 |
| 2.2.1 | Deployment Options | 16 |
| 2.2.2 | Installation Instructions | 16 |
| 2.2.3 | Best Practices for Selecting Hardware | 17 |
| 2.2.4 | Best Practices for Deploying Hadoop Services | 18 |
| 2.3 | Command Line Installation Overview | 19 |
| 2.4 | Pivotal HD Enterprise Prerequisites | 21 |
| 2.4.1 | Package Accessibility | 23 |
| 2.5 | Preparing the Admin Node | 24 |
| 2.5.1 | Install Pivotal Command Center | 24 |
| 2.5.2 | Install Pivotal HD and HAWQ | 26 |
| 2.6 | Cluster Configuration Files | 27 |
| 2.6.1 | Fetching Default Cluster Configuration Templates | 28 |
| 2.6.2 | Editing Cluster Configuration | 28 |
| 2.7 | Configuring HAWQ | 31 |
| 2.8 | Verifying the Cluster Nodes for Pivotal HD | 32 |
| 2.9 | Deploying the Cluster | 33 |
| 2.10 | Verifying a Cluster Installation | 35 |
| 2.10.1 | Verifying Service Status | 36 |
| 2.11 | Preparing the Cluster Nodes (Deprecated) | 36 |
| 2.11.1 | Preparing the Cluster Nodes for Pivotal HD | 37 |
| 2.12 | Pivotal HD Directory Layout | 38 |
| 2.13 | Running Sample Programs | 39 |

| | |
|---|----|
| 2.13.1 Testing Hadoop | 39 |
| 2.13.2 Testing HBase | 40 |
| 2.13.3 Testing HAWQ | 40 |
| 2.13.4 Testing Pig | 41 |
| 2.13.5 Testing Hive | 42 |
| 2.14 Creating a YUM EPEL Repository | 42 |
| 2.15 High Availability (HA) | 43 |
| 2.15.1 Setting up a New Cluster with HA | 44 |
| 2.16 Configuring GemFire XD | 47 |
| 2.16.1 Overview | 48 |
| 2.16.2 Service Roles/Ports | 48 |
| 2.16.3 Best Practices | 48 |
| 2.16.4 Enabling PRTS Services | 48 |
| 2.16.5 GemFire XD Notes | 49 |
| 2.16.6 Managing GemFire XD | 49 |
| 2.17 Installing SSL certificates | 50 |
| 3 Upgrading PHD Using the CLI | 51 |
| 3.1 Overview | 51 |
| 3.2 Quick Guide | 51 |
| 3.3 Prerequisites | 53 |
| 3.4 Upgrade Instructions | 55 |
| 3.5 Upgrade Syntax | 59 |
| 3.6 Changed Configuration Parameters and Files | 60 |
| 3.6.1 core-site.xml | 60 |
| 3.6.2 yarn-site.xml | 61 |
| 3.6.3 hdfs-site.xml | 63 |
| 3.6.4 mapred-site.xml | 64 |
| 3.6.5 httpfs-site.xml | 65 |
| 3.6.6 capacity-scheduler.xml | 65 |
| 3.6.7 hbase-site.xml | 65 |
| 3.6.8 hive-site.xml | 68 |
| 4 Administering PHD Using the CLI | 69 |
| 4.1 Managing a Cluster | 69 |
| 4.1.1 Starting a Cluster | 69 |
| 4.1.2 Stopping a Cluster | 70 |
| 4.1.3 Restarting a Cluster | 71 |
| 4.1.4 Reconfiguring a Cluster | 72 |
| 4.1.5 Add / Remove Services | 74 |
| 4.1.6 Add Hosts to Cluster | 74 |
| 4.1.7 Retrieving Configuration about a Deployed Cluster | 75 |
| 4.1.8 Listing Clusters | 75 |
| 4.1.9 Expanding a Cluster | 76 |
| 4.1.10 Shrinking a Cluster | 77 |
| 4.1.11 Decommissioning Nodes | 78 |
| 4.1.12 High Availability | 79 |
| 4.1.13 Security | 82 |

| | |
|--|----|
| 4.1.14 Uninstalling a Cluster | 82 |
| 4.2 Managing HAWQ | 83 |
| 4.2.1 Initializing HAWQ | 83 |
| 4.2.2 Starting HAWQ | 84 |
| 4.2.3 Stopping HAWQ | 84 |
| 4.2.4 Modifying HAWQ User Configuration | 84 |
| 4.2.5 Expanding HAWQ | 85 |
| 4.3 Managing Roles and Hosts | 85 |
| 4.3.1 Managing Locally | 85 |
| 4.3.2 Managing Remotely | 85 |
| 4.4 Pivotal HD Services Reference | 87 |
| 4.4.1 Overriding Directory Permissions | 87 |
| 4.4.2 Pivotal HD Users and Groups | 88 |
| 4.4.3 Pivotal HD Ports | 89 |
| 5 PHD FAQ (Frequently Asked Questions) | 91 |
| 5.1 Can I deploy multiple clusters from the same admin? | 91 |
| 5.2 Can I modify the topology (host to role mapping) of the cluster after the initial install? | 91 |
| 5.3 How do I reformat the namenode? | 91 |
| 5.4 Certain services such as hadoop-hdfs-namenode or hadoop-hdfs-datanode do not come up when I run "start cluster"? | 91 |
| 5.5 What group and users are created by Pivotal HD? | 91 |
| 5.6 What is the allowed time difference amongst the cluster nodes versus the admin node? | 92 |
| 5.7 Does PCC support simultaneous deployment of multiple clusters? | 92 |
| 5.8 Does PCC support hostname both in IP address and FQDN format? | 92 |
| 5.9 Can a node be shared between different clusters? | 92 |
| 5.10 I installed puppet-2.7.20 from the Puppet Labs repository but Pivotal HD does not work? | 92 |
| 5.11 How do I clean up the nodes if a cluster deployment fails? | 92 |
| 5.12 Will I lose my data if I uninstall the cluster? | 93 |
| 5.13 Will I lose my data if I upgrade the PHD/ADS stack through the stack import utility? | 93 |
| 5.14 Can I upgrade Pivotal Command Center while the clusters are functioning? | 93 |
| 5.15 How do I change the port used by Pivotal HD? | 93 |
| 6 PHD Troubleshooting | 94 |
| 6.1 Debugging Errors | 94 |
| 6.1.1 Pivotal HD Installation | 94 |
| 6.1.2 Cluster Deployment | 94 |
| 6.1.3 Cluster Nodes Installation | 95 |
| 6.1.4 Services Start | 95 |
| 6.2 Puppet SSL Errors | 95 |
| 6.3 Upgrade/Reconfigure Errors | 96 |
| 6.3.1 Following an upgrade of Command Center, unable to Start/Stop cluster with invalid hostnames | 96 |
| 6.3.2 Other Upgrade/Reconfigure Errors | 97 |
| 6.4 HA-related Errors | 97 |
| 6.5 Other Errors | 98 |
| 6.5.1 Cluster Deployment Fails due to RPM Dependencies | 98 |
| 6.5.2 Unable to access the Namenode Status Web page | 98 |

| | | |
|--------|--|-----|
| 6.5.3 | Installation Fails due to Directory Permissions | 98 |
| 6.5.4 | Deployment Fails due to Problems with YUM Repository | 98 |
| 6.5.5 | Installation Fails due to Problems with the SSL certificate | 98 |
| 6.5.6 | Cluster Node Installation Failure without Generating a Log File | 99 |
| 6.5.7 | Puppet certificate failure | 99 |
| 6.5.8 | Package Bundle Not Found | 99 |
| 6.5.9 | Cluster Deployment Fails due to Missing Packages | 99 |
| 6.5.10 | Working with Proxy Servers | 100 |
| 6.5.11 | Capital Letters in Hostname | 101 |
| 6.5.12 | Resolving postgres port Conflict Issue | 101 |
| 6.5.13 | Resolving HTTP Port Conflict | 101 |
| 6.5.14 | Errors like Ambit: Push Failed | 101 |
| 6.5.15 | Preparehosts Errors Out While Creating gpadmin User | 102 |
| 6.5.16 | HAWQ Initialization Failing | 102 |
| 6.5.17 | Installing HAWQ on Dirty Cluster Nodes Previously Configured with HAWQ | 102 |
| 6.5.18 | Errors Related to VM Memory | 103 |

Copyright © 2014 GoPivotal, Inc. All rights reserved.

GoPivotal, Inc. believes the information in this publication is accurate as of its publication date. The information is subject to change without notice. THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." GOPIVOTAL, INC. ("Pivotal") MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any Pivotal software described in this publication requires an applicable software license.

All trademarks used herein are the property of Pivotal or their respective owners.

Use of Open Source

This product may be distributed with open source code, licensed to you in accordance with the applicable open source license. If you would like a copy of any such source code, Pivotal will provide a copy of the source code that is required to be made available in accordance with the applicable open source license. Pivotal may charge reasonable shipping and handling charges for such distribution.

1 Overview of PHD

Pivotal HD Enterprise is an enterprise-capable, commercially supported distribution of Apache Hadoop packages targeted to traditional Hadoop deployments.

This overview describes each component and how it fits into the overall architecture of Pivotal HD Enterprise.

1.1 Components

Pivotal HD Enterprise is a commercially-supported distribution of the Apache Hadoop stack including the following:

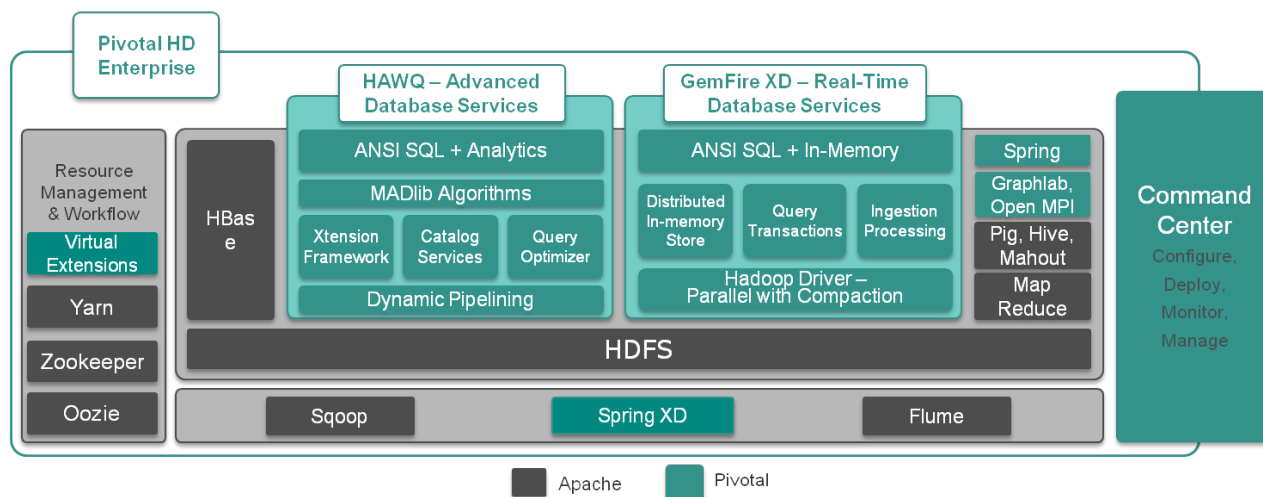
- **Core Apache Stack:**
 - Hadoop
 - HDFS
 - YARN
 - Zookeeper
 - HBase
 - Hive
 - Pig
 - Mahout
 - Flume
 - Sqoop
 - Oozie

Pivotal HD Enterprise enriches the Apache stack distribution by providing the following:

- **Advanced Database Services**
 - **HAWQ** - HAWQ adds SQL's expressive power to Hadoop. By adding rich, proven parallel SQL processing facilities, HAWQ renders queries faster than any other Hadoop-based query interface.
 - **PXF** - Extensibility layer to provide support for external data formats such as HBase and Hive.
- **Pivotal Command Center** - Pivotal Command Center (PCC) Is a Web-based interface for configuration and deployment of clusters, and for monitoring & management of a Pivotal HD environment. With the help of PCC, system administrators can determine if the PHD cluster is running efficiently, quickly diagnose functional or performance issues, and performs cluster management tasks when required.
- **PRTS - Pivotal Real Time Services** - Pivotal HD 2.x includes support for GemFire XD (GFXD), an offering of PRTS. For further information about GemFire XD installation and configuration; refer to the section [Configuring GemFire XD](#) .
- **Hamster** - Developed by Pivotal, Hamster is a framework which enable users running MPI programs on Apache Hadoop YARN platform. (OpenMPI is a A High Performance Message Passing Library.)

- **GraphLab** - GraphLab is a powerful new system for designing and implementing parallel algorithms in machine learning, it is a graph-based, high performance, distributed computation framework written in C++ that makes use of MPI and has its own programming model .

Pivotal HD Architecture



Pivotal Command Center (PCC) includes a CLI (command line interface) and a GUI. You can deploy and configure most of the Hadoop services as well as HAWQ, and PXF, using either the CLI or the GUI. You can start and stop the clusters using either the CLI or the GUI.

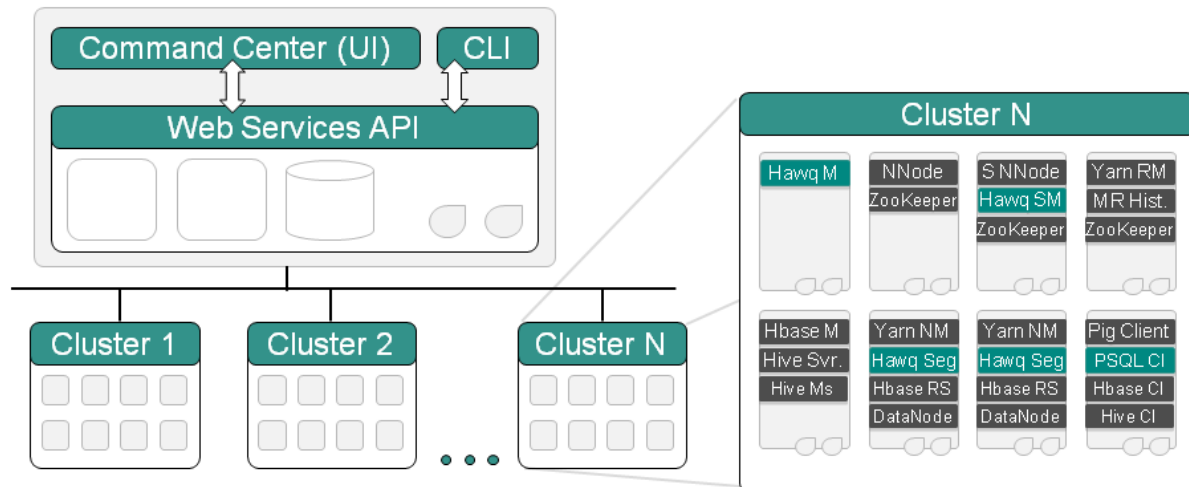


This documentation covers operations performed via the CLI. For Pivotal Command Center GUI operations; including configuring and deploying clusters, see the *Pivotal Command Center 2.1 User Guide*.

PCC stores the metadata for Hadoop cluster nodes and services, the cluster configuration and the system metrics in a PostgreSQL database.

See [Deployment Options](#).

Pivotal Command Center - Architecture



© Copyright 2013 Pivotal. All rights reserved.

Pivotal

2

1.2 About Supported Pivotal HD Services

The following services can be deployed and configured via Pivotal Command Center CLI, or manually.

- [HDFS](#)
- [YARN](#)
- [ZooKeeper](#)
- [Hbase](#)
- [Hive](#)
- [HAWQ](#)
- [PXF](#)
- [Pig](#)
- [Mahout](#)

The following services can only be deployed and configured manually (see the *Pivotal HD Enterprise 2.0 Stack and Tool Reference Guide* for details)

- [Flume](#)
- [Sqoop](#)
- [Oozie](#)
- [Hamster](#)
- [GraphLab](#)

1.2.1 HDFS

HDFS is a fault tolerant distributed file system which is designed to run on commodity hardware.

The following table shows HDFS service roles:

| Role Name | Description |
|--------------------|---|
| NameNode | The NameNode serves as both directory namespace manager and "inode table" for the Hadoop File System (HDFS). Each HDFS deployment must have a running NameNode. |
| Secondary NameNode | The Secondary NameNode periodically downloads the current NameNode image and edits log files. It joins them into a new image and uploads the new image back to the primary NameNode. |
| DataNodes | A DataNode stores data in the HDFS. A functional filesystem has more than one DataNode, with data replicated across all nodes. |
| Hadoop Client | A client machine has Hadoop installed with all the cluster settings, but is not a Master or Slave. Instead, the role of the client is to load data into the cluster, submit Map Reduce jobs that describe how to process the data, and then retrieve or view the results of the finished job. |
| *Journalnodes | A group of daemons to maintain the namenode edits information. These are used by both active and standby namenodes in a HA enabled cluster to keep their state synchronized. |
| *Standby Namenode | Namenode running on a different host in standby mode in a HA enabled cluster. This will take over as the active namenode if the current active namenode fails. |

*Only applicable for HA enabled clusters.

1.2.2 YARN

YARN is a framework that facilitates writing distributed processing frameworks and applications and supports MapReduce version 2.

The following table shows YARN service roles:

| Role Name | Description |
|------------------|--|
| Resource Manager | The ResourceManager is the master that manages all the cluster resources running on the YARN system. |
| Node Manager | The NodeManager manages resources on a particular node. |
| History Server | The History Server stores a history of the mapreduce jobs run on the cluster. |

1.2.3 ZooKeeper

Zookeeper is a centralized service that enable distributed synchronization and manages configuration across a cluster.

The following table shows ZooKeeper service roles:

| Role Name | Description |
|------------------|--------------------------|
| Zookeeper Server | ZooKeeper Quorum Servers |

1.2.4 HBase

HBase is a distributed, column-oriented database that uses HDFS for storing data.

The following table shows HBase service roles:

| Role Name | Description |
|--------------------|---|
| HBase Master | The Master server is responsible for monitoring all RegionServer instances in the cluster, and is the interface for all metadata changes. |
| HBase RegionServer | It is responsible for serving and managing regions which typically coexist with datanodes. |
| HBase Client | It is responsible for accessing HBase service. |

Notes

- HBase requires that you have installed HDFS, YARN, and Zookeeper.
- Pivotal HD installs ZooKeeper if you have not installed it.
- HBase does not manage the Zookeeper service.

1.2.5 Hive

Hive is a [data warehouse](#) infrastructure that provides an interface similar to SQL on top of Hadoop.

| Role Name | Description |
|----------------|--|
| Hive Metastore | The metastore stores the metadata for all Hive tables and partitions. Postgres database is used as the datastore |
| Hive Server | Also known as thrift server, is used by clients written in Java, C++ etc to access Hive |
| Hive Client | This is a launcher or gateway node which is used to launch hive jobs |

Note: Hive requires HDFS and YARN.

1.2.6 HAWQ

HAWQ is a parallel SQL query engine that marries Pivotal Analytic Database (Greenplum) and Hadoop 2.0 and is optimized for analytics, with full transaction support. The following table shows HAWQ service roles:

| Role Name | Description |
|--------------------|--|
| HAWQ Master | Stores the top-level metadata, as well as building the query plan |
| HAWQ StandbyMaster | This is a standby for the HAWQ Master |
| HAWQ Segments | Manages a shard of each table which typically coexist with datanodes |

Note: HAWQ requires HDFS.

1.2.7 PXF

PXF is an extended framework that combines the Pivotal Analytic Database engine (HAWQ) with enterprise class Apache Hadoop, HBase and Hive. The PXF service runs as a java agent on existing Hadoop, HBase and Hive nodes and enables HAWQ to consume data created by the external services.

Note : PXF requires HDFS and HAWQ.

If you do not install PXF via the CLI, and choose to install it later, refer to the *HAWQ 1.2 Administrator Guide* for details.

1.2.8 Pig

Pig is a data flow language used in the analysis of large data sets using mapreduce.

| Role Name | Description |
|------------|---|
| Pig Client | This is a launcher or gateway node which is used to launch Pig jobs |

Note : Pig requires HDFS and YARN/MapReduce..

1.2.9 Mahout

Mahout provides a collection of distributed [machine learning](#) algorithms on [Hadoop](#)

| Role Name | Description |
|---------------|--|
| Mahout Client | This is a launcher or gateway node which is used to launch Mahout jobs |

Note : Mahout requires HDFS and YARN/MapReduce.

1.2.10 Flume

Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data. It has a simple and flexible architecture based on streaming data flows. It is robust and fault tolerant with tunable reliability mechanisms and many failover and recovery mechanisms. It uses a simple extensible data model that allows for online analytic application.

| Role Name | Description |
|--------------|---|
| Flume Agent | Provide Flume service for generating, processing, and delivering data |
| Flume Client | This is a launcher or gateway node which is used to launch Flume jobs |

Note : Flume requires HDFS and YARN/MapReduce..

1.2.11 Sqoop

Sqoop is a tool designed for efficiently transferring bulk data between [Apache Hadoop](#) and structured datastores such as relational databases.

| Role Name | Description |
|-----------------|---|
| Sqoop Metastore | Provide shared metadata repository for Sqoop |
| Sqoop Client | This is a launcher or gateway node which is used to launch sqoop jobs |

Note : Sqoop requires HDFS, YARN/MapReduce and HBase.

1.2.12 Oozie

Oozie is a workflow scheduler system to manage Apache Hadoop jobs.

| Role Name | Description |
|-----------------|---|
| Oozie Metastore | provide Oozie service |
| Oozie Client | This is a launcher or gateway node which is used to launch Oozie jobs |

Note : Oozie requires HDFS, YARN/MapReduce , Pig(optional) and Hive(optional).

1.2.13 Hamster

Hamster is a framework that enables users running MPI programs on Apache Hadoop YARN platform.

1.2.14 GraphLab

GraphLab is a powerful new system for designing and implementing parallel algorithms in machine learning, it is a graph-based, high performance, distributed computation framework written in C++ that makes use of MPI and has its own programming model .

2 Installing PHD Using the CLI

This section describes how to install and configure Pivotal HD using command line interface (CLI) of Pivotal Command Center (PCC).

2.1 Command Line Installation Features

Using Pivotal Command Center's CLI to install Pivotal HD provides the following functionality:

| Feature | Support |
|----------------------------|--|
| Checking prerequisites | <ul style="list-style-type: none">Checks that specified hosts meet the prerequisites to install the supported components. |
| Supported cluster services | <ul style="list-style-type: none">Installs and configures Hadoop, YARN, ZooKeeper, HBase, Mahout, HAWQ, PXF, Hive, and Pig with default settings.Reconfigures the supported cluster services.Multi-cluster support.Monitors clusters with Pivotal Command Center. |
| Starting and stopping | <ul style="list-style-type: none">Starts and stops the cluster or individual services.Ensures that all dependent services start and stop in the correct order. |
| Logging | <ul style="list-style-type: none">Provides installation data logs. |
| Uninstallation | <ul style="list-style-type: none">Uninstalls individual services and Pivotal HD Enterprise. |

2.2 Planning your Pivotal HD Cluster Deployment

To deploy a Hadoop cluster, Pivotal recommends that you consider the following:

- Select the appropriate hardware configuration for cluster & management nodes.
- Map Hadoop services roles to cluster nodes.
- Configure the roles to effectively leverage underlying hardware platform.

2.2.1 Deployment Options

The following table illustrates the deployment options and limitations:

| Component | | CLI install | Manual install (rpm) |
|-----------------------------------|------|-------------|----------------------|
| Command Center (installs the CLI) | | | ✓ |
| Hadoop MR2: HDFS, YARN | | ✓ | ✓ |
| Pig | | ✓ | ✓ |
| Hive | | ✓ | ✓ |
| HBase | | ✓ | ✓ |
| Mahout | | ✓ | ✓ |
| Zookeeper | | ✓ | ✓ |
| Flume | | | ✓ |
| Sqoop | | | ✓ |
| Oozie | | | ✓ |
| Hamster | | | ✓ |
| GraphLab | | | ✓ |
| Advanced Database Services: | HAWQ | ✓ | ✓ |
| | PXF | ✓ | ✓ |

2.2.2 Installation Instructions

You can find installation instructions for the above components in these documents:

- Pivotal Command Center: *Pivotal HD Enterprise Installation and Administrator Guide* (this document) and also in *Pivotal Command Center User Guide*.
- Hadoop MR2 (via CLI): *Pivotal HD Enterprise Installation and Administrator Guide* (this document)

- Hadoop MR2 manual installs: *Pivotal HD Enterprise 2.0 Stack and Tool Reference Guide*.

2.2.3 Best Practices for Selecting Hardware

Typically you should select your cluster node hardware based on the resource requirements of your analytics workload and overall need for data storage. It is hard to anticipate the workload that may run on the cluster and so designing for a specific type of workload may lead to under utilization of hardware resources. Pivotal recommends that you select the hardware for a balanced workload across different types of system resources but also have the ability to provision more specific resources such as CPU, I/O bandwidth & Memory, as workload evolves over the time and demands for it.

Hardware and capacity requirements for cluster nodes may vary depending upon what service roles running on them. Typically failure of cluster slave nodes is tolerated by PHD services but disruption to master node can cause service availability issues. So it is important to provide more reliable hardware for master nodes (such as NameNode, YARN Resource manager, HAWQ master) for higher cluster availability.

Overall when choosing the hardware for cluster nodes, select equipment that lowers power consumption.



Following are not minimum requirements, they are Pivotal best practices recommendations.

Any configuration higher than the minimum recommendations is always preferable.

Cluster Slaves

Cluster slaves nodes run Hadoop service slaves such as the Datanode, NodeManager, RegionServer, and SegmentServer.

- 2 CPUs (4 to 8 cores)--- You can also have single CPU with more (6 to 8) cores and ability to add additional CPU, if needed in future. An algorithm to measure this is as follows: total map+reduce tasks per node are ≈ 1.5 times number of cores per node. Note: You might consider decreasing the number of map/reduce task per node when using PHD with HAWQ and assigning more cores to HAWQ segment servers based on mix work load of HAWQ vs. MapReduce.
- 24 to 64GB RAM per node — Typically 1 GB for each Hadoop daemons such as DataNode, NodeManager, Zookeeper etc. 2 to 3GB for OS and other services; and 1.5 or 2GB for each map/reduce task. **Note:** memory per map/reduce tasks on slave nodes depends on application requirement.
- 4 to 10, 2TB or 3TB disks, 7.2K RPM, SATA drives (JBOD) -- More disks per node provides more I/O bandwidth. Although more disk capacity per node may put more memory requirement on the HDFS Namenode. The reason for this is the total HDFS storage capacity grows with more number of cluster nodes while average HDFS file size stays small.
- 2 x 2TB or 3TB disks, RAID 1 configured for System OS. It can also store Hadoop daemon logs.
- 1GbE or 10GbE network connectivity within RACK

Cluster Masters

Cluster master nodes run Hadoop service masters such as the NameNode, ResourceManager, and HAWQ Master

You must select more more reliable hardware for cluster master nodes.

- Memory (RAM) requirement would be higher depending on the size of the cluster, number of HDFS storage and files. Typical memory ranges would be 24GB to 64 GB.
- Local disk storage requirement is 1 to 2TB, SAS disks, with RAID5/6



Master nodes require less storage than cluster slave nodes.

Pivotal HD Admin node

Ensure that the Admin node is separate from cluster nodes especially if cluster size has more than 15 - 20 nodes. The minimum hardware requirements are as follows:

- 1 Quad core CPU,
- 4 to 8GB RAM,
- 2x2TB SATA disks,
- 1GbE network connectivity

2.2.4 Best Practices for Deploying Hadoop Services

When creating your test environment, you can deploy all the Hadoop services and roles on a single node. A test cluster usually comprises 3 to 5 nodes. However, when deploying a production cluster with more nodes, here are some guidelines for better performance, availability, and use:

- Hadoop services Master roles: For example, HDFS NameNode, YARN ResourceManager and History Server, HBase Master, HAWQ Master. These should reside on separate nodes. These services and roles require dedicated resources since they communicate directly with Hadoop client applications. Running Hadoop slave application tasks (map/reduce tasks) on the same node interferes with master resource requirements.
- Hadoop services slave roles: For example, HDFS DataNode, YARN NodeManager, HBase RegionServer, HAWQ SegmentServer. These should reside on the cluster slave nodes. This helps provide optimal data access as well as better hardware use.
- HBase requires Zookeeper: Zookeeper should have an odd number of zookeeper servers. This application does not need dedicated nodes and can reside on the master server with ~ 1GB RAM and dedicated disk with ~ 1 TB of space.
- Hadoop Clients: For example, Hive, Pig etc. These should be installed on the separate gateway nodes depending on multi-user application requirements.

At this point you should have a bunch of systems with defined roles (admin node, namenode, HAWQ master, etc) all ready for install/deploy of the PHD software distribution.

2.3 Command Line Installation Overview

The table below provides a brief overview of the installation steps. Each step is covered in more detail in the following sections of this document.

| Task | Subtasks |
|--|--|
| Prerequisites See Pivotal HD Prerequisites for more details | Check JDK version <pre>\$ /usr/java/default/bin/java -version</pre> Ensure you're running Oracle Java JDK Version 1.7. If not, download the appropriate version from Oracle. Java version 1.7 is required; 1.7u15 is recommended |
| | Check Yum accessibility Verify that all hosts have yum access to an EPEL yum repository. <pre>\$ sudo yum list <LIST_OF_PACKAGES></pre> See Pivotal HD Prerequisites for the list of packages. Note that this is not mandatory if the required rpms are accessible locally. |
| | Verify iptables is turned off (as root) <pre>\$ chkconfig iptables off \$ service iptables stop \$ service iptables status iptables: Firewall is not running.</pre> Note: All machines in the cluster must also allow ICMP between boxes, and the Admin server must respond to ping. This is used during the <code>icm_client scan hosts'</code> command to test that the nodes can reach the admin server. |
| | Disable SELinux (as root) <pre>\$ echo 0 > /selinux/enforce</pre> |
| Prepare the Admin Node See Preparing the Admin Node for more details | Install Pivotal Command Center (as root) 1. Copy tar file to your specified directory on the admin node, for example: <pre>\$ scp ./PCC-2.2.x-<BUILD>.<ARCH>.tar.gz host:/root/phd/</pre> 2. Login as root and untar to that directory: <pre>\$ cd /root/phd \$ tar --no-same-owner -zxvf PCC-2.2.x-<BUILD>.<ARCH>.tar.gz</pre> 3. Run the installation script from the directory where it is installed: <pre>\$./install</pre> |

| Task | Subtasks |
|--|--|
| | <p>4. As the rest of the installation is done as the <code>gpadmin</code> user, change to <code>gpadmin</code> user:</p> <pre>\$ su - gpadmin</pre> <p>5. Enable Secure Connections (see Enabling Secure Connections for details)</p> |
| | <p>Import the Pivotal HD, HAWQ, and PRTS packages to the Admin node (as <code>gpadmin</code>)</p> <ol style="list-style-type: none"> Copy the Pivotal HD, and ADS (HAWQ) tarballs from the initial download location to the <code>gpadmin</code> home directory Change the owner of the packages to <code>gpadmin</code> and untar the tarballs <p>Note: If you want to use GemFire XD, you also need to import and enable the PRTS package. Complete instructions are in the Configuring GemFire XD section.</p> |
| | <p>Enable Pivotal HD Service (as <code>gpadmin</code>)</p> <pre>\$ icm_client import -s <PATH_OF_EXTRACTED_PHD_PACKAGE></pre> |
| | <p>Enable HAWQ and PXF services (as <code>gpadmin</code>)</p> <pre>\$ icm_client import -s <PATH_OF_EXTRACTED_ADS_PACKAGE></pre> |
| <p>Edit the Cluster Configuration File See Cluster Configuration Files for more details</p> | <p>Fetch the default Cluster Configuration template (as <code>gpadmin</code>)</p> <pre>\$ icm_client fetch-template -o ~/ClusterConfigDir</pre> <p>Note: <code>ClusterConfigDir</code> is created automatically.</p> |
| | <p>Edit the default Cluster Configuration template (<code>clusterConfig.xml</code>) (as <code>gpadmin</code>)</p> <p>At a minimum, you must replace all instances of your selected services with valid hostnames for your deployment.</p> <p>Notes:</p> <p>If you want to use GemFire XD, you need to add that service to the <code>clusterConfig.xml</code> file. Complete instructions are available in the Configuring GemFire XD section.</p> <p>If you want to enable HA, you need to make some HA-specific changes to some configuration files.</p> <p>Complete instructions are available in the High Availability section.</p> |

| Task | Subtasks |
|---|--|
| | Configure other Pivotal HD and ADS Components (as gpadmin) Optional: Configure HAWQ and other stack components in their corresponding configuration files (for example: <code>hawq/gpinitssystem_config</code> file), as needed |
| Deploy the Cluster See Deploying the Cluster for more details | Deploy/Install a cluster (as gpadmin) <pre>\$ icm_client deploy -c ~/ClusterConfigDir</pre> <p>Note: This command creates the <code>gpadmin</code> user on the cluster nodes. Do NOT create this user manually. If <code>gpadmin</code> already exists on the cluster nodes, delete that user before running this command.</p> |
| Verify the Cluster See Verifying a Cluster for more details | Start the Cluster (as gpadmin) <pre>\$ icm_client start -l <CLUSTER_NAME></pre> |
| | Verify HDFS (as gpadmin) <pre>\$ ssh <NAME_NODE> \$ hdfs dfs -ls /</pre> |
| | Initialize HAWQ (as gpadmin) ssh to the HAWQ master, the run the following: <pre>\$ source /usr/local/hawq/greenplum_path.sh \$ /etc/init.d/hawq init</pre> <p>If you have a HAWQ standby master configured, initialize that:</p> <pre>\$ gpinitstandby -s <STANDBY HAWQ MASTER FQDN></pre> |

2.4 Pivotal HD Enterprise Prerequisites

1. Have working knowledge of the following:

- **Yum:** Enables you to install or update software from the command line. See <http://yum.baseurl.org/>.
- **RPM** (Redhat Package Manager). See information on RPM at Managing RPM-Based Systems with Kickstart and Yum. See <http://shop.oreilly.com/product/9780596513825.do?sortby=publicationDate>
- **NTP.** See information on NTP at: <http://www.ntp.org>
- **SSH** (Secure Shell Protocol). See information on SSH at http://www.linuxproblem.org/art_9.html


2. **DNS lookup.** Verify that the admin host is be able to reach every cluster node using its hostname and IP address. Verify that every cluster node is able to reach every other cluster node using its hostname and IP address:

```
$ ping -c myhost.mycompany.com // The return code should be 0
$ ping -c 3 192.168.1.2 // The return code should be 0
```

3. **iptables.** Verify that iptables is turned off:

As root:

```
$ chkconfig iptables off
$ service iptables stop
```

 All machines in the cluster must also allow ICMP between boxes, and the admin server must respond to ping. This is used during the `icm_client scan hosts'` command to test that the nodes can reach the admin server.

4. **SELinux.** Verify that SELinux is disabled:

As root:

```
$ sestatus
```

If SELinux is disabled, one of the following is returned:

```
SELinuxstatus: disabled
```

or

```
SELinux status: permissive
```

If SELinux status is *enabled*, you can temporarily disable it or make it permissive (this meets requirements for installation) by running the following command:

As root:

```
$ echo 0 > /selinux/enforce
```



This only temporarily disables SELinux; once the host is rebooted, SELinux will be re-enabled. We therefore recommend permanently disabling SELinux, described below, while running Pivotal HD/HAWQ (however this requires a reboot).


You can permanently disable SE Linux by editing the `/etc/selinux/config` file as follows:

Change the value for the SELINUX parameter to:

```
SELINUX=disabled
```

Reboot the system.

5. JAVA JDK. Ensure that you are running Oracle JAVA JDK version 1.7.

 Version 1.7 is required; version 1.7u15 is recommended.

As root:

```
$ /usr/java/default/bin/java -version
```

If you are not running the correct JDK, you can download a supported version from the Oracle site at <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

6. YUM. Verify that all hosts have are available locally or that you have yum access to an EPEL yum repository. See Package Accessibility, below.

2.4.1 Package Accessibility

Pivotal Command Center and Pivotal HD Enterprise expect some prerequisite packages to be pre-installed on each host, depending on the software that gets deployed on a particular host. In order to have a smoother installation it is recommended that each host would have yum access to an EPEL yum repository. If you have access to the Internet, then you can configure your hosts to have access to the external EPEL repositories. However, if your hosts do not have Internet access (or you are deploying onto a large cluster), then having a local yum EPEL repo is highly recommended. This will also give you some control on the package versions you want deployed on your cluster. See [Creating a YUM EPEL Repository](#) for instructions on how to setup a local yum repository or point your hosts to an EPEL repository.

The following packages need to be either already installed on the admin host or be on an accessible yum repository:

- httpd
- mod_ssl
- postgresql
- postgresql-devel
- postgresql-server
- postgresql-jdbc

- compat-readline5
- createrepo
- sigar
- sudo

Run the following command on the admin node to make sure that you are able to install the prerequisite packages during installation:

```
$ sudo yum list <LIST_OF_PACKAGES>
```

If any of them are not available or not already installed, then you may have not added the repository correctly to your admin host.

For the cluster hosts (where you plan to install the cluster), the prerequisite packages depend on the software you will eventually install there, but you may want to verify that the following two packages are installed or accessible by yum on all hosts:

- nc
- postgresql-devel

[Back to Installation Overview](#)

2.5 Preparing the Admin Node

1. Make sure the following tarballs are available on the Admin node:

- * Pivotal Command Center (PCC)
- * Pivotal HD tarball (Apached Hadoop related services)
- * Pivotal ADS tarball (HAWQ, PXF services)
- Oracle JDK 1.7 Package - (you can download this from the Oracle site at <http://www.oracle.com/technetwork/java/javase/downloads/index.html>)

Note: Version 1.7 is required; version 1.7u15 is recommended.

* You can download these packages from the EMC Download Center at <https://emc.subscribenet.com>.

2. Make sure that all the tarballs are extracted and readable by the `gpadmin` user.

2.5.1 Install Pivotal Command Center

1. Copy the PCC tar file to your specified directory on the admin node, for example:

```
$ scp ./PCC-2.2.x-<BUILD>.<ARCH>.tar.gz host:/root/phd/
```


2. Login as root and untar to that directory:

```
$ cd /root/phd
$ tar --no-same-owner -zxvf PCC-2.2.x-<BUILD>.<ARCH>.tar.gz
```

3. Run the installation script from the directory where it is installed:

```
$ cd PCC-2.2.x-<BUILD>
$ ./install
```

This installs the required packages and configures Pivotal Command Center and starts PCC services.

4. Enable Secure Connections:

Pivotal Command Center uses HTTPS to secure data transmission between the client browser and the server. By default, the PCC installation script generates a self-signed certificate.

Alternatively you can provide your own Certificate and Key by following these steps:

- a. Set the ownership of the certificate file and key file to `gpadmin`.
- b. Change the permission to owner read-only (mode 400)
- c. Edit the PCC configuration file `/usr/local/greenplum-cc/config/commander` as follows:

Change the path referenced in the variable `PCC_SSL_KEY_FILE` to point to your own key file.

Change the path referenced in the variable `PCC_SSL_CERT_FILE` to point to your own certificate file.



See [SSL Certificates](#) for details

- d. Restart PCC with the following command:

```
$ service commander restart
```

5. Verify that your PCC instance is running by executing the following command:

```
# service commander status
```

The PCC installation also includes a CLI (Command Line Interface tool: `icm_client`). You can now deploy and manage the cluster using the CLI.

From now on you can switch to the `gpadmin` user. You should no longer need to be root for anything else.

```
$ su - gpadmin
```

2.5.2 Install Pivotal HD and HAWQ

Once you have Pivotal Command Center installed (the Command Center installation includes a CLI tool, `icm_client`, you now use to deploy and configure PHD services), you can use the `import` utility to sync the RPMs from the specified source location into the Pivotal Command Center (PCC) local yum repository of the Admin Node. This allows the cluster nodes to access the RPMs during deployment.



Notes

- If you want to use GemFire XD, you also need to import and enable the PRTS package. Complete instructions are in the [Configuring GemFire XD](#) section.
- Run `import` each time you wish to sync/import a new version of the package.

1. Copy the Pivotal HD, and ADS tarballs from the initial download location to the `gpadmin` home directory
2. Change the owner of the packages to `gpadmin` and untar the tarballs. For example:

```
# If the file is a tar.gz or tgz, use
$ tar zxf PHD-2.0.x-<BUILD>.tar.gz

# If the file is a tar, use
$ tar xf PHD-2.0.x-<BUILD>.tar

# Similarly for the Pivotal ADS tar.gz or tgz file, use
$ tar zxf PADS-1.2.x-<BUILD>.tar.gz

# If the file is a tar, use
$ tar xf PADS-1.2.x-<BUILD>.tar
```

Import Pivotal HD Service


1. As `gpadmin`, extract the following tarball for Pivotal HD:

```
$ icm_client import -s <PATH_OF_EXTRACTED_PHD_PACKAGE>
```

Example:

```
$ icm_client import -s PHD-2.0.x-<BUILD>/
```

Import HAWQ and PXF Services

 This is required only if you wish to deploy HAWQ.

1. As `gpadmin`, extract the following tar ball for HAWQ and PXF:

```
$ icm_client import -s <PATH_OF_EXTRACTED_ADS_PACKAGE>
```

Example:

```
$ icm_client import -s PADS-1.2.x-<BUILD>/
```

For more information, see the log file located at: `/var/log/gphd/gphdmgr/gphdmgr-import.log`

Syntax:

```
icm_client import --help
Usage: icm_client [options]

Options:
  -h, --help                Show this help message and exit
  -s STACK, --stack=STACK   Directory location of the PHD/PADS/PRTS stack
  -r RPM, --rpm=RPM         Location of the rpm to be included in the PHDMgr local
                           yum repository
  -f FILE, --file=FILE      Location of the file like JDK-version.bin which will
                           be used by PHDMgr during cluster deployment
```

Note: The version of the stack to be imported is now taken from the `-s` option (directory location).

[Back to Installation Overview](#)

2.6 Cluster Configuration Files

We provide a default Cluster configuration file (`clusterConfig.xml`) that you need to edit for your own cluster, all the cluster nodes are configured based on this configuration file.

At a minimum you must replace all instances of your selected services with valid hostnames for your deployment.

Advanced users can further customize their cluster configuration by editing the stack component configuration files such as `hdfs > core-site.xml`.

Specifically, for HAWQ you may have to edit the HAWQ configuration file, see [Configuring HAWQ](#); and if you want to enable HA, you need to make some HA-specific changes to several configuration files. Complete instructions are available in the [High Availability](#) section.



Do not use file paths in the configuration XML files that take this format: `file:///path/to/file`

These file paths will not work due to a puppet hand-off issue and will error out.

2.6.1 Fetching Default Cluster Configuration Templates

The `fetch-template` command saves a default cluster configuration template into the specified directory, such as a directory on disk. You can manually modify the template and use it as input to subsequent commands.

1. As `gpadmin`, run the `fetch-template` command.

Example:

```
$ icm_client fetch-template -o ~/ClusterConfigDir
```

The above example uses the `fetch-template` command to place a template in a directory called `ClusterConfigDir` (automatically created). This directory contains files which describe the topology of the cluster and the configurations for the various services installed on the cluster.

Syntax:

```
icm_client fetch-template --help
Usage: /usr/bin/icm_client fetch-template options

Options:
-h, --help          Show this help message and exit
-o OUTDIR, --outdir=OUTDIR
                    Directory path to store the cluster configuration template files
-v version          Optional. PHD version. If not specified defaults to the latest PHD stack
                    available.
```

2.6.2 Editing Cluster Configuration

1. Locate and update the `clusterConfig.xml` file based on your cluster requirements. At a minimum, you must update the default names of the nodes in this file to match the names of the nodes in your own cluster.



Notes

- If you want to use GemFire XD, you need to add that service to the `clusterConfig.xml` file. Complete instructions are available in the [Configuring GemFire XD](#) section.

- If you want to enable HA, you need to make some HA-specific changes to the `clusterConfig.xml` file and edit some other configuration files. Complete instructions are available in the [High Availability](#) section.

2. Once you've made your changes, we recommend you check that your xml is well-formed using the `xmlwf` command, as follows:

```
xmlwf ~/ClusterConfigDir/clusterConfig.xml
```

About the Cluster Configuration File

This section provides more information about what is contained in the Cluster Configuration file and what you can edit based on your cluster.

The `clusterConfig.xml` contains a default Cluster Configuration template.

The following is an example of the configuration files directory structure:

```
clusterConfig.xml
hdfs
  core-site.xml
  hadoop-env.sh
  hadoop-metrics2.properties
  hadoop-metrics2.properties
  hadoop-policy.xml
  hdfs-site.xml
  log4j.properties
yarn
  container-executor.cfg
  mapred-env.sh
  mapred-queues.xml
  mapred-site.xml
  postex_diagnosis_tests.xml
  yarn-env.sh
  yarn-site.xml
zookeeper
  log4j.properties
  zoo.cfg
hbase
  hadoop-metrics.properties
  hbase-env.sh
  hbase-policy.xml
  hbase-site.xml
  log4j.properties
hawq
  gpinitssystem_config
pig
  log4j.properties
  pig.properties
hive
  hive-env.sh
  hive-exec-log4j.properties
```

```
hive-log4j.properties
hive-site.xml
```

Note: There may not be a folder that corresponds to every service, for example Pig and Mahout do not have their own directories, they can be configured directly using the `client` tag in the `clusterConfig.xml` file.

The `clusterConfig.xml` file contains the following sections:

Head Section

This is the metadata section and must contain the following mandatory information:

- `clusterName`: Configure the name of the cluster
- `gphdStackVer`: Pivotal HD Version.
- `services`: Configure the services to be deploy. By default every service that Pivotal HD Enterprise supports is listed here. ZooKeeper, HDFS, and YARN are mandatory services. HBase and HAWQ are optional.
- `client`: The host that can be used as a gateway or launcher node for running the Hadoop, Hive, Pig, Mahout jobs.

Topology Section

<hostRoleMapping>

This is the section where you specify the roles to be installed on the hosts. For example, you can specify where your hadoop namenode, data node etc. should be installed. Please note that all mandatory roles should have at least one host allocated. You can identify the mandatory role by looking at the comment above that role in the `clusterConfig.xml` file.



Important

Always use fully-qualified domain names (FQDN) rather than short hostnames in the `clusterConfig.xml` file.

Global Service Properties

<servicesConfigGlobals>

This section defines mandatory global parameters such as Mount Points, Directories, Ports, `JAVA_HOME`. These configured mount points such as `datanode.disk.mount.points`, `namenode.disk.mount.points`, and `secondary.namenode.disk.mount.points` are used to derive paths for other properties in the `datanode`, `namenode` and `secondarynamenode` respectively. These properties can be found in the service configuration files.



Important

- `hawq.segment.directory` and `hawq.master.directory` need to be configured only if HAWQ is used.
- The values in this section are pre-filled with defaults. Check these values, they may not need to be changed.
- The mount points mentioned in this section are automatically created by Pivotal HD during cluster deployment.
- We recommend you have multiple disk mount points for datanodes, but it is not a requirement.

Other Configuration Files



Please ensure that the directories specified for `dfs.datanode.name.dir` and `dfs.datanode.data.dir` in the `hdfs/hdfs-site.xml` are empty.

Configuring Your Hadoop Service

Each service has a corresponding directory that containing standard configuration files. You can override properties to suit your cluster requirements, or consult with Pivotal HD support to decide on a configuration to suite your specific cluster needs.



You must not override properties derived from the global service properties, especially those derved from role information.

Example In `hdfs/core-site.xml`: `fs.defaultFS` which is set to

```
hdfs://$<NAMENODE>:$<dfs.port>
```

2.7 Configuring HAWQ

HAWQ system configuration is defined in `hawq/gpinitssystem_config`.

You can override the HAWQ database default database port setting, 5432, using the `MASTER_PORT` parameter. You can also change the HAWQ DFS path using the `DFS_URL` parameter.



Important

If you are planning to deploy a HAWQ cluster on VMs with memory lower than the optimized/recommended requirements, do the following:

Remove the entry `vm.overcommit_memory = 2` from `/usr/lib/gphd/gphdmgr/hawq_sys_config/sysctl.conf` prior to running the `prepare hawq` utility.

In the `clusterConfig.xml`, update `<hawq.segment.directory>` to include only one segment directory entry (instead of the default 4 segments).

[Back to Installation Overview](#)

2.8 Verifying the Cluster Nodes for Pivotal HD

1. As `gpadmin`, run the `scanhosts` command to verify certain prerequisites are met.

Example:

```
$ icm_client scanhosts -f ./HostFile.txt
```

The `scanhosts` command verifies that prerequisites for the cluster node and provides a detailed report of any missing prerequisites. Running this command ensures that clusters are deployed smoothly.

Syntax:

```
icm_client scanhosts --help
Usage: /usr/bin/icm_client scanhosts [options]

Options:
  -h, --help                Show this help message and exit
  -v, --verbose              Increase output verbosity
  -f HOSTFILE, --hostfile=HOSTFILE
                           File containing new-line separated list of hosts to be
                           scanned
  -j JAVAHOME, --java_home=JAVAHOME
                           java_home path configured
```

You can troubleshoot using the following files:

On the Admin Node:

- `/var/log/gphd/gphdmgr/ScanCluster.log`



We recommend running `scanhosts` before every deployment or reconfiguration of the cluster.

2.9 Deploying the Cluster

Pivotal HD deploys clusters using input from the cluster configuration directory. This cluster configuration directory contains files that describes the topology and configuration for the cluster and the installation procedure.

Deploy the cluster as `gpadmin`.

The deploy command internally does three steps:

1. Prepares the cluster nodes with the pre-requisites (runs `preparehosts` command)
2. Verifies the prerequisites (runs `scanhosts` command)
3. Deploys the cluster



Ensure that the JDK file downloaded from Oracle has execute permission.

The `preparehosts` command run internally as part of `deploy` performs the following on the hosts listed in the `clusterConfig.xml` file:

- Creates the `gpadmin` user.
- As `gpadmin`, sets up password-less SSH access from the Admin node.
- Installs the provided Oracle Java JDK.
- Disables SELinux across the cluster.
- Synchronizes the system clocks.
- Installs Puppet version 2.7.20 (the one shipped with the PCC tarball, not the one from puppetlabs repo)
- Installs `sshpas`.

If Oracle JDK is not already installed on the cluster nodes, you can install it via the `deploy` command. Here are the steps to deploy JDK.

1. Accept the license agreement and download Oracle JDK from the Oracle website
2. Import the JDK file into the PHDMgr files repo. PHDMgr looks for the files in its repository during deployment:



Java JDK version 1.7 is required; version 1.7u15 is recommended.

Use the following command to import the `.rpm` file:

```
icm_client import -r <JDK rpm>
```

3. Once the file is imported, you can use the name of the file with the `-j` option of `preparehosts` during deployment.



Pivotal recommends that you run only one deploy command at a time, because running simultaneous deployments might result in failure.

Example:

```
$ icm_client deploy -c clusterConfigDir/ -i -d -j jdk-7u15-linux-x86_64.rpm
```



Use the `-t` option only if you have NTP setup on the nodes.

You can check the following log files to troubleshoot any failures:

On Admin

- `/var/log/gphd/gphdmgr/GPHDClusterInstaller_XXX.log`
- `/var/log/gphd/gphdmgr/gphdmgr-webservices.log`
- `/var/log/messages`
- `/var/log/gphd/gphdmgr/installer.log`

On Cluster Nodes

- `/tmp/GPHDNodeInstaller_XXX.log`

Syntax:

```
icm_client deploy --help
Usage: /usr/bin/icm_client deploy [options]

Options:
  -h, --help                show this help message and exit
  -c CONFDIR, --confdir=CONFDIR
                           Directory path where cluster configuration is stored
  -s, --noscanhosts         Do not verify cluster nodes as part of deploying the
                           cluster
  -p, --nopreparehosts      Do not prepare hosts as part of deploying the cluster
  -j JDKPATH, --java=JDKPATH
                           Location of Sun Java JDK RPM (Ex: jdk-
                           7u15-linux-x64.rpm). Ignored if -p is specified
  -t, --ntp                 Synchronize system clocks using NTP. Optionally takes
                           NTP server as argument. Defaults to pool.ntp.org
                           (requires external network access). Ignored if -p is
                           specified
  -d, --selinuxoff          Disable SELinux. Ignored if -p is specified
  -i, --iptablesoff        Disable iptables. Ignored if -p is specified
  -y SYSCONFIGDIR, --sysconf=SYSCONFIGDIR
                           [Only if HAWQ is part of the deploy] Directory
                           location of the custom conf files (sysctl.conf and
                           limits.conf) which will be appended to
```

```
/etc/sysctl.conf and /etc/limits.conf on slave nodes.  
Default: /usr/lib/gphd/gphdmgr/hawq_sys_config/.  
Ignored if -p is specified
```

The deploy command has the option to skip running `preparehosts` or `scanhosts` as part of the deployment. If you have manually run `preparehosts` and `scanhosts` commands to confirm that all pre-requisites are met then you can skip running them during deployment using the `-s` and `-p` options.

You can also specify options to the `preparehosts` using `-j`, `-t`, `-d`, or `-i` options.

Your Pivotal HD installation is now complete.

You can now start a cluster and start HAWQ. Both these steps are described in "Verifying a Cluster Installation", next.

[Back to Installation Overview](#)

2.10 Verifying a Cluster Installation

We recommend that you verify your cluster installation.

To verify your cluster installation:

1. As `gpadmin`, start your cluster.

Example:

```
$ icm_client start -l <CLUSTERNAME>
```

See [Managing a Cluster](#) for more detailed instructions and other start up options.

2. Verify HDFS is running (you will not be able to initialize HAWQ if HDFS is not running). On the Name node run:

```
$ hdfs dfs -ls/
```

Sample Output:

```
Found 4 items  
drwxr-xr-x - mapred hadoop      0 2013-06-15 15:49 /mapred  
drwxrwxrwx - hdfs   hadoop      0 2013-06-15 15:49 /tmp  
drwxrwxrwx - hdfs   hadoop      0 2013-06-15 15:50 /user  
drwxr-xr-x - hdfs   hadoop      0 2013-06-15 15:50 /yarn
```

3. As `gpadmin`, initialize HAWQ from the HAWQ master.

Note that HAWQ is implicitly started as part of the initialization.

ssh to the HAWQ Master before you initialize HAWQ

Example:

```
$ su - gpadmin
$ source /usr/local/hawq/greenplum_path.sh
$ /etc/init.d/hawq init
```

If you have a HAWQ Standby master in your cluster configuration, initialize that by running the following:

```
$ gpinitstandby -s <HAWQ STANDBY MASTER FQDN>
```

See [Managing HAWQ](#) sections for more detailed instructions.

2.10.1 Verifying Service Status

You can use the `service status` command to check the running status of a particular service role from its appropriate host(s).

Refer to [Running Sample Programs](#) where you can see the sample commands for each Pivotal HD service role.

The following example shows an aggregate status view of hadoop, zookeeper and hbase service roles from all the cluster nodes:

```
[gpadmin]\# massh ./HostFile.txt verbose 'sudo service --status-all | egrep "hadoop | zookeeper  
| hbase"
```

Below is an example to check the status of all datanodes in the cluster:

```
# Create a newline separated file named 'datanodes.txt' containing all the datanode belonging to  
the service role \\  
[gpadmin]\# massh datanodes.txt verbose 'sudo service hadoop-hdfs-datanode status'
```

2.11 Preparing the Cluster Nodes (Deprecated)

This is an optional command that can be used to prepare the hosts before a cluster deployment. This command is internally run as part of deploy so it not necessary that you run this before a cluster deployment. This command has been retained for backward compatibility.

1. Create a hostfile (`HostFile.txt`) that contains the hostnames of all your cluster nodes except the Admin node; separated by newlines. You will have to input this file in many Pivotal HD commands.

For example, the hostfile should look like the following:

```
[gpadmin] cat HostFile.txt
host1.pivotal.com
host2.pivotal.com
host3.pivotal.com
```

Note: The following script shows how to create a hostfile as input for a large number of hosts:

```
[gpadmin] for i in `seq -w 1 3`; do sudo sh -c "echo \"host$i.pivotal.com\" >> HostFile.txt";
done
```



The hostfile should contain all nodes within your cluster EXCEPT the Admin node.

2.11.1 Preparing the Cluster Nodes for Pivotal HD

1. As `gpadmin`, run the `preparehosts` command to perform some administrative tasks to prepare the cluster for Pivotal HD.

Note: One of the tasks this command performs is to create the `gpadmin` user on the cluster nodes. Do **NOT** create this user manually. If `gpadmin` user already exists on the cluster nodes, delete that user by running:

```
$ pkill -KILL -u gpadmin
$ userdel -r gpadmin
```

Run `preparehosts`:

Example:

```
$ icm_client preparehosts --hostfile=./HostFile.txt --java=<PATH TO THE DOWNLOADED SUN JAVA
JDK> --ntp --selinuxoff --iptablesoff
```

The `preparehosts` command performs the following on the hosts listed in the hostfile:

- Creates the `gpadmin` user.
- As `gpadmin`, sets up password-less SSH access from the Admin node.
- Installs the provided Oracle Java JDK Version 1.7.
- Disables SELinux across the cluster.
- Synchronizes the system clocks.

- Installs Puppet version 2.7.20 (the one shipped with the PCC tarball, not the one from puppetlabs repo)
- Installs sshpass.



Do not use the Admin node as part of your cluster. The `preparehosts` command will automatically remove the admin host if included. It will not prepare the admin node as it might corrupt the admin nodes' certification process that is part of the puppet orchestration during deploy.

Syntax:

```
icm_client preparehosts --help
Usage: /usr/bin/icm_client preparehosts [options]
Options:
  -h, --help            show this help message and exit
  -f HOSTFILE, --hostfile=HOSTFILE
                        file containing a list of all cluster hosts (newline
                        separated)
  -j JAVA, --java=JAVA  location of Sun Java JDK RPM (Ex: jdk-
                        7u15-linux-x64.rpm)
  -t, --ntp              Synchronize system clocks using NTP. Optionally takes
                        NTP server as argument. Defaults to pool.ntp.org
                        (requires external network access).
  -d, --selinuxoff       disable SELinux
  -i, --iptablesoff     disable iptables
  -v, --verbose          increase output verbosity
```

2.12 Pivotal HD Directory Layout

The * indicates a designated folder for each Pivotal HD component.

| Directory Location | Description |
|--------------------|---|
| /usr/lib/gphd/* | The default <code>\$GPHD_HOME</code> folder. This is the default parent folder for Pivotal HD components. |
| /etc/gphd/* | The default <code>\$GPHD_CONF</code> folder. This is the folder for Pivotal HD component configuration files. |
| /etc/default/ | The directory used by service scripts to set up the component environment variables. |
| /etc/init.d | The location where a components' Linux Service scripts are stored. |
| /var/log/gphd/* | |

| Directory Location | Description |
|--------------------|---|
| | The default location of the \$GPHD_LOG directory. The directory for Pivotal HD component logs. |
| /var/run/gphd/* | The location of the any daemon process information for the components. |
| /usr/bin | The folder for the component's command scripts; only sym-links or wrapper scripts are created here. |

2.13 Running Sample Programs

Make sure you are logged in as user `gpadmin` on the appropriate host before testing the service.

2.13.1 Testing Hadoop

You can run Hadoop commands can be executed from any configured hadoop nodes.

You can run Map reduce jobs from the datanodes, resource manager, or historyserver.

```
# clear input directory, if any |
$ hadoop fs -rmr /tmp/test_input

# create input directory
$ hadoop fs -mkdir /tmp/test_input

# ensure output directory does not exist
$ hadoop fs -rmr /tmp/test_output

# copy some file having text data to run word count on
$ hadoop fs -copyFromLocal /usr/lib/gphd/hadoop/CHANGES.txt /tmp/test_input

# run word count
$ hadoop jar /usr/lib/gphd/hadoop-mapreduce/hadoop-mapreduce-examples-<version>.jar wordcount
/tmp/test_input /tmp/test_output

# dump output on console
$ hadoop fs -cat /tmp/test_output/part*
```



When you run a map reduce job as a custom user, *not* as `gpadmin`, `hdfs`, `mapred`, or `hbase`, note the following:

- Make sure the appropriate user staging directory exists.
- Set permissions on `yarn.nodemanager.remote-app-log-dir` to `777`. For example if it is set to the default value `/yarn/apps`, do the following:

```
$ sudo -u hdfs hadoop fs -chmod 777 /yarn/apps
```

- Ignore the Exception trace, this is a known Apache Hadoop issue.

2.13.2 Testing HBase

You can test HBase from the HBase master node

```
gpadmin# ./bin/hbase shell
hbase(main):003:0> create 'test', 'cf'
0 row(s) in 1.2200 seconds
hbase(main):003:0> list 'test'
..
1 row(s) in 0.0550 seconds
hbase(main):004:0> put 'test', 'row1', 'cf:a', 'value1'
0 row(s) in 0.0560 seconds
hbase(main):005:0> put 'test', 'row2', 'cf:b', 'value2'
0 row(s) in 0.0370 seconds
hbase(main):006:0> put 'test', 'row3', 'cf:c', 'value3'
0 row(s) in 0.0450 seconds

hbase(main):007:0> scan 'test'
ROW COLUMN+CELL
row1 column=cf:a, timestamp=1288380727188, value=value1
row2 column=cf:b, timestamp=1288380738440, value=value2
row3 column=cf:c, timestamp=1288380747365, value=value3
3 row(s) in 0.0590 seconds

hbase(main):012:0> disable 'test'
0 row(s) in 1.0930 seconds
hbase(main):013:0> drop 'test'
0 row(s) in 0.0770 seconds
```

2.13.3 Testing HAWQ



Use the HAWQ Master node to run HAWQ tests.

```
gpadmin# source /usr/local/hawq/greenplum_path.sh

gpadmin# psql -p 5432
psql (8.2.15)
Type "help" for help.

gpadmin=# \d
No relations found.
```



```

gadmin=# \l
List of databases
Name | Owner | Encoding | Access privileges
---+-----+-----
gadmin | gadmin | UTF8 |
postgres | gadmin | UTF8 |
template0 | gadmin | UTF8 |
template1 | gadmin | UTF8 |
(4 rows)

gadmin=# \c gadmin
You are now connected to database "gadmin" as user "gadmin".
gadmin=# create table test (a int, b text);
NOTICE: Table doesn't have 'DISTRIBUTED BY' clause -
Using column named 'a' as the Greenplum Database data
distribution key for this table.
HINT: The 'DISTRIBUTED BY' clause determines the distribution
of data. Make sure column(s) chosen are the optimal data
distribution key to minimize skew.

CREATE TABLE

gadmin=# insert into test values (1, '435252345');
INSERT 0 1
gadmin=# select * from test;
 a | b
--+-----
 1 | 435252345
(1 row)

gadmin=#

```

2.13.4 Testing Pig

You can test Pig from the client node

```

# Clean up input/output directories

hadoop fs -rmr /tmp/test_pig_input
hadoop fs -rmr /tmp/test_pig_output

#Create input directory

hadoop fs -mkdir /tmp/test_pig_input

# Copy data from /etc/passwd

hadoop fs -copyFromLocal /etc/passwd /tmp/test_pig_input

```

In the grunt shell, run this simple pig job

```
$ pig // Enter grunt shell
A = LOAD '/tmp/test_pig_input' using PigStorage(':');
B = FILTER A by $2 > 0;
C = GROUP B ALL;
D = FOREACH C GENERATE group, COUNT(B);
STORE D into '/tmp/test_pig_output';

# Displaying output

hadoop fs -cat /tmp/test_pig_output/part*

Cleaning up input and output'

hadoop fs -rmr /tmp/test_pig_*
```

2.13.5 Testing Hive

Test Hive from the client node:

```
gpadmin# hive

# Creating passwords table
hive> create table passwords (col0 string, col1 string, col2 string, col3 string, col4 string,
col5 string, col6 string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ":";
hive> SHOW TABLES;
hive> DESCRIBE passwords;

# Loading data
hive> load data local inpath "/etc/passwd" into table passwords;

# Running a Hive query involving grouping and counts
hive> select col3,count(*) from passwords where col2 > 0 group by col3;

# Cleaning up passwords table
hive> DROP TABLE passwords;
hive> quit;
```

2.14 Creating a YUM EPEL Repository

Pivotal Command Center and Pivotal HD Enterprise expect some prerequisite packages to be pre-installed on each host, depending on the software that gets deployed on a particular host. In order to have a smoother installation it is recommended that each host would have yum access to an EPEL yum repository. If you have access to the Internet, then you can configure your hosts to have access to the external EPEL repositories. However, if your hosts do not have Internet access (or you are deploying onto a large cluster) or behind a firewall, then having a local yum EPEL repository is highly recommended. This also gives you some control on the package versions you want deployed on your cluster.

Following are the steps to create a local yum repository from a RHEL or CentOS DVD:

1. Mount the RHEL/CentOS DVD on a machine that will act as the local yum repository
2. Install a webserver on that machine (e.g. httpd), making sure that HTTP traffic can reach this machine
3. Install the following packages on the machine:

```
yum-utils  
createrepo
```

4. Go to the directory where the DVD is mounted and run the following command:

```
$ createrepo ./
```

5. Create a repo file on each host with a descriptive filename in the /etc/yum.repos.d/ directory of each host (for example, CentOS-6.1.repo) with the following contents:

```
[CentOS-6.1]  
name=CentOS 6.1 local repo for OS RPMS  
baseurl=http://172.254.51.221/centos/$releasever/os/$basearch/  
enabled=1  
gpgcheck=1  
gpgkey=http://172.254.51.221/centos/$releasever/os/$basearch/RPM-GPG-KEY-CentOS-6
```

6. Validate that you can access the local yum repos by running the following command:

```
Yum list
```

You can repeat the above steps for other software. If your local repos don't have any particular rpm, download it from a trusted source on the internet, copy it to your local repo directory and rerun the `createrepo` step.

2.15 High Availability (HA)

- High availability is disabled by default.
- Currently we only support Quorum Journal based storage for high availability.

To enable HA for a new cluster; follow the instructions below.

To enable HA for an existing cluster, see *Enabling High Availability on a Cluster* in [Administering PHD Using the CLI](#) for details.

2.15.1 Setting up a New Cluster with HA

1. Follow the instructions for [Preparing the Admin Node](#) and for fetching and editing the [Cluster Configuration Files](#) earlier in this document.

To enable HA, you then need to make HA-specific edits to the following configuration files:

- `clusterConfig.xml`
- `hdfs/hdfs-site.xml`
- `hdfs/core-site.xml`
- `hbase/hbase-site.xml`
- `yarn/yarn-site.xml`



When specifying the `nameservices` in the `clusterConfig.xml`, do not use underscores ('_'), for example, `phd_cluster`.

2. Edit `clusterConfig.xml` as follows:

Comment out `secondarynamenode` role in `hdfs` service

Uncomment `standbynamenode` and `journalnode` roles in `hdfs` service

Uncomment `nameservices`, `namenodelid`, `namenode2id`, `journalpath`, and `journalport` entries in `serviceConfigGlobals`

3. Edit `hdfs/hdfs-site.xml` as follows:

Uncomment the following properties:

```
<property>
  <name>dfs.nameservices</name>
  <value>${nameservices}</value>
</property>

<property>
  <name>dfs.ha.namenodes.${nameservices}</name>
  <value>${namenodelid},${namenode2id}</value>
</property>

<property>
  <name>dfs.namenode.rpc-address.${nameservices}.${namenodelid}</name>
  <value>${namenode}:8020</value>
</property>

<property>
  <name>dfs.namenode.rpc-address.${nameservices}.${namenode2id}</name>
  <value>${standbynamenode}:8020</value>
</property>

<property>
  <name>dfs.namenode.http-address.${nameservices}.${namenodelid}</name>
  <value>${namenode}:50070</value>
```

```

</property>

<property>
  <name>dfs.namenode.http-address.${nameservices}.${namenode2id}</name>
  <value>${standbynamenode}:50070</value>
</property>

<property>
  <name>dfs.namenode.shared.edits.dir</name>
  <value>qjournal://${journalnode}/${nameservices}</value>
</property>

<property>
  <name>dfs.client.failover.proxy.provider.${nameservices}</name>
  <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>

<property>
  <name>dfs.ha.fencing.methods</name>
  <value>sshfence</value>
</property>

<property>
  <name>dfs.ha.fencing.ssh.private-key-files</name>
  <value>/home/exampleuser/.ssh/id_rsa</value>
</property>

<property>
  <name>dfs.ha.fencing.methods</name>
  <value>shell(/bin/true)</value>
</property>

<property>
  <name>dfs.journalnode.edits.dir</name>
  <value>${journalpath}</value>
</property>

<property>
  <name>dfs.ha.automatic-failover.enabled</name>
  <value>true</value>
</property>

```

Comment the following properties

```

<property>
  <name>dfs.namenode.secondary.http-address</name>
  <value>${secondarynamenode}:50090</value>
  <description>
    The secondary namenode http server address and port.
  </description>
</property>

```

4. Edit yarn/yarn-site.xml

```
<property>
  <name>mapreduce.job.hdfs-servers</name>
  <value>hdfs://${nameservices}</value>
</property>
```

5. Edit `hdfs/core-site.xml` as follows:

Set the following property key value:

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://${nameservices}</value>
  <description>The name of the default file system. A URI whose
  scheme and authority determine the FileSystem implementation. The
  uri's scheme determines the config property (fs.SCHEME.impl) naming
  the FileSystem implementation class. The uri's authority is used to
  determine the host, port, etc. for a filesystem.</description>
</property>
```

Uncomment following property:

```
<property>
  <name>ha.zookeeper.quorum</name>
  <value>${zookeeper-server}:${zookeeper.client.port}</value>
</property>
```

6. Edit `hbase/hbase-site.xml` as follows:

Set the following property key value:

```
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://${nameservices}/apps/hbase/data</value>
  <description>The directory shared by region servers and into
  which HBase persists. The URL should be 'fully-qualified'
  to include the filesystem scheme. For example, to specify the
  HDFS directory '/hbase' where the HDFS instance's namenode is
  running at namenode.example.org on port 9000, set this value to:
  hdfs://namenode.example.org:9000/hbase. By default HBase writes
  into /tmp. Change this configuration else all data will be lost
  on machine restart.
  </description>
</property>
```

7. To enable HA for HAWQ, comment out the default `DFS_URL` property and uncomment `DFS_URL` in `hawq/gpinitssystem_config` as follows:

```
#DFS_URL=${namenode}:${dfs.port}/hawq_data
#### For HA uncomment the following line
DFS_URL=${nameservices}/hawq_data
```

8. Add the following properties to `hawq/hdfs-client.xml`:

```
<property>
  <name>dfs.nameservices</name>
  <value>${nameservices}</value>
</property>

<property>
  <name>dfs.ha.namenodes.${nameservices}</name>
  <value>${namenodelid},${namenode2id}</value>
</property>

<property>
  <name>dfs.namenode.rpc-address.${nameservices}.${namenodelid}</name>
  <value>${namenode}:8020</value>
</property>

<property>
  <name>dfs.namenode.rpc-address.${nameservices}.${namenode2id}</name>
  <value>${standbynamenode}:8020</value>
</property>
<property>
  <name>dfs.namenode.http-address.${nameservices}.${namenodelid}</name>
  <value>${namenode}:50070</value>
</property>

<property>
  <name>dfs.namenode.http-address.${nameservices}.${namenode2id}</name>
  <value>${standbynamenode}:50070</value>
</property>

<property>
  <name>dfs.client.failover.proxy.provider.${nameservices}</name>
  <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
```

9. Either:

If the cluster is not already deployed, continue configuring your cluster as described earlier in this document; then deploy (see [Deploying the Cluster](#)).

or

If the cluster has already been deployed, use the `reconfigure` command to update the configuration files on the cluster.

2.16 Configuring GemFire XD

Pivotal HD Enterprise 2.x provides support for GemFire XD 1.0. GemFire XD is optional and is distributed separately from other PHD components.

GemFire XD is installed via the CLI. CLI installation instructions and configuration steps are provided below. GemFire XD can be added during initial deployment, like any other service, or can be added during a reconfiguration of a cluster.

Further operational instructions for GemFire XD are provided in the *Pivotal GemFire XD User's Guide*.

2.16.1 Overview

GemFire XD is a memory-optimized, distributed data store that is designed for applications that have demanding scalability and availability requirements.

2.16.2 Service Roles/Ports

The following table shows GemFire service roles:

| Role Name | Description | Port |
|--------------|---|------|
| gfxd-locator | The GemFire XD locator process provides discovery services for all members in a GemFire XD distributed system. A locator also provides load balancing and failover for thin client connections. As a best practice, deploy a locator in its own process (LOCATOR=local_only) to support network partitioning detection. | 1527 |
| gfxd-server | A GemFire XD server hosts database schemas and provides network connectivity to other GemFire XD members and clients. You can deploy additional servers as necessary to increase the capacity for in-memory tables and/or provide redundancy for your data. | 1527 |

2.16.3 Best Practices

HAWQ and GFXD services are both memory intensive and it is best to configure these services to be deployed on different nodes.

2.16.4 Enabling PRTS Services

Follow the instructions below to add GemFire XD before you deploy or reconfigure a cluster.

If you wish to deploy Gemfire XD Beta, perform the following:

1. Download the PRTS tarball from the initial download location to the gpadmin home directory.
2. Change ownership of the packages to gpadmin and untar. For example:
If the file is a tar.gz or tgz:tar xzf PRTS-1.0.x-<BUILD>.tgz
If the file is a tar:tar xf PRTS-1.0.x-<BUILD>.tar
3. As gpadmin, enable the PRTS service:


```
$ icm_client import -s <PATH_OF_EXTRACTED_PRTS_PACKAGE>
```

```
$ icm_client import -s PRTS-1.0.x-<BUILD>/
```

4. Edit the Cluster Configuration file as follows:

During initial deployment: Retrieve the `clusterConfig.xml` file using the `icm_client fetch-template` command. See [Cluster Configuration Files](#) for more details.

Adding to an exiting cluster: Edit the `clusterConfig.xml` file (`icm_client fetch-configuration`) then reconfigure the cluster (`icm_client reconfigure`). See [Reconfiguring a Cluster](#).

- Open `clusterConfig.xml` and add `gfxd` to the services listed in the `<services></services>` tag.
- Define the `gfxd-server` and `gfxd-locator` roles in the `clusterConfig.xml` file for every cluster by adding the following to the `<hostrolemapping> </hostrolemapping>` tag:
`<gfxd> <gfxd-locator>host.yourdomain.com</gfxd-locator>`
`<gfxd-server>host.yourdomain.com</gfxd-server></gfxd>`

2.16.5 GemFire XD Notes

NOTE 1:

Gemfire XD binaries have been deployed at this point but each node is not configured as needed.

```
<gfxd>
<gfxd-locator>host1</gfxd-locator>
<gfxd-server>host2</gfxd-server>
</gfxd>
# but host1 does not act as server upon service gfxd start command at this point.
```

Refer to the [Pivotal GemFire XD User's Guide](#) to complete the configuration.

NOTE 2:

You cannot start GemFire XD (`gfxd`) using the `icm_client start` command. Refer to the [Pivotal GemFire XD User's Guide](#) for instructions about starting your `gfxd` services.

2.16.6 Managing GemFire XD

Refer to the [Pivotal GemFire XD User's Guide](#).

A *Quick Start Guide* that includes instructions for starting and stopping `gfxd` servers and locators is also available, here:

http://gemfirexd-05.run.pivotal.io/index.jsp?topic=/com.pivotal.gemfirexd.0.5/getting_started/15-minutes.html

2.17 Installing SSL certificates

The following table contains information related to SSL certificates:

| Port | 443 | 5443 |
|--------------------------------|------------------------------------|---------------------------------------|
| Used by | Apache Default SSL | Command Center UI |
| Default Certificate Path | /etc/pki/tls/certs/localhost.crt | /usr/local/greenplum-cc/ssl/FQDN.cert |
| Default Key Path | /etc/pki/tls/private/localhost.key | /usr/local/greenplum-cc/ssl/FQDN.key |
| Config File | /etc/httpd/conf.d/ssl.conf | /etc/httpd/conf.d/pcc-vhost.conf |
| Post Key Change Step | service httpd restart | service httpd restart |
| SSL Version | SSLv3 TLSv1.0 | SSLv3 TLSv1.0 |
| Compression | No | No |
| Minimal Encryption Strength | medium encryption (56-bit) | strong encryption (96-bit or more) |
| ICM Upgrade | No Impact | Check configuration file and key |
| Support CA Signed Certificates | Yes | Yes |

3 Upgrading PHD Using the CLI

This section describes how to upgrade Pivotal HD using Pivotal Command Center's command line interface (CLI).


3.1 Overview


Pivotal HD Enterprise only supports upgrading cluster services that were originally installed via the CLI. Services that were manually installed using tar or rpms directly on cluster hosts will not be available post-CLI upgrade and must be manually re-installed.

 Complete Upgrade Syntax is provided [here](#).

3.2 Quick Guide

The table below briefly describes the steps you need to take to upgrade a cluster; more details are provided in the following sections:

| Step | Details |
|-------------------------------|--|
| Prerequisites | <p>PADS file location: If you are upgrading PADS, make note of the path to the extracted old PADS tar ball.</p> <p>Backup Data: We recommend that you backup any critical data before running any upgrade.</p> <p>Backup Service Configuration File(s): Backup the configuration files of any services you will be manually re-installing, post CLI-upgrade.</p> <p>Fetch original Template: Fetch original template provided by PCC.</p> <p>JDK 1.7: Make sure you are running JDK 1.7. If you are not, download it from Oracle.</p> <p>High Availability-enabled and Secure Clusters:</p> <div> You cannot upgrade High Availability-enabled or Secure clusters. Revert you cluster to non-HA and / or non-secure before proceeding with an upgrade. See Administering PHD Using the CLI for details.</div> |

| Step | Details |
|---|--|
| | <p>GemfireXD: The PHD 2.0 upgrade does not support an upgrade of the Gemfire XD service. You will have to remove the GemFire XD service temporarily in order to proceed with your upgrade.</p> |
| Stop Services | <p>Stop HAWQ (if applicable):</p> <pre># sudo /etc/init.d/hawq stop</pre> <p>(See Managing HAWQ for details.)</p> <p>As gpadmin, stop all PHD services :</p> <pre># icm_client stop -l <CLUSTER NAME></pre> <p>(See Managing a Cluster for details.)</p> <p>As root, stop PCC:</p> <pre># service commander stop</pre> |
| Extract and upgrade PCC | <p>Untar the new PCC package, then run (as root):</p> <pre># ./install</pre> <p>Change the user to gpadmin for the rest of the upgrade</p> |
| Import and extract new stacks | <p>Import PHD and PADS</p> <p>For each package, run:</p> <pre># icm_client import -s < PATH TO EXTRACTED TAR BALL ></pre> |
| CLI Self-Upgrade | <pre># icm_client self-upgrade</pre> |
| Upgrade PADS (HAWQ) | <pre># icm_client upgrade -l <CLUSTERNAME> -s pads -o < PATH TO EXTRACTED OLD ADS TAR BALL > -n < PATH TO EXTRACTED NEW ADS TAR BALL ></pre> <div>  If you are using HAWQ; it must be upgraded first, before PHD. </div> |
| Upgrade PHD | <p>PHD 2.0 requires Oracle JDK 1.7. Get the JDK rpm (for example: jdk-7u15-linux-x64.rpm) and include it in the upgrade command as shown below, so that the preparehosts command can deploy it to the cluster nodes later.</p> |

| Step | Details |
|--|--|
| | <code>icm_client upgrade -l <CLUSTERNAME> -s phd -j ~/jdk-7u15-linux-x64.rpm</code> |
| Reinstall Manually Installed Services | Services that were manually installed on an existing cluster will not be available post-CLI upgrade and must be manually re-installed. |
| Post Upgrade Configuration | <ol style="list-style-type: none"> 1. Synchronize configuration files 2. Reconfigure the cluster |
| Upgrade HDFS | <ul style="list-style-type: none"> • Backup Name Node data • Run <code>HdfsUpgrader.py</code> with appropriate options (see <code>Upgrade HDFS</code> for details) |
| Finalize HBase Upgrade | <ul style="list-style-type: none"> • Compact HBase tables • Run HBase upgrade |

For more details instructions of the above steps, see below:

3.3 Prerequisites

- **PADS file location:** Make note of the path to the extracted old PADS tar ball; you will need this information to upgrade PADS.
- **Backup Data:** We recommend you backup any critical data before performing any backups
- **Backup Service Configuration Files:** Services that were manually installed on an existing cluster will not be available post-CLI upgrade and must be manually re-installed. Backup the configuration files for these services. See the *Pivotal HD Enterprise Stack Tool and Reference Guide* for the locations of these configuration files.
- **Fetch original template:** As user `gpadmin`, fetch original template by running:

```
icm_client fetch-template -o ~/origTemplate
```

- **Oracle JDK 1.7.** Ensure that you are running Oracle JAVA JDK version 1.7.0_xx (minimum 1.7.0.15)

If you are not running the correct JDK, you can download a supported version from the Oracle site here:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

```
# Make sure that you have JDK1.7
$ java -version
java version "1.7.0_15"
Java(TM) SE Runtime Environment (build 1.7.0_15-b03)
Java HotSpot(TM) 64-Bit Server VM (build 23.7-b01, mixed mode)
```

- **High Availability-enabled clusters and secure clusters.** You cannot upgrade High Availability-enabled or Secure clusters. Revert you cluster to non-HA and / or non-secure before proceeding with an upgrade. See [Administering PHD Using the CLI](#) for details.
- **Remove Gemfire XD**

The PHD 2.0 upgrade does not support an upgrade of the Gemfire XD service.

You will have to remove the GemFire XD service temporarily in order to proceed with your upgrade.

To remove Gemfire XD, run the following:

```
vim icmconf/clusterConfig.xml # Remove gfxd from <services>
icm_client reconfigure -c icmconf -l test
```

Note that you will see the following error if you attempt to upgrade a cluster with GemFire XD installed.

Gemfire Upgrade Error Example

```
-bash-4.1$ icm_client upgrade -l test -s phd
Please ensure you've backed up manually installed service configurations (not installed by
icm_client) if any. Do you wish to proceed with the PHD upgrade ? . (Press 'y' to continue, any
other key to quit): y
Please enter the root password for the cluster nodes:
PCC creates a gpadmin user on the newly added cluster nodes (if any). Please enter a non-empty
password to be used for the gpadmin user:
Starting upgrade
```

```
Return Code : 6000
Message : Upgrade Cluster Error
Details :
Cluster Hosts :
    Operation Code : UPGRADE_FAILURE
    Operation Error : GEMFIRE XD must not be present if upgrading
    Log File : /var/log/gphd/gphdmgr/gphdmgr-webservices.log
```

```
[=====
100%
Results:
centos64-5... [Success]
centos64-4... [Success]
centos64-3... [Success]
centos64-2... [Success]
Details at /var/log/gphd/gphdmgr/gphdmgr-webservices.log
[ERROR] Cluster upgrade failed
-
```

Once you've completed your PHD Upgrade, you will need to reinstall Gemfire XD as a new install. See [Installing PHD Using the CLI](#) for details.

3.4 Upgrade Instructions

Follow the instructions below to upgrade your PHD system.

1. Stop Services:

- a. Stop HAWQ on hawq master:

```
# sudo /etc/init.d/hawq stop
```
- b. As `gpadmin`, stop all PHD services:

```
# icm_client stop -l <CLUSTER NAME>
```
- c. As `root`, top PCC:

```
# service commander stop
```

2. Import, extract, and upgrade PCC:

- a. Copy the new PCC tar file to your installation directory on the admin node, for example:

```
# scp ./PCC-2.2.x. version.build.os .x86_64.tar.gz host:/root/phd/
```
- b. Login as `root` and untar to that directory:

```
# cd /root/phd  
# tar --no-same-owner -zxvf PCC-2.2.x. version.build.os .x86_64.tar.gz
```
- c. As `root`, run the PCC installation script from the directory where it is installed:

```
# ./install
```



There is no need to specify that this is an upgrade; the install utility (`./install`) detects whether it is a fresh install or an upgrade.



The rest of the upgrade procedure is performed by the `gpadmin` user. Switch to that user now.

3. CLI Self-Upgrade:

As `gpadmin`, run the following command for the CLI to self-upgrade *after* importing but *before* upgrading the stacks (PHD, PADS):

```
# icm_client self-upgrade
```

4. Import and extract new PADS (HAWQ) package:

Optional, for HAWQ users:

```
# icm_client import -s < PATH TO EXTRACTED PADS TAR BALL >
```



Important

You need to upgrade HAWQ before upgrading PHD.

5. Upgrade HAWQ:

Optional, for HAWQ users:



This section is only applicable if you installed Pivotal ADS (HAWQ) using PHD's CLI; if you installed Pivotal ADS manually, refer to the *HAWQ Installation and Upgrade Guide* for manual upgrade instructions.

- a. If you have a standby HAWQ master, remove it. On the HAWQ master, as `gpadmin`, run:

```
$ gpinitstandby -r
```

For details, refer to the *HAWQ Installation and Upgrade Guide* for details.

- b. As `gpadmin` run the following command to upgrade PADS (HAWQ):

```
$ icm_client upgrade -l <CLUSTERNAME> -s pads -o < PATH TO EXTRACTED  
OLD ADS TAR BALL > -n < PATH TO EXTRACTED NEW ADS TAR BALL >
```

- c. On the HAWQ master node run the following commands to migrate data as:

```
su - gpadmin
source /usr/lib/gphd/hawq/greenplum_path.sh
gpmigrator <old_HAWQHOME_path> <new_HAWQHOME_path> # Look into ls -laF /usr/local
and find the old and new homes.

# For example:
gpmigrator /usr/local/hawq-1.1.3.0/ /usr/local/hawq-1.2.0.0/ -d
/data/master/gpseg-1/
```



If you encounter errors migrating HAWQ data, refer to the *HAWQ Administrator Guide* for help.

- d. Optional: You can delete the old HAWQ rpm file by running:

```
$ yum erase <HAWQ_OLD_RPM_NAME>
```

6. Import and extract new PHD package:

```
$ icm_client import -s < PATH TO EXTRACTED PHD TAR BALL >
```

7. Upgrade PHD:

If your cluster is configured with HAWQ, make sure you complete upgrading Pivotal ADS (see previous step),

before proceeding with Pivotal HD upgrade.

With the release of PHD-2.0.0.0, the HDFS version has changed. It creates a wire incompatibility with previous versions of HDFS data.

In order to upgrade the existing clusters to use PHD-2.0.0.0 stack, following pre-requisites must be met:

- The cluster to be upgraded should be running the last release of PHD:

PCC and PHD versions


```
PCC version - PCC-2.1.1-73.x86_64.tar.gz
PHD version - PHD-1.1.1.0-82.tar.gz (* - Update the exact certified version of
PHD-1.1.1 with latest patch )
```

- If the cluster is in HA mode, it must be reconfigured not to use Namenode HA.
- If security is enabled, it should be disabled. The `icm_client reconfigure` command removes the security configuration.
- It is safe to back up Namenode and Hive metadata before upgrade.

Once you have met the prerequisites, you can upgrade PHD by running (as `gpadmin`) the following command:

```
$ icm_client upgrade -l <CLUSTERNAME> -s phd
```

8. Re-install Manually Installed Services:

Services that were manually installed on an existing cluster will not be available post-CLI upgrade and must be manually re-installed. Backup the configuration files for these services. See the *Pivotal HD Enterprise Stack Tool and Reference Guide* for the locations of these configuration files.

Perform the cluster upgrade (See [Upgrade PHD](#) and / or [Upgrade PADS](#)).

Manually re-install the service(s) as described in the *Pivotal HD Enterprise Stack and Tool Reference Guide*.

Restore the service's configuration files across the cluster

9. Post-Upgrade Configuration:

Following an upgrade or reconfiguration, you need to synchronize the configuration files, as follows:

- a. Fetch the new templates that come with the upgraded software by running `icm_client fetch-template`, for example:

```
icm_client fetch-template -o ~/newTemplate
```

- b. Retrieve the existing configuration from the database by running `icm_client fetch-configuration`, for example:

```
icm_client fetch-configuration -o ~/origConfiguration -l <CLUSTERNAME>
```

- c. Identify the changes between the configurations by running the `diff` command, for example:

```
diff -ruBw origTemplate/ origConfiguration/
```

Where: `origTemplate` - ICM provided template for earlier version without the user changes

`origConfiguration` - User changes on top of `origTemplate`

`newTemplate` - ICM provided template for newer version without the user changes

- d. Once you have identified the changes between the configurations, apply those changes to the `newTemplate` you retrieved using the `fetch-template` command.



If you are re-using the `clusterConfig.xml` from `origConfiguration`, please ensure the `clusterConfig.xml` is updated with any new `servicesConfigGlobals` by checking the file in `newTemplate`

- e. Upgrade or reconfigure service by specifying the cluster configuration directory as `~/newTemplate` with your updated contents:

```
icm_client reconfigure -c ~/newTemplate -l test
```

- f. If you were utilizing a standby HAWQ master, you should have removed it before the upgrade. It should now be reinitialized. On the HAWQ master, as `gpadmin`, run:

```
$ gpinitstandby -s <standby_hostname>
```

For details, refer to the *HAWQ Installation and Upgrade Guide* for details.

10. Upgrade HDFS:

- a. Backup Namenode metadata by running:

```
/usr/bin/python /usr/lib/gphd/gphdmgr/lib/client/HdfsUpgrader.py -l <CLUSTER NAME>
-o backupNNMetadata -s 2.0.5_alpha_gphd_2_1_1_0 -t 2.2.0_gphd_3_0_0_0

# Source prefix would be 2.0.5_alpha_gphd_2_1_0_0 instead of
2.0.5_alpha_gphd_2_1_1_0 if you are upgrading from (PHD-1.1.0.0)
```

- b. Run NameNode upgrade by running:

```
/usr/bin/python /usr/lib/gphd/gphdmgr/lib/client/HdfsUpgrader.py -l <CLUSTER NAME>
-o nnuupgrade -s 2.0.5_alpha_gphd_2_1_1_0 -t 2.2.0_gphd_3_0_0_0
# Source prefix would be 2.0.5_alpha_gphd_2_1_0_0 instead of
2.0.5_alpha_gphd_2_1_1_0 if you are upgrading from (PHD-1.1.0.0)
```

- c. Run Data Node upgrade by running:

```
/usr/bin/python /usr/lib/gphd/gphdmgr/lib/client/HdfsUpgrader.py -l <CLUSTER NAME>
-o dnuupgrade -s 2.0.5_alpha_gphd_2_1_1_0 -t 2.2.0_gphd_3_0_0_0
# Source prefix would be 2.0.5_alpha_gphd_2_1_0_0 instead of
2.0.5_alpha_gphd_2_1_1_0 if you are upgrading from (PHD-1.1.0.0)
```

- d. Finalize the HDFS upgrade:

Your cluster is ready to be started at this point. Before you continue you should run a few tests to make sure your data upgrade was successful, and then you can run `finalizeUpgrade`. Once you have confirmed your cluster is working as expected, run the following command to finalize upgrade process:

```
/usr/bin/python /usr/lib/gphd/gphdmgr/lib/client/HdfsUpgrader.py -l <CLUSTER NAME>
-o finalizeUpgrade -s 2.0.5_alpha_gphd_2_1_1_0 -t 2.2.0_gphd_3_0_0_0
```



HBase master will not start unless the HBase upgrade is finalized. Please ensure HDFS upgrade is finalized before finalizing HBase upgrade.

11. Finalize HBase Upgrade:

- a. Compact all tables on the existing HBase 0.94 cluster : For example: to compact table `t1`, login to the HBase shell, then run:
`major_compact 't1'`
- b. Make sure Zookeeper and HDFS are running but HBase is stopped, then run:

```
$ sudo -u hbase bin/hbase upgrade -execute
```

3.5 Upgrade Syntax

Run the upgrade utility if you wish to upgrade the underlying stacks (Pivotal HD or ADS) in an existing cluster.

```
[gpadmin]# icm_client upgrade --help
Usage: /usr/bin/icm_client upgrade [options]

Options:
  -h, --help                show this help message and exit
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           the name of the cluster on which the operation is
                           performed
  -x, --noscanhosts         Do not verify cluster nodes.
  -s STACK, --stackname=STACK
                           stack to upgrade (phd or pads)
  -v VERSION, --version=VERSION
                           PHD Stack version, default is PHD-2.0.0.0 Stack
  -o OLDDIR, --old=OLDDIR
                           (Required for only for pads/hawq upgrade) Old PADS
                           Directory
  -n NEWDIR, --new=NEWDIR
                           (Required for only for pads/hawq upgrade) New PADS
                           Directory
  -p, --nopreparehosts      Do not prepare hosts as part of deploying the cluster
  -j JDKPATH, --java=JDKPATH
                           Location of Sun Java JDK RPM (Ex: jdk-
                           7u15-linux-x64.rpm). Ignored if -p is specified
  -t, --ntp                 Synchronize system clocks using NTP. Optionally takes
                           NTP server as argument. Defaults to pool.ntp.org
                           (requires external network access). Ignored if -p is
                           specified
  -d, --selinuxoff          Disable SELinux. Ignored if -p is specified
  -i, --iptablesoff        Disable iptables. Ignored if -p is specified
  -y SYSCONFIGDIR, --sysconf=SYSCONFIGDIR
                           [Only if HAWQ is part of the deploy] Directory
                           location of the custom conf files (sysctl.conf and
                           limits.conf) which will be appended to
```

```
/etc/sysctl.conf and /etc/limits.conf on slave nodes.  
Default: /usr/lib/gphd/gphdmgr/hawq_sys_config/.  
Ignored if -p is specified
```

3.6 Changed Configuration Parameters and Files

The following information is provided solely as reference material; you do not need to make any changes to your configuration files beyond those you have already completed.

The following configuration parameters were changed in PHD 2.0 as described below:

3.6.1 core-site.xml

Removed Parameters

The following parameters have been removed from core-site.xml:

| Name | Default | Notes |
|------------------------------|---|---|
| kfs.stream-buffer-size | 4096 | KFS is no longer supported, see HADOOP-8886 |
| mapred.outdir.resolverClass | org.apache.hadoop.mapreduce.DefaultPathResolver | |
| kfs.client-write-packet-size | 65536 | KFS is no longer supported, see HADOOP-8886 |
| kfs.blocksize | 67108864 | KFS is no longer supported, see HADOOP-8886 |
| kfs.bytes-per-checksum | 512 | KFS is no longer supported, see HADOOP-8886 |
| kfs.replication | 3 | KFS is no longer supported, see HADOOP-8886 |

New Parameters

The following parameters have been added to core-site.xml:

| Name | Default |
|---|---------|
| fs.client.resolve.remote.symlinks | true |
| nfs3.server.port | 2049 |
| nfs3.mountd.port | 4242 |
| hadoop.security.group.mapping.ldap.directory.search.timeout | 10000 |
| ipc.client.fallback-to-simple-auth-allowed | false |

3.6.2 yarn-site.xml

Changed Defaults

The following parameters in yarn-site.xml have new default values:

| Name | Old Value | New Value |
|-------------------------------|-------------------|-------------------|
| yarn.nodemanager.aux-services | mapreduce.shuffle | mapreduce_shuffle |

New Names

The following parameters in yarn-site.xml have new names:

| Old Name | New Name | Default Value |
|---|--|---------------------------------|
| yarn.resourcemanager.fs.rm-state-store.uri | yarn.resourcemanager.fs.state-store.uri | \${hadoop.tmp.dir}/y |
| yarn.nodemanager.resource.cpu-cores | yarn.nodemanager.resource.cpu-vcores | 8, See YARN-782 |
| yarn.nodemanager.aux-services. mapreduce.shuffle.class | yarn.nodemanager.aux-services. mapreduce_shuffle.class | org.apache.hadoop. |
| yarn.nodemanager.heartbeat.interval-ms | yarn.resourcemanager.nodemangers. heartbeat-interval-ms | 1000 |
| yarn.resourcemanager.am.max-retries | yarn.resourcemanager.am.max-attempts | 1->2 |

Removed Parameters

The following parameters have been removed from yarn-site.xml:

| Name | Default Value | Note |
|---|---------------|---|
| net.topology.with.nodegroup | false | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.dynamic.resource.memory.minimum.mb | 0 | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.dynamic.resource.vcores.maximum | -1 | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.dynamic.resource.enable | true | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.dynamic.resource.memory.maximum.mb | -1 | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.dynamic.resource.vcores.minimum | 0 | Introduced by hve patch. Will be added when the patch is added again to hadoop 2.2.0 |
| yarn.nodemanager.vcores-pcores-ratio | 2 | See YARN-782 |

New Parameters

The following parameters have been added to yarn-site.xml:

| Name | Default Value |
|---|---------------|
| yarn.resourcemanager.connect.retry-interval.ms | 30000 |
| yarn.resourcemanager.connect.max-wait.ms | 900000 |
| yarn.client.nodemanager-client-async.thread-pool-max-size | 500 |
| yarn.resourcemanager.hostname | 0.0.0.0 |
| yarn.resourcemanager.scheduler.monitor.enable | false |
| yarn.http.policy | HTTP_ONLY |

| Name | Default Value |
|--|---|
| yarn.nodemanager.hostname | 0.0.0.0 |
| yarn.client.max-nodemangers-proxies | 500 |
| yarn.resourcemanager.webapp.https.address | 0.0.0.0:8090 |
| yarn.nodemanager.resourcemanager.connect.wait.secs | 900 |
| yarn.client.app-submission.poll-interval | 1000 |
| yarn.resourcemanager.scheduler.monitor.policies | org.apache.hadoop.yarn.server.resource.monitor.capacity.ProportionalCapacityPre |
| yarn.nodemanager.local-cache.max-files-per-directory | 8192 |
| yarn.nodemanager.resourcemanager.connect.retry_interval.secs | 30 |

3.6.3 hdfs-site.xml

Changed Defaults

The following parameters in hdfs-site.xml have new default values:

| Name | Old Default Value | New Default Value |
|------------------------------|-------------------|-------------------|
| dfs.namenode.checkpoint.txns | 40000 | 1000000 |
| dfs.blocksize | 67108864 | 134217728 |

New Parameters

The following parameters have been added to hdfs-site.xml

| Name | Default Value |
|--|---------------|
| dfs.namenode.retrycache.heap.percent | 0.03f |
| dfs.client.write.exclude.nodes.cache.expiry.interval.millis | 600000 |
| dfs.namenode.retrycache.expirytime.millis | 600000 |
| dfs.image.transfer.timeout | 600000 |
| dfs.namenode.enable.retrycache | true |
| dfs.datanode.available-space-volume-choosing-policy.balanced-space-preference-fraction | 0.75f |
| dfs.namenode.edits.noeditlogchannelflush | false |

| Name | Default Value |
|--|---------------|
| dfs.namenode.fs-limits.max-blocks-per-file | 1048576 |
| dfs.namenode.fs-limits.min-block-size | 1048576 |
| dfs.datanode.available-space-volume-choosing-policy.balanced-space-threshold | 10737418240 |

3.6.4 mapred-site.xml

Changed Defaults

The following parameters in mapred-default.xml have new default values:

| Name | Old Default Value | New Default Value |
|--|---|-------------------|
| mapreduce.shuffle.port | 8080 | 13562 |
| yarn.app.mapreduce.client-am.ipc.max-retries | 1 | 3 |
| mapreduce.application.classpath | \$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*,\$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/lib/* | No default value |

New Parameters

The following parameters have been added to mapred-site.xml:

| Name | Default Value |
|--|--|
| mapreduce.jobhistory.loadedjobs.cache.size | 5 |
| mapreduce.am.max-attempts | 2 |
| mapreduce.jobhistory.done-dir | \${yarn.app.mapreduce.am.staging-dir}/history/done |
| mapreduce.jobhistory.cleaner.enable | true |
| mapreduce.jobhistory.datestring.cache.size | 200000 |
| mapreduce.jobhistory.max-age-ms | 604800000 |
| mapreduce.job.token.tracking.ids.enabled | false |
| mapreduce.jobhistory.joblist.cache.size | 20000 |
| mapreduce.jobhistory.move.thread-count | 3 |

| Name | Default Value |
|--|---|
| mapreduce.jobhistory.cleaner.interval-ms | 86400000 |
| mapreduce.jobhistory.client.thread-count | 10 |
| mapreduce.jobhistory.move.interval-ms | 180000 |
| mapreduce.jobhistory.minicluster.fixed.ports | false |
| mapreduce.jobhistory.http.policy | HTTP_ONLY |
| mapreduce.jobhistory.intermediate-done-dir | \${yarn.app.mapreduce.am.staging-dir}/history/done_intermediate |

3.6.5 httpfs-site.xml

New Parameters

The following parameters have been added to httpfs-site.xml:

| Name | Default Value |
|-----------------------------------|------------------------------------|
| httpfs.user.provider.user.pattern | ^[A-Za-z_][A-Za-z0-9._-]*[\$]?\$\$ |

3.6.6 capacity-scheduler.xml

Changed Defaults

The following parameters in capacity-scheduler.xml have new default values:

| Name | Old Default Value | New Default Value |
|---|--|---|
| yarn.scheduler.capacity.resource-calculator | org.apache.hadoop.yarn.server.resourcemanager.resource.DefaultResourceCalculator | org.apache.hadoop.yarn.util.DefaultResourceCalculator |

3.6.7 hbase-site.xml

Changed Defaults

The following parameters in hbase-site.xml have new default values:

| Name | Old Default Value | New Default Value |
|---|--|---|
| hbase.client.pause | 1000 | 100 |
| hbase.client.retries.number | 10 | 35 |
| hbase.client.scanner.caching | 1 | 100 |
| hbase.hregion.majorcompaction | 86400000 | 604800000 |
| hbase.hstore.blockingStoreFiles | 7 | 10 |
| hbase.regionserver.checksum.verify | false | true |
| hbase.regionserver.global.memstore.lowerLimit | 0.35 | 0.38 |
| hbase.regionserver.handler.count | 10 | 30 |
| hbase.regionserver.hlog.reader.impl | org.apache.hadoop.hbase.regionserver. wal.SequenceFileLogReader | org.apache.hadoop.hbase. wal.ProtobufLog |
| hbase.regionserver.hlog.writer.impl | org.apache.hadoop.hbase.regionserver. wal.SequenceFileLogWriter | org.apache.hadoop.hbase. wal.ProtobufLog |
| hbase.rootdir | file:///tmp/hbase-\${user.name}/hbase | \${hbase.tmp.dir} |
| hfile.block.cache.size | 0.25 | 0.4 |
| zookeeper.session.timeout | 180000 | 90000 |

New Names

The following parameters in hbase-site.xml have new names:

| Old Name | New Name | Default Value |
|---------------------------------|--------------------------------|--|
| hbase.rpc.engine | hbase.rpc.server.engine | org.apache.hadoop.hbase.ipc.WritableRpcEngine org.apache.hadoop.hbase.ipc.ProtobufRpcEngine |
| io.storefile.bloom.cacheonwrite | hfile.block.bloom.cacheonwrite | false (See HBASE-5957) |

Removed Parameters

The following parameters have been removed from hbase-site.xml:

| Name | Default Value | Description |
|---|---------------|---|
| hbase.table.archive.directory | .archive | Removed due to HBASE-8195 |
| hbase.regionserver. separate.hlog.for.meta | false | |

| Name | Default Value | Description |
|---|--|-------------------------------------|
| dfs.support.append | true | HDFS now support append by default. |
| hbase.mapreduce. hfileoutputformat.blocksize | 65536 | |
| hbase.regionserver.nbreservationblocks | 4 | |
| hbase.regionserver.lease.period | 60000 | |
| hbase.hash.type | murmur | |
| hbase.regionserver.class | org.apache.hadoop.hbase. ipc.HRegionInterface | |

New Parameters

The following parameters have been added to hbase-site.xml:

| Name | Default Value |
|---|---|
| hbase.client.scanner.timeout.period | 60000 |
| hbase.storescanner.parallel.seek.enable | false |
| hbase.thrift.htablepool.size.max | 1000 |
| hbase.hstore.bytes.per.checksum | 16384 |
| hbase.config.read.zookeeper.config | false |
| hbase.master.loadbalancer.class | org.apache.hadoop.hbase.master. balancer.StochasticLoadBalancer |
| hbase.rpc.shortoperation.timeout | 10000 |
| hbase.snapshot.enabled | true |
| hbase.hstore.checksum.algorithm | CRC32 |
| hbase.status.publisher.class | org.apache.hadoop.hbase.master. ClusterStatusPublisher\$MulticastPublisher |
| hbase.status.listener.class | org.apache.hadoop.hbase.client. ClusterStatusListener\$MulticastListener |
| hbase.security.authentication | simple |
| hbase.master.catalog.timeout | 600000 |
| hbase.hstore.compaction.kv.max | 10 |

| Name | Default Value |
|--|--|
| fail.fast.expired.active.master | false |
| hbase.metrics.exposeOperationTimes | true |
| hbase.client.localityCheck.threadPoolSize | 2 |
| hbase.status.published | false |
| hbase.status.multicast.address.ip | 226.1.1.3 |
| hbase.dynamic.jars.dir | \${hbase.rootdir}/lib |
| hbase.hregion.majorcompaction.jitter | 0.50 |
| hbase.status.multicast.address.port | 6100 |
| hbase.lease.recovery.dfs.timeout | 64000 |
| hbase.server.compactchecker.interval.multiplier | 1000 |
| hbase.rpc.timeout | 60000 |
| hbase.lease.recovery.timeout | 900000 |
| hbase.storescanner.parallel.seek.threads | 10 |
| hbase.regionserver.catalog.timeout | 600000 |
| hbase.ipc.client.tcpcnodelay | true |
| hbase.rest.filter.classes | org.apache.hadoop.hbase.rest.filter.GzipFilter |
| hbase.ipc.client.fallback-to-simple-auth-allowed | false |
| hbase.table.lock.enable | true |

3.6.8 hive-site.xml

The following parameters have been added to hive-site.xml:

| Name | Default Value |
|---------------------------|--|
| hive.default.rcfile.serde | org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe |

4 Administering PHD Using the CLI

This section describes the administrative actions that can be performed via Pivotal Command Center's command line interface (CLI).

4.1 Managing a Cluster

4.1.1 Starting a Cluster

You can use the `start` command to start all the configured services of the cluster, to start individual services configured for the cluster, and to start individual roles on a specific set of hosts.

```
icm_client start --help
Usage: /usr/bin/icm_client start [options]

Options:
  -h, --help                show this help message and exit
  -v, --verbose              increase output verbosity
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           the name of the cluster on which the operation is
                           performed
  -s SERVICES, --service=SERVICES
                           service to be started
  -f, --force                forcibly start cluster (even if install is incomplete)
  -r ROLES, --role=ROLES
                           The name of the role which needs to be started
  -o HOSTFILE, --hostfile=HOSTFILE
                           The absolute path for the file containing host names
                           for the role which needs to be started
```

The following table describes the list of values for the HDFS, MapRed, ZooKeeper, HBase, and HAWQ services:

| Option | Description |
|--------------------|---|
| <code>start</code> | Starts all configured cluster services in the right topological order based on service dependencies. |
| <code>-s</code> | Starts the specified service and all services it depends on in the right topological order. The supported services are hdfs, yarn, zookeeper, hbase, hive, hawq, pig, and mahout. |
| <code>-r</code> | Starts only the specified role on a specific set of hosts. Hosts can be specified using the <code>-o</code> option. |
| <code>-f</code> | Forces the cluster to start even if the installation is incomplete. |

The first time the cluster is started, Pivotal HD implicitly initializes the cluster. For subsequent invocations of the `start` command, the cluster is not initialized.

Cluster initialization includes the following:

- Namenode format
- Create directories on the local filesystem of cluster nodes and on the hdfs with the correct permission overrides. See the [Overriding Directory Permissions](#) section.
- Create HDFS directories for additional services, such as HBase, if these are included in the configured services.



Notes

Refer to the "Verifying the Cluster Nodes for Pivotal HD" section to make sure the cluster services are up and running.

Make sure you back up all the data prior to installing or starting a new cluster on nodes that have pre-existing data on the configured mount points.

For example:

Cluster level start:

```
[gpadmin]# icm_client start -l CLUSTERNAME
```

Service level start:

```
[gpadmin]# icm_client start -l CLUSTERNAME -s hdfs
```

Role level start:

```
[gpadmin]# icm_client start -l CLUSTERNAME -r datanode -o hostfile
```

4.1.2 Stopping a Cluster

You can use the `stop` command to stop an entire cluster, to stop a single service, and to stop a single role on a specific set of hosts on which it is configured.

```
[gpadmin]# icm_client stop -h
Usage: icm_client stop [options]

Options:
  -h, --help                Show this help message and exit
  -v, --verbose              Increase output verbosity
  -l CLUSTERNAME, --clustername=CLUSTERNAME
```

```

        The name of the cluster on which the operation is
        performed
-s SERVICES, --service=SERVICES
        Service to be stopped
-r ROLES, --role=ROLES
        The name of the role which needs to be stopped
-o HOSTFILE, --hostfile=HOSTFILE
        The absolute path for the file containing host names
        for the role that needs to be stopped

```

The following table describes the list of values for the HDFS, MapRed, ZooKeeper, HBase, and HAWQ services.

| Option | Description |
|--------|--|
| stop | Stops all configured cluster services in the right topological order based on service dependencies. |
| -s | Stops the specified service and all the dependent services in the right topological order. The supported services are hdfs, yarn, zookeeper, hbase, hive, hawq, pig, and mahout. |
| -r | Stops the specified role on a specific set of hosts. Hosts can be specified using the -o option. |

For example:

Cluster level stop:

```
[gpadmin]# icm_client stop -l CLUSTERNAME
```

Service level stop:

```
[gpadmin]# icm_client stop -l CLUSTERNAME -s hdfs
```

Role level stop:

```
[gpadmin]# icm_client stop -l CLUSTERNAME -r datanode -o hostfile
```

4.1.3 Restarting a Cluster

You can use the `-restart` command to stop, then restart a cluster.

See stopping and starting a cluster, above, for more details about the stop/start operations.

```

[gpadmin]# icm_client restart -h
Usage: /usr/bin/icm_client restart [options]

Options:
  -h, --help          Show this help message and exit

```

```
-v, --verbose          Increase output verbosity
-l CLUSTERNAME, --clustername=CLUSTERNAME
                        The name of the cluster on which the operation is
                        performed
-s SERVICES, --service=SERVICES
                        The service to be restarted
-f, --force            Forcibly start cluster (even if install is incomplete)
-r ROLES, --role=ROLES
                        The name of the role which needs to be started
-o HOSTFILE, --hostfile=HOSTFILE
                        The absolute path for the file containing host names
                        for the role which needs to be started
```

4.1.4 Reconfiguring a Cluster

Run the `reconfigure` command to update specific configuration for an existing cluster.



Caution

Running the `reconfigure` command on a secure cluster will disable security.

Some cluster specific configurations cannot be updated:



Important

- Reconfiguring the topology of a cluster (host to role mapping) is not allowed. For example: changing the `NameNode` to a different node or adding new set of datanodes to a cluster
- Properties based on hostnames: For example, `fs.defaultFS`, `dfs.namenode`. and the `http-address`.
- Properties with directory paths as values.

The following table lists properties that can only be changed with a `--force` option.



- You are expected to take care of all the necessary prerequisites prior to making changes to any of the following properties using the force flag
- Incorrect provisioning can make the cluster get into an inconsistent/unusable state

| Property Name | Configuration File |
|--------------------------------------|--------------------|
| datanode.disk.mount.points | clusterConfig.xml |
| namenode.disk.mount.points | clusterConfig.xml |
| secondary.namenode.disk.mount.points | clusterConfig.xml |
| hawq.master.directory | clusterConfig.xml |
| hawq.segment.directory | clusterConfig.xml |
| zookeeper.data.dir | clusterConfig.xml |

```
icm_client reconfigure -h
Usage: /usr/bin/icm_client reconfigure [options]

Options:
  -h, --help                show this help message and exit
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           the name of the cluster on which the operation is
                           performed
  -c CONFDIR, --confdir=CONFDIR
                           Directory path where cluster configuration is stored
  -s, --noscanhosts         Do not verify cluster nodes.
  -p, --nopreparehosts      Do not preparehosts as part of deploying the cluster.
  -j JDKPATH, --java=JDKPATH
                           Location of Sun Java JDK RPM (Ex: jdk-
                           7u15-linux-x64.rpm). Ignored if -p is specified
  -t, --ntp                 Synchronize system clocks using NTP. Optionally takes
                           NTP server as argument. Defaults to pool.ntp.org
                           (requires external network access). Ignored if -p is
                           specified
  -d, --selinuxoff          Disable SELinux. Ignored if -p is specified
  -i, --iptablesoff        Disable iptables. Ignored if -p is specified
  -y SYSCONFIGDIR, --sysconf=SYSCONFIGDIR
                           [Only if HAWQ is part of the deploy] Directory
                           location of the custom conf files (sysctl.conf and
                           limits.conf) which will be appended to
                           /etc/sysctl.conf and /etc/limits.conf on slave nodes.
                           Default: /usr/lib/gphd/gphdmgr/hawq_sys_config/.
                           Ignored if -p is specified
  -f, --force               Forcibly reconfigure the cluster (allows changes to
                           any servicesConfigGlobals property)
```

To reconfigure an existing cluster:

1. Stop the cluster:
`icm_client stop -l CLUSTERNAME`
2. Fetch the configurations for the cluster in a local directory:
`icm_client fetch-configuration -l CLUSTERNAME -o LOCALDIR`
3. Edit the configuration files in the cluster configuration directory (LOCALDIR).

4. Reconfigure the cluster:

```
icm_client reconfigure -l CLUSTERNAME -c LOCALDIR
```

Following an upgrade or reconfiguration, you need to synchronize the configuration files, as follows:

1. Fetch the new templates that come with the upgraded software by running `icm_client fetch-template`.
2. Retrieve the existing configuration from database using `icm_client fetch-configuration`.
3. Synchronize the new configurations (`hdfs/hadoop-env`) from the template directory to the existing cluster configuration directory.
4. Upgrade or reconfigure service by specifying the cluster configuration directory with updated contents.

4.1.5 Add / Remove Services

Services can be added / removed using `icm_client reconfigure` command.

- Edit the `clusterConfig.xml` file to add or remove services from the service list in `services` tag
- Edit `hostRoleMapping` section to add or remove hosts for the specific services configured
- Edit the `servicesConfigGlobals` if required for the specific service added
- Follow the steps for [Reconfiguring a Cluster](#).
- As in a new deployment you can use the `-p` or `-s` option to disable scanhosts or preparehosts on the newly added hosts
- If you want to prepare the new hosts with java or if you want to disable iptables or SELinux, follow the instructions for installing Java mentioned in the Deploying a cluster section of this document



Removing a specific service using the `icm_client reconfigure` command does not remove rpms from the nodes. The rpms are only removed when the Cluster is uninstalled

4.1.6 Add Hosts to Cluster

If you plan to add hosts as part of adding a new service, perform the following:

- Prepare the new hosts using the `icm_client preparehosts` command
- Refer to *Add / Remove Services* section

If you plan to add/remove hosts as part of an existing service in the cluster do the following:



You can only add or remove hosts for slave roles (refer to *Expanding a Cluster* section for the list of slave roles). You cannot make host changes for any other role.

- Prepare the new hosts using the `icm_client preparehosts` command

- You can add the new hosts to the corresponding slave roles in the `hostRoleMapping` section in `clusterConfig.xml`
- Follow the steps for [Reconfiguring a Cluster](#)



You cannot add one service and remove another at the same time. You have to perform these as two separate steps; however you can add multiple services OR remove multiple services at the same time.

4.1.7 Retrieving Configuration about a Deployed Cluster

Run the `fetch-configuration` command to fetch the configurations for an existing cluster and store them in a local file system directory.

```
icm_client fetch-configuration -h
Usage: icm_client fetch-configuration [options]

Options:
  -h, --help                show this help message and exit
  -o OUTDIR, --outdir=OUTDIR
                           Directory path to store the cluster configuration
                           template files
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           Name of the deployed cluster whose configurations need
                           to be fetched
```

Sample Usage

```
icm_client fetch-configuration -l CLUSTERNAME -o LOCALDIR
```

4.1.8 Listing Clusters

Run the `list` command to see a list of all the installed clusters:

```
[gpadmin]# icm_client list --help
Usage: icm_client list [options]

Options:
  -h, --help                show this help message and exit
  -v, --verbose              increase output verbosity
```

Sample Usage:

```
icm_client list
```

4.1.9 Expanding a Cluster



Make sure you run `preparehosts` against the new slave hosts prior to adding them to the cluster. (See the `preparehosts` command example in the "Preparing the Cluster for Pivotal HD" section.)

Run the `add-slaves` command to add additional slave hosts to an existing cluster. All the slave roles for **existing** cluster services will be installed on the new cluster hosts.

The following table indicates the services and their corresponding slave roles. Services not included in this list are not allowed for expansion (or shrinking).

| Service Name | Slave |
|--------------|--------------------|
| hdfs | datanode |
| yarn | yarn-nodemanager |
| hbase | hbase-regionserver |
| hawq | hawq-segment |

If you only want to install an individual component on a node, you should do this by manually editing the `clusterConfig.xml` file, then running the `reconfigure` command (see [Reconfiguring a Cluster](#)).

```
icm_client add-slaves --help
Usage: /usr/bin/icm_client add-slaves [options]

Options:
  -h, --help                show this help message and exit
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           the name of the cluster on which the operation is
                           performed
  -f HOSTFILE, --hostfile=HOSTFILE
                           file containing new-line separated list of hosts that
                           are going to be added.
  -s, --noscanhosts         Do not verify cluster nodes.
  -j JAVAHOME, --java_home=JAVAHOME
                           JAVA_HOME path to verify on cluster nodes
  -p, --nopreparehosts      Do not preparehosts as part of deploying the cluster.
  -k JDKPATH, --java=JDKPATH
                           Location of Sun Java JDK RPM (Ex: jdk-
                           7u15-linux-x64.rpm). Ignored if -p is specified
  -t, --ntp                 Synchronize system clocks using NTP. Optionally takes
                           NTP server as argument. Defaults to pool.ntp.org
                           (requires external network access). Ignored if -p is
                           specified
  -d, --selinuxoff          Disable SELinux for the newly added nodes. Ignored if
```

```

        -p is specified
-i, --iptablesoff    Disable iptables for the newly added nodes. Ignored if
                    -p is specified
-y SYSCONFIGDIR, --sysconf=SYSCONFIGDIR
                    [Only if HAWQ is part of the deploy] Directory
                    location of the custom conf files (sysctl.conf and
                    limits.conf) which will be appended to
                    /etc/sysctl.conf and /etc/limits.conf of the newly
                    added slave nodes. Default:
                    /usr/lib/gphd/gphdmgr/hawq_sys_config/. Ignored if -p
                    is specified

```

Sample Usage:

```
icm_client add-slaves -l CLUSTERNAME -f slave_hostfile
```

Make sure you start datanode and yarn nodemanager on the newly added slave hosts.

```
icm_client start -l CLUSTERNAME -r datanode -o hostfile
icm_client start -l CLUSTERNAME -r yarn-nodemanager -o hostfile
```



Important

- If HBase is configured, start hbase-regionserver as well.
- Don't expect data blocks to be distributed to the newly added slave nodes immediately.



If HAWQ is configured, refer to the *Expanding HAWQ* section



Hive does not have any slave roles, and therefore cannot be provisioned for an expansion.

4.1.10 Shrinking a Cluster



Make sure you decommission the slave hosts (refer to the next section) prior to removing them to avoid potential data loss.

Run the `remove-slaves` command lets the user to remove slave hosts from an existing cluster. All the slave roles for the existing cluster services will be removed from the given hosts.

```
icm_client remove-slaves --help
Usage: /usr/bin/icm_client remove-slaves [options]

Options:
  -h, --help                show this help message and exit
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                           the name of the cluster on which the operation is
                           performed
  -f HOSTFILE, --hostfile=HOSTFILE
                           file containing new-line separated list of hosts that
                           are going to be removed.
```

Sample Usage:

```
icm_client remove-slaves -l CLUSTERNAME -f hostfile
```

4.1.11 Decommissioning Nodes

Decommissioning is required to prevent potential loss of data blocks when you shutdown/remove slave hosts from a cluster. This is not an instant process since it requires replication of potentially a large number of blocks to other cluster nodes.

The following are the manual steps to decommission slave hosts (datanodes,nodemangers) from a cluster.

- On the NameNode host machine
 - Edit the `/etc/gphd/hadoop/conf/dfs.exclude` file and add the datanode hostnames to be removed (separated by newline character). Make sure you use the FQDN for each hostname.
 - Execute the dfs refresh command

```
[gpadmin] sudo -u hdfs hdfs dfsadmin -refreshNodes
```

- On the Yarn Resource Manager host machine
 - Edit `/etc/gphd/hadoop/conf/yarn.exclude` file and add the node manager hostnames to be removed (separated by newline character). Make sure you use the FQDN for each hostname.
 - Execute the yarn refresh command

```
[gpadmin] sudo -u hdfs yarn rmadmin -refreshNodes
```

- Check Decommission status
 - Monitor decommission progress with name-node Web UI http://NAMENODE_FQDN:50070 and navigate to Decommissioning Nodes page

- Check whether the admin state has changed to Decommission In Progress for the DataNodes being decommissioned. When all the DataNodes report their state as Decommissioned then all the blocks have been replicated.
- Shut down the decommissioned nodes
 - Stop datanode and yarn node manager on the targeted slaves to be removed

```
[gpadmin] icm_client stop -l CLUSTERNAME -r datanode -o hostfile
[gpadmin] icm_client stop -l CLUSTERNAME -r yarn-nodemanager -o hostfile
```



For HBase regionservers you can proceed with shutting down the region servers on the slave hosts to be removed. It is preferable to use `graceful_stop` script that hbase provides if load balancer is disabled.

4.1.12 High Availability

Enabling High Availability on a Cluster

- High availability is disabled by default.
- Currently we only support Quorum Journal based storage for high availability.

To enable HA for a new cluster; follow the instructions provided in the *High Availability* section of [Installing PHD Using the CLI](#).

To enable HA for an existing cluster, see below.

1. Download and import the latest version of Pivotal Command Center (PCC) (see [Installing PHD Using the CLI](#) for details)

Note: PCC 2.1 is the first version to support HA.

2. Reconfigure your cluster.

- a. Stop the cluster:

```
icm_client stop -l CLUSTERNAME
```

- b. Fetch the configurations for the cluster in a local directory:

```
icm_client fetch-configuration -l CLUSTERNAME -o LOCALDIR
```

- c. Fetch the new template configuration:

```
icm_client fetch-template -o ~/ClusterConfigDir
```

- d. Merge the HA-related configuration changes into your existing cluster configuration.

See the [High Availability](#) section of [Installing PHD Using the CLI](#) for details of the HA-specific information you need to add to the configuration files.

- e. Reconfigure the cluster:

```
icm_client reconfigure -l CLUSTERNAME -c LOCALDIR
```



Caution

Running the `reconfigure` command on a secure cluster disables security.

3. Update the HIVE Metastore:

Hive metastore contains references to `hdfs` path with `namenode:port` in the url. This needs to be updated to use the nameservices so HIVE scripts can work when ever NameNode failure happens.

Note: Make sure metastore is not running and is backed up to a persistent store before running the update commands.

a. Login to host configured as `hive-metastore`.

b. Display the current NameNode and `hdfs`path for hive warehouse directory:

```
/usr/lib/gphd/hive/bin/metatool -listFSRoot
```

c. Run the following command:

```
/usr/lib/gphd/hive/bin/metatool -updateLocation hdfs://<nameservices>  
hdfs://<current_namenode>:<dfs_port>
```

Where `nameservices` is the logical name used for the nameservices in a HA enabled cluster and `current_namenode` is the hostname of the NameNode on the cluster before reconfiguring to enable HA.



When specifying the `nameservices`, do not use underscores ('_'), for example, `phd_cluster`.

4. Restart HAWQ services for your configuration changes to take effect. Since `icm_client start` and `stop` do not start/stop the HAWQ master, you will have to do this manually.

From the HAWQ master, run the following:

```
sudo /etc/init.d/hawq stop  
sudo /etc/init.d/hawq start
```

You can now start the entire cluster with all configured services running.

Disabling High Availability on a Cluster

- High availability is disabled by default.
- You must disable high availability before upgrading a cluster.



The following procedure is only valid for environments running Pivotal Command Center 2.2 or above

To disable high availability:

1. Run the following command to fetch a new blank, non-HA template:

```
icm_client fetch-template -o NonHATemplate
```

You can use the `-v` option as follows to specify a template for either PHD 1.1.0 or PHD 1.1.1:

```
icm_client fetch-template -o NonHATemplateDir -v PHD-1.1.0.0
```

```
icm_client fetch-template -o NonHATemplateDir -v PHD-1.1.1.0
```

2. Replace `clusterconfig.xml` in your existing template directory with the template you just fetched.
3. Made the following changes to `clusterconfig.xml` to disable HA:
 - a. Locate the `<hostRoleMapping>` `<hdfs>` tags and comment out the `<standbynamenode>` and `<journalnode>` tags.
 - b. Locate the `<hostRoleMapping>` `<hdfs>` tags and add `<secondarynamenode>` tag and provide a host.
 - c. Locate the `servicesConfigGlobals` section and comment out the following elements:

```
<!-- Choose a logical name for HA nameservice, for example "test". -->
<!-- The name you choose will be used both for configuration and as the -->
<!-- authority component of absolute HDFS paths in the cluster -->

<!--<nameservices>test</nameservices>-->

<!-- Choose ids for the two namenodes being used. -->
<!-- These ids will be used in the configuration -->
<!-- of properties related to these namenodes -->
<!--<namenodelid>nn1</namenodelid>-->
<!--<namenode2id>nn2</namenode2id>-->

<!--specify the path on quorum journal nodes where the shared data is
written -->
<!--<journalpath>/data/qjournal/namenode</journalpath>-->
<!--specify the port where quorum journal nodes should be run-->
<!--<journalport>8485</journalport>-->
```

4. Run the following command to reconfigure the cluster with your new configuration file: `icm_client reconfigure -l <CLUSTER NAME> -c NonHATemplateDir`

HAAdmin Command Reference

- `hdfs haadmin` prints help for all subcommands and options. `serviceid` is the logical name configured for each NameNode, as `namenodelid` and `namenode2id`, in `clusterConfig.xml`
- Check state of a given NameNode:

```
hdfs haadmin -getServiceState <serviceid> Ex : hdfs haadmin -getServiceState nn1
```

- Transition a given NameNode to standby:

```
hdfs haadmin -transitionToStandby <serviceid>
```

For example:

```
hdfs haadmin -transitionToStandby nn1
```

- Transition a given NameNode to active:

```
hdfs haadmin -transitionToActive <serviceid>
```

For example:

```
hdfs haadmin -transitionToActive nn1
```

- Failover between two NameNode:

```
hdfs haadmin --failover <serviceid> <serviceid>
```

For example:

```
hdfs haadmin --failover nn1 nn2
```

4.1.13 Security

PHD clusters can be configured to use Kerberos authentication.

Kerberos is a network authentication protocol that provides strong authentication for client/server applications using secret-key cryptography.

Enabling Kerberos Authentication

You must install and configure Kerberos to enable security in Pivotal HD 1.1.x. and higher. Complete instructions are provided in the *PHD Stack and Tools Reference Guide*.

Disabling Kerberos Authentication

Anytime you run the `reconfigure` command; security will automatically be disabled for that cluster and must be manually reconfigured.

4.1.14 Uninstalling a Cluster

You must run the `stop` command to stop running clusters before running the `uninstall` command. You must also ensure that HAWQ has been stopped before uninstall.

You will be prompted as to whether you want to preserve the history metrics of the cluster; the default behavior is to preserve the history.



Running the `uninstall` will not delete `dfs.data.dir`, `dfs.name.dir`, `dfs.mapred.dir` and `dfs.checkpoint.dir` directories. This is intentional behavior and preserves user data.

```
[gpadmin]# icm_client uninstall -h
Usage: icm_client uninstall [options]

Options:
  -h, --help            Show this help message and exit
  -v, --verbose          Increase output verbosity
  -l CLUSTERNAME, --clustername=CLUSTERNAME
                        The name of the cluster to be uninstalled
```

Sample Usage

```
icm_client uninstall -l CLUSTERNAME
```



If you had HAWQ installed as part of the cluster, you will have to manually reset the `limits.conf` and `sysctl.conf` files on the HAWQ nodes before you can reuse those nodes again.

4.2 Managing HAWQ

Starting and stopping HAWQ can only be initiated directly on the HAWQ Master. More information about HAWQ can be found in the *Pivotal HAWQ 1.x Installation Guide* and the *Pivotal ADS 1.x Administrator Guide*.

4.2.1 Initializing HAWQ

You must initialize HAWQ only once after the cluster has started and specifically after the HDFS is up and running:

```
[gpadmin]# source /usr/local/hawq/greenplum_path.sh
[gpadmin]# /etc/init.d/hawq init
```

Running the `init` command, completes the following:

- Initializes the HAWQ master and the segment hosts.
- Starts the HAWQ master, segments, and the underlying postgres database.
- Exchanges SSH keys between the Master and Segment nodes.



- This operation takes a few minutes to complete.
- If you need to initialize the HAWQ standby master refer to the *Pivotal HAWQ Installation Guide* for instructions

If you have a HAWQ Standby master in your cluster configuration, initialize that by running the following:

```
# gpinitstandby -s <HAWQ STANDBY MASTER FQDN>
```

4.2.2 Starting HAWQ

Run the `start` command to start up the HAWQ master and all the segments hosts including the postgres database.

Note that this is implicitly done as part of the HAWQ Initialization.

```
[gpadmin]# /etc/init.d/hawq start
```

4.2.3 Stopping HAWQ

Run the `stop` command to stop the hawq master, segments hosts, and the postgres database on the HAWQ master.

```
[gpadmin]# /etc/init.d/hawq stop
```

4.2.4 Modifying HAWQ User Configuration

If you are using Pivotal Command Center, you must modify your HAWQ user configuration file.

This is because the Admin host is not part of the HAWQ cluster. Modifying the `pg_hba.conf` file on the HAWQ Master host, gives the Admin host the ability to remote query HAWQ .

1. Logon to the HAWQ Master as user `gpadmin`.
2. In the `$MASTER_DATA_DIRECTORY/pg_hba.conf` (the location of the HAWQ Master Directory is defined in the `<hawq.master.directory>` section of the `clusterConfig.xml` file used for deployment of the Cluster.

Find the entry:

```
host all gpadmin <master_host_ip>/32 trust
```

Change the subnet entry depending on your network configuration:

```
host all gpadmin <master_host_ip>/24 trust
```

3. Restart HAWQ

```
/etc/init.d/hawq restart
```

Run the following command to test HAWQ from the Admin host:

```
$ sudo -u gpadmin psql -h <HAWQ MASTER NODE> -p <HAWQ PORT> -U gpadmin postgres -c "select *  
from pg_stat_activity;"
```

4.2.5 Expanding HAWQ

HAWQ Segments can be expanded.

Before you expand a HAWQ segment you need to add slaves to the cluster by either:

- Running the `add-slaves` command (see [Expanding a Cluster](#))
- Manually editing the `hawq-segments` section of the `clusterConfig.xml` file, then running the `reconfigure` command (see [Reconfiguring a Cluster](#))

Once you have added the slaves, you can then expand HAWQ using the `gpexpand` command; refer to the *HAWQ Administration Guide - Expanding the HAWQ System* for details.

4.3 Managing Roles and Hosts

Pivotal HD supports starting or stopping entire clusters or individual roles on a selected hosts. If you want to start and stop the roles manually follow these steps:

You have the following options when managing cluster and individual roles:

- Managing locally
- Managing from the Admin Node

4.3.1 Managing Locally

You can manage the service role on the target host locally. For example, to restart datanode:

```
node100:gpadmin# ssh gpadmin@node100  
gpadmin# sudo service hadoop-hdfs-namenode restart
```

4.3.2 Managing Remotely

You can manage the service role remotely across one of the target hosts. For example, to restart datanode:

```
node100.gpadmin# massh node100 verbose 'sudo service hadoop-hdfs-datanode restart'
```

To restart all the datanodes remotely:

Create a newline separated file named `hostfile` that contains all the datanodes to *start*, *stop*, *restart*, or *check* status.

```
gpadmin# massh hostfile verbose 'sudo service hadoop-hdfs-datanode restart'
```

Pivotal HD Services Scripts

The following table shows the service commands to *start*, *stop*, *restart*, or *check* status for each service role,.

| Role Name | Service Command |
|---------------------|---|
| Namenode | <pre>sudo service hadoop-hdfs-namenode {starts stop status restart}</pre> |
| Secondary NameNode | <pre>sudo service hadoop-hdfs-secondarynamenode {starts stop status restart}</pre> |
| Datanode | <pre>sudo service hadoop-hdfs-datanode {starts stop status restart}</pre> |
| Resource Manager | <pre>sudo service hadoop-yarn-resourcemanager {starts stop status restart}</pre> |
| Node Manager | <pre>sudo service hadoop-yarn-nodemanager {starts stop status restart}</pre> |
| History Server | <pre>sudo service hadoop-mapreduce-historyserver {starts stop status restart}</pre> |
| Zookeeper Server | <pre>sudo service zookeeper-server {starts stop status restart}</pre> |
| HBase Master | <pre>sudo service hbase-master {starts stop status restart}</pre> |
| HBase Region Server | <pre>sudo service hbase-regionserver {starts stop status restart}</pre> |

| Role Name | Service Command |
|---------------------|---|
| HAWQ Master | <pre>sudo /etc/init.d/hawq {starts stop status restart}</pre> |
| Quorum Journal node | <pre>sudo /etc/init.d/hadoop-hdfs-journalnode {start stop status restart}</pre> |

4.4 Pivotal HD Services Reference

4.4.1 Overriding Directory Permissions

The following table shows the list of directories that Pivotal HD overrides with specific ownership and permissions.

Directories not mentioned in the below list follow standard Apache ownership and permission convention.

On the Local Filesystem

| Service | Directory | Location | Owner | Permissions |
|-----------|-------------------------------------|-----------------------|----------------------------|-------------|
| HDFS | <i>hadoop.tmp.dir</i> | All hadoop nodes | <i>hdfs:hadoop</i> | 777 |
| | <i>dfs.namenode.name.dir</i> | Namenode | <i>hdfs:hadoop</i> | 700 |
| | <i>dfs.datanode.data.dir</i> | Datanodes | <i>hdfs:hadoop</i> | 770 |
| | <i>dfs.namenode.checkpointdir</i> | Secondary Namenode | <i>hdfs:hadoop</i> | 700 |
| | <i>dfs.journalnode.edits.dir</i> | Journal Node | <i>hdfs:hadoop</i> | 755 |
| YARN | <i>mapreduce.cluster.local.dir</i> | All yarn nodes | <i>mapred:hadoop</i> | 755 |
| | <i>mapreduce.cluster.temp.dir</i> | All yarn nodes | <i>mapred:hadoop</i> | 755 |
| | <i>yarn.nodemanager.local-dirs</i> | Node Managers | <i>yarn:yarn</i> | 755 |
| | <i>yarn.nodemanager.log-dirs</i> | Node Managers | <i>yarn:yarn</i> | 755 |
| ZooKeeper | <i>dataDir (/var/lib/zookeeper)</i> | Zookeeper Servers | <i>zookeeper:zookeeper</i> | 775 |
| | <i>dataDir/myid</i> | Zookeeper Servers | <i>gpadmin</i> | 644 |
| HAWQ | <i>MASTER_DIRECTORY</i> | HAWQ Master & Standby | <i>gpadmin:hadoop</i> | 755 |

| Service | Directory | Location | Owner | Permissions |
|---------|-----------------------|---------------|-----------------------|-------------|
| | <i>DATA_DIRECTORY</i> | HAWQ Segments | <i>gpadmin:hadoop</i> | 755 |

On HDFS

| Service | Directory | Owner | Permissions |
|--------------|--|----------------------|-------------|
| HDFS | <i>hadoop.tmp.dir</i> | <i>hdfs:hadoop</i> | 777 |
| | <i>/tmp</i> | <i>hdfs:hadoop</i> | 777 |
| | <i>mapreduce.jobtracker.system.dir</i> | <i>mapred:hadoop</i> | 700 |
| | <i>yarn.app.mapreduce.am.staging-dir (/user)</i> | <i>mapred:hadoop</i> | 777 |
| | <i>mapreduce.jobhistory.intermediate-done-dir (/user/history/done)</i> | <i>mapred:hadoop</i> | 777 |
| | <i>mapreduce.jobhistory.done-dir (/user/history/done)</i> | <i>mapred:hadoop</i> | 777 |
| | <i>yarn.nodemanager.remote-app-log-dir</i> | <i>mapred:hadoop</i> | 755 |
| HBase | <i>hbase directory (/apps/hbase/data)</i> | <i>hdfs:hadoop</i> | 775 |
| HAWQ | <i>hawq directory (/hawq_data)</i> | <i>hdfs:hadoop</i> | 755 |

4.4.2 Pivotal HD Users and Groups

| Service | Users | Group | Login |
|-----------|-----------|-----------|-------|
| PHD | gpadmin | gpadmin | Yes |
| HDFS | hdfs | hadoop | Yes |
| MapReduce | mapred | hadoop | Yes |
| Hbase | hbase | hadoop | No |
| Hive | hive | hadoop | No |
| Zookeeper | zookeeper | zookeeper | No |
| Yarn | yarn | yarn | No |
| PHD, HAWQ | postgres | postgres | Yes |
| Puppet | puppet | puppet | No |

4.4.3 Pivotal HD Ports

If you are running a firewall, ensure that the following ports are open

| Service | Port |
|---------------------------|----------------------|
| ssh | 22 |
| NameNode | 8020 (Apache 9000) |
| NameNode Web UI | 50070, 50470 (https) |
| Secondary NameNode | 50090 |
| DataNode Communication | 50010 |
| DataNode IPC | 50020 |
| DataNode HTTP Address | 50075 |
| ResourceManager Web UI | 8042,8088 |
| ResourceManager | 8030,8031,8032,8033 |
| MapReduce Shuffle Port | 7070 |
| Job History Server | 10020 |
| Job History Web UI | 19888 |
| JobTracker | (Apache 9001) |
| JobTracker Web UI | (Apache 50030) |
| TaskTracker | (Apache 50060) |
| Puppet | 443,8140,61613 |
| Jetty | 8080 |
| HBase Master | 60000 |
| HBase Master UI | 60010 |
| HBase RegionServer | 60020 |
| HBase RegionServer Web UI | 60030 |
| ZooKeeper Client | 2181 |
| ZooKeeper Leader | 3888 |
| ZooKeeper Peers | 2888 |
| HAWQ Master | 8432 |

| Service | Port |
|--------------------------|-------|
| HAWQ Port Base | 40000 |
| Quorum Journal node port | 8485 |

5 PHD FAQ (Frequently Asked Questions)

5.1 Can I deploy multiple clusters from the same admin?

Yes, you can deploy any number of Pivotal HD clusters from the same admin. You must deploy them in succession, not simultaneously.

5.2 Can I modify the topology (host to role mapping) of the cluster after the initial install?

Yes, you can change slaves roles using the CLI, but the master role must be changed manually. If you want to change the master role, contact Support.

5.3 How do I reformat the namenode?



These steps will erase all data on HDFS.

As user `hdfs`:

1. On the namenode, clean up the data in the directories specified for `dfs.datanode.name.dir`
2. On all the datanodes, clean up the data in the directories specified for `dfs.datanode.data.dir`
3. On the namenode, run: `hadoop namenode format -force`

5.4 Certain services such as `hadoop-hdfs-namenode` or `hadoop-hdfs-datanode` do not come up when I run "start cluster"?

Refer to Debugging tips in the Troubleshooting section. It may be that the ports being used by the specific service are already in use. Verify whether the port is already being used using `netstat -na`. Kill the existing process if necessary

5.5 What group and users are created by Pivotal HD?

Please refer to the Troubleshooting section for details about the users and directories created by PCC.

5.6 What is the allowed time difference amongst the cluster nodes versus the admin node?

The allowed time difference between the cluster nodes is +/-60 secs of admin node time. If the time difference is more, the SSL authentication might fail leading to cluster deployment failures.

5.7 Does PCC support simultaneous deployment of multiple clusters?

No. Concurrent deployment is not allowed. Please wait till the first deployment is complete before starting another.

5.8 Does PCC support hostname both in IP address and FQDN format?

No, only FQDN format is currently supported.

5.9 Can a node be shared between different clusters?

No, nodes cannot be shared between clusters.

5.10 I installed puppet-2.7.20 from the Puppet Labs repository but Pivotal HD does not work?

Pivotal HD requires the version of puppet shipped with the product and not the downloadable version from the Puppet Labs repository. Uninstall Puppet and install the one shipped with the product using the `icm_client preparehosts` command.

5.11 How do I clean up the nodes if a cluster deployment fails?

Uninstall the cluster using the `icm_client uninstall` command then follow instructions for deploying the cluster again.

5.12 Will I lose my data if I uninstall the cluster?

Uninstalling the cluster will not wipe out any data. But a subsequent installation would wipe out the configured mount points upon confirmation. Make sure you back out the data.

5.13 Will I lose my data if I upgrade the PHD/ADS stack through the stack import utility?

Upgrading any stack using the import utility will not affect your cluster/data as long as the upgrade is compatible with the existing data layout.

5.14 Can I upgrade Pivotal Command Center while the clusters are functioning?

Yes you can. Upgrading the Admin node will not interfere with any of the clusters.

5.15 How do I change the port used by Pivotal HD?

1. Log onto the machine as `root`.
2. Stop Pivotal Command Center:

```
service commander stop
```

3. Change the port in the jetty file, say from 8080 to 8085:

```
Update the JETTY_PORT property to 8085 in: /usr/lib/gphd/gphdmgr/bin/setenv.sh
Update ICM_URL property to 8085 in /etc/gphd/gphdmgr/conf/gphdmgr.properties
Update the gphdmgr_port to 8085 in /usr/local/greenplum-cc/config/app.yml
```

```
\#Replace 8080 with 8085 in the following files
sed -i 's/8080/8085/g' /usr/lib/gphd/gphdmgr/lib/client/InputReaders.py
sed -i 's/8080/8085/g' /usr/lib/gphd/gphdmgr/lib/client/GPHDSync.py
sed -i 's/8080/8085/g' /usr/lib/gphd/gphdmgr/lib/client/WSHelper.py
```

4. Start Pivotal Command Center again:

```
service commander start
```

6 PHD Troubleshooting

This section provides common errors you may receive and how to troubleshoot or workaround those errors.

6.1 Debugging Errors

Pivotal Command Center has many different log files. Finding the exact log may initially be challenging at the beginning.

Here is a quick guide on how to identify the issues:

6.1.1 Pivotal HD Installation

All installation errors will be logged under:

```
/var/log/gphd/gphdmgr/installer.log
```

6.1.2 Cluster Deployment

If you see a 500 Internal Server Error, check the following logs for details:

```
/var/log/gphd/gphdmgr/gphdmgr-webservices.log
```

If you see Puppet cert generation errors, check

```
/var/log/gphd/gphdmgr/gphdmgr-webservices.log
```

If config properties are not making into the cluster nodes, check

```
/var/log/gphd/gphdmgr/gphdmgr-webservices.log
```

If you see `GPHDClusterInstaller.py` script execution error, check

```
/var/log/gphd/gphdmgr/GPHDClusterInstaller_XXX.log
```

Sometimes `/var/log/messages` can also have good information especially if the deployment fails during the puppet deploy stage.

In general if something fails on the server side, look at the logs in this order:

```
/var/log/gphd/gphdmgr/gphdmgr-webservices.log
```

```
/var/log/gphd/gphdmgr/GPHDClusterInstaller_XXX.log
```

```
/var/log/messages
```

6.1.3 Cluster Nodes Installation

If there are no errors on the admin side, but the installation failed on the cluster nodes, check the latest log file:

```
/tmp/GPHDNodeInstaller_XXX.log
```

Search for the first occurrence of the word `merr` that will point to the most probable issue.

6.1.4 Services Start

Check for the corresponding log file under `/var/log/gphd/` directory.

For example, if the namenode doesn't start, look at the

`/var/log/gphd/hadoop/hadoop-hdfs-namenode-hostname.log` file for details.

6.2 Puppet SSL Errors

For errors like:

```
"Unable to generate certificates"
```

```
"SSLv3 authentication issues on the client"
```

As root, do the following:

Ensure the hostname on all machines is a fully qualified domain name. (see the `HOSTNAME` field in `/etc/sysconfig/network`)

Run:

```
service commander stop
```

On **all machines including cluster nodes**, run:

```
rm -rf /var/lib/puppet/ssl-icm/*
```

On the **admin node**, ensure there is no puppet master process running by running:

```
ps ef | grep puppet
```

If there is, kill `-9` any running puppet process:

```
ps -ef|grep puppet|awk '{print $2}'|xargs kill -9
```

Make sure there are no certificates listed by running:

```
puppetca list --all
```

You can run `puppetca clean --all` to clean any certificates

Restart the puppet master:

```
service puppetmaster start
```

Verify there is just one certificate:

```
puppetca list --all
```

Stop the puppet master and start nmon:

```
service puppetmaster stop  
  
service commander start
```

Now retry your deployment.

6.3 Upgrade/Reconfigure Errors

6.3.1 Following an upgrade of Command Center, unable to Start/Stop cluster with invalid hostnames

This is because there is now a check for invalid characters in cluster names.

Workaround: First reconfigure the cluster to a different name:

```
icm_client reconfigure -l <old_cluster_name> -c <config directory with new clustername>
```

Then try starting/stopping the cluster:

```
icm_client start -l <cluster_name>  
icm_client stop -l <cluster_name>
```


6.3.2 Other Upgrade/Reconfigure Errors

After upgrading PHD stack from 1.0.2 to 1.0.3 release, hbase master fails to start if hbase-master is not co-located with either namenode or datanode.

Workaround: On hbase-master node, run: `yum upgrade hadoop-hdfs`. Go to `/usr/lib/gphd` directory. Point the `hadoop-hdfs` symlink to the newer `hadoop-hdfs` version.

If see a `hostRoleMapping should not be changed for other services` error, make sure the `clusterconfig.xml` file has not been changed for any of the already existing services. Even if it is the same set of hosts but in a different order, ensure to maintain the order in the comma separated list.

If you see `ERROR:Fetching hadoop rpm name on namenode: <host> failed` error, it is most likely a case where the cluster was being upgraded from 1.0.0 to 1.0.2 and there was an error during upgrade.

Workaround: Run `yum install hadoop-2.0.2_alpha_gphd_2_0_1_0-14.x86_64` on the namenode and retry upgrade.

If you are upgrading a cluster with hbase, hive or pxf configured as a service you must manually reinstall those services. See [Upgrading PHD Using the CLI](#) for details.

6.4 HA-related Errors

If the cluster fails to start with HA enabled:

- Check status of journal node (`/etc/init.d/hadoop-hdfs-journalnode status`) on all hosts and ensure they are running.
- Check if the "namenode" (configured as `namenodeid1` in `clusterconfig.xml`) is formatted and successfully started. Be sure to check `/var/log/gphd/gphdmgr/gphdmgr-webservices.log` and if needed the namenode logs on the namenode host:
`/usr/lib/gphd/hadoop/logs/hadoop-hdfs-namenode*log`
- Check if the "standbynamenode" (configured as `namenodeid2` in `clusterconfig.xml`) is formatted and successfully started. The namenode logs should have details on any errors if the standbynamenode failed to format or start.
- If standbynamenode fails to start because it is not formatted and restarting the cluster does not format the name node please contact support team for help.
- If you are converting a non-HA cluster to HA please make sure to follow the documented steps. It is important to start the journal nodes and initialize the edit logs from the namenode of the existing cluster before starting the cluster.

6.5 Other Errors

6.5.1 Cluster Deployment Fails due to RPM Dependencies

Ensure that the base OS repo is available. You might have to mount the CD that comes with the OS installation or point yum to the correct location such as NFS mount point on all the cluster nodes

6.5.2 Unable to access the Namenode Status Web page

If the host returns a short hostname instead of FQDN for `hostname()`, it is possible that the namenode status link cannot be accessed from external networks.

The solution is to either ensure that the `hostname()` returns FQDN on the namenode host, or change the `dfs.http.address` value to `0.0.0.0` in the `hdfs-site.xml` and restart namenode.

```
<property>
<name>dfs.http.address</name>
<value>0.0.0.0:50070</value>
</property>
```

6.5.3 Installation Fails due to Directory Permissions

Check if the umask is set to 0022. If not, set the umask in the `.bashrc` as "umask 0022", then retry the PCC installation.

6.5.4 Deployment Fails due to Problems with YUM Repository

Verify that the admin node is reachable from the agent node.

If you have configured proxy servers, refer the section titled [Working with Proxy Servers](#).

6.5.5 Installation Fails due to Problems with the SSL certificate

Check if `dnsdomainname` returns empty value. If yes, you need to ensure that the `dnsdomainname` returns the correct domain.

6.5.6 Cluster Node Installation Failure without Generating a Log File

Ensure that passwordless ssh is setup between the admin node and the cluster nodes.

Ensure that the puppet, facter and ruby rpms are the same as that on the admin node

Ensure that the user `gpadmin` has sudo and no requiretty access on the cluster node (check for the existence of file: `/etc/sudoers.d/gpadmin`)

Then retry the deployment.

6.5.7 Puppet certificate failure

Follow the instructions in the [Puppet SSL Errors](#) section.

6.5.8 Package Bundle Not Found

If you sudo in to the system as root, ensure that you sudo with the environment. That is: `sudo su -` Do not forget the hyphen at the end.

If you directly login as root with password and if you still see the above issue, check if `/usr/local/bin/bundle` exists. If not, build it:

```
gem install bundler
```

Add `/usr/local/bin` to `PATH`, regardless: `[]# vi ~/.bashrc`

Append `export PATH=$PATH:/usr/local/bin`, then save

```
[]# source ~/.bashrc
```

6.5.9 Cluster Deployment Fails due to Missing Packages

The above error can be identified by following the instructions on [Cluster Nodes Installation](#) errors section above.

Install `nc` and `postgres-devel` packages on all the cluster nodes or point them to a repo that contains the rpms.

6.5.10 Working with Proxy Servers

It is sometimes required that all outgoing http traffic use a HTTP proxy. PCC installer sometimes pulls rpms from an external repos such as an EPEL6 repo if the external repos are configured and if any packages are missing on the host.

If you configure the proxy settings in `/etc/yum.conf` the cluster node, cluster deployments might fail because yum will send all `gphd.repo` requests to the proxy which in turn will fail to connect to the admin node local repo.

Here are a few workarounds:

Workaround 1

- Remove the proxy settings from `yum.conf` and
- Make sure following params are set in `~root/.bashrc`

For example:

```
export http_proxy=http://proxy:3333
export no_proxy=local.domain ## this is the local domain for hadoop cluster
```

- Modify these files so `gphd.repo` gets pushed out with a FQDN name instead of shortname:
`/etc/puppet/modules/yumrepo/templates/yumrepo.erb`

Change from:

```
baseurl=http://<%= scope.lookupvar("params::config::admin_host") %>/<%=
scope.lookupvar("params::config::repopath") %>
```

Change to:

```
<replace node.full.domain.com> with the FQDN of the admin node
baseurl=http://node.full.domain.com/<%= scope.lookupvar("params::config::repopath") %>
```

Workaround 2

- Enable NFS and export `/usr/lib/gphd/rpms` to all cluster nodes
- Mount the nfs repo on all cluster nodes:

```
mount gpcc:/usr/lib/gphd/rpms /local_repo
```

- Modify these files:
`/etc/puppet/modules/yumrepo/templates/yumrepo.erb`

Change from:

```
baseurl=http://<%= scope.lookupvar("params::config::admin_host") %>/<%=  
scope.lookupvar("params::config::repopath") %>
```

Change to:

```
baseurl={nolink:file:///local_repo/}
```

6.5.11 Capital Letters in Hostname

PCC fails to deploy if the hostnames contain uppercase letters. For example: `Node0781.domain.com`.

Rename the hostname with only lowercase letters before proceeding with the deployment.

6.5.12 Resolving postgres port Conflict Issue

If you face a postgres port conflict or wish to change the default postgres port, follow the steps below:

1. Stop PCC service:

```
root# service commander stop
```

2. Add the new port <hostname>:5435 in the Pivotal HD properties file: `vim /etc/gphd/gphdmgr/conf/gphdmgr.properties`

```
gphdmgr.db.url=jdbc:postgresql://localhost:5435/gphdmgr
```

3. Change the port number in `postgresql.conf`:
`vim /var/lib/pgsql/data/postgresql.conf "port = 5435"`
4. Edit the `init.d/postgresql` file: `vim /etc/init.d/postgresql`

```
#Change the PGPORT to 5435 "PGPORT=5435"  
root# service commander start
```

6.5.13 Resolving HTTP Port Conflict

Check the FAQ section: [How do I change the port used by Pivotal HD?](#)

6.5.14 Errors like Ambit: Push Failed

If you see errors like the following:

```
root# icm_client add-user-gpadmin -f hosts
Ambit : Push Failed
Had : Push Failed
Issues : Push Failed
Generating : Push Failed
A : Push Failed
List : Push Failed
```

This is an ambit bug. If there are hostnames (only the name part, not the domain) which are substrings of other hostnames then this issue can occur.

For example: host1.emc.com, host11.emc.com

This error can be ignored for now as the actual deployment still goes through.

6.5.15 Preparehosts Errors Out While Creating gpadmin User

Make sure SELinux needs to be either disabled or in permissive mode for the hosts.

(See the *Pivotal Command Center Installation and User Guide* for instructions to disable SELinux)

6.5.16 HAWQ Initialization Failing

Make sure your cluster is up and running with the hadoop services prior to initializing HAWQ (`hawq init`). If the failure still persists, make sure the HAWQ nodes have been prepared (refer to `prepare-hawq-hosts`) to reflect the kernel configurations required for HAWQ. If you still have a problem you might be running short of the memory required to run HAWQ at scale. Refer to the *HAWQ Administrator Guide* to configure/modify the system memory requirements.

6.5.17 Installing HAWQ on Dirty Cluster Nodes Previously Configured with HAWQ

If you wish to deploy or initialize HAWQ on

- a) Cluster which had an older uninstalled HAWQ cluster, or
- b) Cluster that failed in its attempts to initialize HAWQ

You will need to do the following before initializing HAWQ with the new cluster nodes:

Ensure that `HAWQ_Hosts.txt` contains all the HAWQ hosts that you want to clean up.

Run the following command against each DIRECTORY configured in `<hawq.segment.directory>` and in `<hawq.master.directory>` in the cluster configuration (`clusterConfig.xml`)

```
gpadmin# massh HAWQ_Hosts.txt verbose 'sudo rm -rf DIRECTORY/*'
```

The above command cleans up the stale HAWQ master and segment data directory contents.

6.5.18 Errors Related to VM Memory

If you are planning to deploy a HAWQ cluster on VMs with memory lower than the optimized/recommended requirements, you may encounter `Could not create the Java virtual machine type errors`. In these cases you can reconfigure memory usage, as follows:

- Remove the entry `vm.overcommit_memory = 2` from `/usr/lib/gphd/gphdmgr/hawq_sys_config/sysctl.conf` prior to running the prepare HAWQ utility.
- In the `clusterConfig.xml`, update `<hawq.segment.directory>` to include only one segment directory entry (instead of the default 4 segments).