Apache Hadoop Operations for Production Systems

Philip Zeyliger, Philip Langdale, Kathleen Ting, Miklos Christine
Strata Hadoop San Jose, 18 February 2015
Your Hosts

Philip Langdale

Kathleen Ting

Philip Zeyliger

Miklos Christine
Philip Langdale
- Architect of Cloudera Manager
- philipl@

Kathleen Ting
- Technical account manager for large Hadoop clusters
- kate@

Philip Zeyliger
- Architect of Cloudera Manager
- philip@

Miklos Christine
- Systems engineer for large Hadoop clusters
- mwc@
Overall Agenda

- Intro
- Installation
- Configuration
(stretch) (official break 3-3:30)
- Troubleshooting
- Enterprise Considerations
- Q & A

Q&A at end of every section
Hands On

We’ve set up a handful of clusters in various configurations.

URLs and passwords:

http://tiny.cloudera.com/strata1 (user1/strata2015) (Kerberos)
admin@54.153.123.57 pass: admin321

http://tiny.cloudera.com/strata2 (user2/strata2015)
admin@54.67.90.80 pass: admin321

admin@54.153.83.11 pass: admin321
Why Apache Hadoop?

Solves problems that don’t fit on a single computer.

Doesn’t require you to be a distributed systems person.

Handles failures for you.
A Distributed System

Many processes, which, taken together, are trying to act like a single system.

Ideally, users deal with the system as a whole.

For operations, you need to understand the system by parts too.
The Hadoop Stack

User interface: HUE

Languages/APIs: Hive, Pig, Crunch, Kite, Mahout

Batch processing: MapReduce
In Mem processing: Spark
Interactive SQL: Impala
Search: Solr

Resource Management: YARN

Random Access Storage: HBase
File Storage: HDFS

Coordination: ZooKeeper, Security: Sentry

Other systems: httpd, sas, custom apps etc

Event ingest: Flume, Kafka
DB Import Export: Sqoop
Storage

Random Access Storage: HBase

File Storage: HDFS

Search: Solr
Execution

Batch processing: MapReduce

In Mem processing: Spark

Interactive SQL: Impala

Search: Solr
Compilers

Languages/APIs: Hive, Pig, Crunch, Kite, Mahout

Hive—SQL to MR/Spark compiler, metadata mapping between files and tables

Pig—PigLatin to MR compiler
In Mem processing:
Spark

Interactive SQL:
Impala

Search:
Solr

Resource Management:
YARN

Random Access Storage:
HBase

File Storage:
HDFS

Languages/APIs:
Hive, Pig, Crunch, Kite, Mahout

Batch processing:
MapReduce

In Mem processing:
Spark

Event ingest:
Flume, Kafka

DB Import Export:
Sqoop

Coordination:
ZooKeeper; Security: Sentry

Other systems:
httpd, sas, custom apps etc

User interface:
HUE

Disk, CPU, Mem

JVM

Linux
How to learn these things...

These systems are largely defined by the state they store on disk and the defined protocols (RPCs) they use to interact amongst themselves.
Questions?
Apache Hadoop Operations for Production Systems: Installation

Philip Langdale
Agenda

- Hardware Considerations
  - Node types and recommended role allocations
  - Host configuration
  - Rack configuration
- Software Installation
  - OS Prerequisites
  - Installing Hadoop and other Ecosystem components
- Launch
  - Initial Configuration
  - Sanity testing
  - Security considerations
Hardware Considerations

- As a distributed system, Hadoop is going to be deployed onto multiple interconnected hosts
- How large will the cluster be?
- What services will be deployed on the cluster?
  - Can all services effectively run together on the same hosts or is some form of physical partitioning required?
- What role will each host play in the cluster?
  - This impacts the hardware profile (CPU, Memory, Storage, etc)
- How should the hosts be networked together?
# Host Roles within a Cluster

For larger clusters, roles will be spread across multiple nodes of a given type (except workers).

<table>
<thead>
<tr>
<th>Master Node</th>
<th>Worker Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS NameNode</td>
<td>HDFS DataNode</td>
</tr>
<tr>
<td>YARN ResourceManager</td>
<td>YARN NodeManager</td>
</tr>
<tr>
<td>HBase Master</td>
<td>HBase RegionServer</td>
</tr>
<tr>
<td>Impala StateStore</td>
<td>Impalad</td>
</tr>
<tr>
<td>ZooKeeper</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Utility Node</th>
<th>Edge Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational Database</td>
<td>Gateway Configuration</td>
</tr>
<tr>
<td>Management (eg: CM)</td>
<td>Client Tools</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td>Hue</td>
</tr>
<tr>
<td>Oozie</td>
<td>HiveServer2</td>
</tr>
<tr>
<td>Impala Catalog Server</td>
<td>Ingest (eg: Flume)</td>
</tr>
</tbody>
</table>
# Roles vs Cluster Size

<table>
<thead>
<tr>
<th>Size</th>
<th>Master</th>
<th>Worker</th>
<th>Utility</th>
<th>Edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Small (≤10)</td>
<td>1</td>
<td>≤10</td>
<td>1 shared Host</td>
<td></td>
</tr>
<tr>
<td>Small (≤20)</td>
<td>2</td>
<td>≤20</td>
<td>1 shared Host</td>
<td></td>
</tr>
<tr>
<td>Medium (≤200)</td>
<td>3</td>
<td>≤200</td>
<td>2</td>
<td>1+</td>
</tr>
<tr>
<td>Large (≤500)</td>
<td>5</td>
<td>≤500</td>
<td>2</td>
<td>1+</td>
</tr>
</tbody>
</table>
Host Hardware Configuration

- **CPU**
  - There’s no such thing as too much CPU
  - Jobs typically do not saturate their cores, so raw clock speed is not at a premium
  - Cost and Budget are the major factors here

- **Memory**
  - You really don’t want to overcommit and swap
  - Java heaps should fit into physical RAM with additional space for OS and non-Hadoop processes
Host Configuration (cont.)

- **Disk**
  - More spindles == More I/O capacity
  - Larger drives == lower cost per TB
  - More hosts with less capacity increases parallelism and decreases re-replication costs when replacing a host
  - Fewer hosts with more capacity generally means lower unit cost
  - Rule of thumb: One disk per two configured YARN vcores
  - Lower latency disks are generally not a good investment, except for specific use-cases where random I/O is important
A Hadoop Machine

- Hadoop Daemons
- JVM
- Linux
- Disk, CPU, Mem
A Hadoop Rack

Rack

Machine
Hadoop Daemons
JVM
Linux
Disk, CPU, Mem

Machine
Hadoop Daemons
JVM
Linux
Disk, CPU, Mem

Machine
Hadoop Daemons
JVM
Linux
Disk, CPU, Mem

Top of Rack Switch
A Hadoop Cluster

Cluster

Backbone Switch

Rack

Top of Rack Switch

Rack

Top of Rack Switch

Machine

Hadoop Daemons

JVM

Linux

Disk, CPU, Mem

Machine

Hadoop Daemons

JVM

Linux

Disk, CPU, Mem

Machine

Hadoop Daemons

JVM

Linux

Disk, CPU, Mem
Rack Configuration

- Common: 10Gb or (20Gb bonded) to the server, 40Gb to the spine
- Cost sensitive: 2Gb bonded to the server, 10Gb to the spine
  - This is likely to be a false economy, with the real potential of being network bottlenecked with disks idle
- Look for 25/100 in the next couple of years
- Network I/O is generally consumed by reading/writing data from disks on other nodes
  - Teragen is a useful benchmark for network capacity
  - Typically 3-9x more intensive than normal workloads
Software Installation

- Linux Distribution
- Operating System Configuration
- Hadoop Distribution
- Distribution Lifecycle Management
Linux Distributions

- All enterprise distributions are credible choices
- Do you already buy support from a vendor?
- Which distributions does your Hadoop distro support?
- What are you already familiar with
- Cloudera supports
  - RHEL 5.x, 6.x
  - Ubuntu 12.04, 14.04
  - Debian 6.x, 7.x
  - SLES 11
- RHEL 6.x is the most common
Operating System Configuration

- Turn off IPTables (or any other firewall tool)
- Turn off SELinux
- Turn down swappiness to 0/1 (depending on what kernel you have)
- Turn off Transparent Huge Page Compaction
- Use a network time source to keep all hosts in sync (also timezones!)
- Make sure forward and reverse DNS work on each host to resolve all other hosts consistently
- Use the Oracle JDK - OpenJDK is subtly different and may lead you to grief
Cloudera Manager provides a Host Inspector to check for these situations.
Hadoop Distributions

- Well, we’re obviously biased here
- CDH is a jolly good Hadoop Distro. We recommend it
- You’re free to try others
- While it’s technically possible to have a go at running a cluster without using any management tools, it’s not going to be fun and we’re not going to talk about that much
Distribution Lifecycle Management

- Theoretically, you might want to just install the binaries for services and programs you are running on a specific node
- But honestly, space is not at that much of a premium
  - Install everything everywhere and don’t worry about it again
  - If you decide to alter the footprint of a service, you don’t need to worry about binaries
  - You don’t need different profiles for different hosts in the cluster
  - Cloudera Manager works this way
Distribution Lifecycle Management (cont.)

- For CDH, Cloudera Manager can handle lifecycle management through parcels
- For package based installations, there are a variety of recognised options
  - Puppet, Chef, Ansible, etc
- But please use something
  - Long term package management by hand is a recipe for disaster
  - Exposes you to having inconsistencies between hosts
    - Missing Packages
    - Un-upgraded Packages
Lifecycle Management with Parcels in Cloudera Manager

Cloudera recommends the use of parcels for installation over packages, because parcels enable Cloudera Manager to easily manage the software on your cluster, automating the deployment and upgrade of service binaries. Electing not to use parcels will require you to manually upgrade packages on all hosts in your cluster when software updates are available, and will prevent you from using Cloudera Manager’s rolling upgrade capabilities.

A parcel deployment consists of the following steps:

1. Download: A parcel first needs to be downloaded. After the download is complete, the parcel will reside in a local directory on the Cloudera Manager host.
2. Distribute: After the parcel has been downloaded, it will be distributed to all the hosts in the cluster and unpacked.
3. Activate: Once distributed, activating the parcel prepares it to be used by the cluster after a restart. An upgrade may also be necessary.
Installation with Cloudera Manager

Cluster Installation

Installation in progress.

1 of 10 host(s) completed successfully. Abort Installation

<table>
<thead>
<tr>
<th>Hostname</th>
<th>IP Address</th>
<th>Progress</th>
<th>Status</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>philip03-1.vpc.cloudera.com</td>
<td>172.26.12.142</td>
<td>✅</td>
<td>✔️ Installation completed successfully.</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-10.vpc.cloudera.com</td>
<td>172.26.13.162</td>
<td></td>
<td>_INSTALLING jdk package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-2.vpc.cloudera.com</td>
<td>172.26.13.195</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-3.vpc.cloudera.com</td>
<td>172.26.13.199</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-4.vpc.cloudera.com</td>
<td>172.26.15.247</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-5.vpc.cloudera.com</td>
<td>172.26.12.15</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-6.vpc.cloudera.com</td>
<td>172.26.14.57</td>
<td></td>
<td>🟢 Installing oracle-j2sdk1.7 package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-7.vpc.cloudera.com</td>
<td>172.26.12.55</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
<tr>
<td>philip03-8.vpc.cloudera.com</td>
<td>172.26.12.51</td>
<td></td>
<td>🟢 Installing cloudera-manager-agent package...</td>
<td>Details</td>
</tr>
</tbody>
</table>
Installation with Cloudera Manager

Cluster Installation

Installing Selected Parcels
The selected parcels are being downloaded and installed on all the hosts in the cluster.

CDH 5.3.1-1.cdh5.3.1.p0.5

Downloaded

Distributing 28%

Activating 0%
Cluster Setup

Choose the CDH 5 services that you want to install on your cluster.

Choose a combination of services to install.

- **Core Hadoop**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, and Sqoop
- **Core with HBase**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and HBase
- **Core with Impala**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Impala
- **Core with Search**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Solr
- **Core with Spark**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Spark
- **All Services**
  - HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, HBase, Impala, Solr, Spark, and Key-Value Store Indexer
- **Custom Services**
  - Choose your own services. Services required by chosen services will automatically be included. Flume can be added after your initial cluster has been set up.
Installation with Cloudera Manager

Cluster Setup

Customize Role Assignments

You can customize the role assignments for your new cluster here, but if assignments are made incorrectly, such as assigning too many roles to a single host, this can impact the performance of your services. Cloudera does not recommend altering assignments unless you have specific requirements, such as having pre-selected a specific host for a specific role.

You can also view the role assignments by host. [View By Host]

HBase

- Master x 1 New
  - philip03-2.vpc.cloude...
- HBase REST Server
  - Select hosts
- HBase Thrift Server
  - Select hosts
- RegionServer x 8 New
  - Same As DataNode

HDFS

- NameNode x 1 New
  - philip03-2.vpc.cloude...
- SecondaryNameNode
  - philip03-2.vpc.cloude...
- Balancer x 1 New
  - philip03-2.vpc.cloude...
- HttpFS
- NFS Gateway
  - Select hosts
- DataNode x 8 New
  - philip03-[3-10].vpc.cloude...

Hive

- Gateway x 9 New
  - philip03-[2-10].vpc.cloude...
- Hive Metastore Server
  - philip03-2.vpc.cloude...
- WebHCat Server
- HiveServer2 x 1 New
  - philip03-2.vpc.cloude...

cloudera
Launch

• Initial Configuration
• Sanity testing
• Security Considerations
Initial Configuration

- Recall our earlier discussion of hardware
  - YARN vcores (or MapReduceV1 slots) proportional to physical cores
    - Typically 1/1.5:1 (1.5 with hyperthreading)
- Heap sizes
  - Don’t overcommit on memory, but don’t make Java heaps too large (GC)
  - See Memory table in Appendix
- Mounting data disks
  - One partition per disk
  - No RAID
  - Use a well established filesystem (ext4)
  - Use a uniform naming scheme
Initial Configuration (cont.)

- Think about space on the OS partition(s)
  - /var is where your logs go by default
  - /opt is where Cloudera Manager stores parcels
  - By default, these are part of / on modern distros, which works out fine
- Your IT policies may require separate partitions
  - If these areas are too small, then you’ll need to change these configurations
Cloudera Manager can help

- Assigns roles based on cluster size
- Tries to detect masters based on physical capabilities
- Sets vcore count based on detected host CPUs
- Sets heap sizes to avoid overcommitting RAM
- Autodetects mounted drives and assigns them for DataNodes
Initial Service Configuration in Cloudera Manager
Implications of Services in Use

- Different services running concurrently means we need to consider how resources are shared between them
- Services that use YARN can be managed through YARN’s scheduler
- But certain services do not - most visibly HBase, but also services like Accumulo or Flume
  - These services can run on a shared cluster through the use of static resource partitioning with cgroups
- Cloudera Manager can configure these
Dynamic Resource Management

Applications can run in a pool based on the user, the group of the submitting user, as well as specific pools and the default pool. Allocate resources across pools using weights, minimum, and maximum limits. Configuration sets allow switching on different weight and limit settings activated by user-defined schedules.

Pools can be nested, each level of which can support a different scheduler, such as FIFO or fair scheduler. Each pool can be configured to allow only a certain set of users and groups to access the pool.

<table>
<thead>
<tr>
<th>Name</th>
<th>Weight</th>
<th>Virtual Cores Min / Max</th>
<th>Memory Min / Max</th>
<th>Max Running Apps</th>
<th>Scheduling Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>1</td>
<td>- / -</td>
<td>- / -</td>
<td>-</td>
<td>DRF</td>
</tr>
<tr>
<td>default</td>
<td>1</td>
<td>33.3%</td>
<td>- / -</td>
<td>-</td>
<td>DRF</td>
</tr>
<tr>
<td>priority</td>
<td>2</td>
<td>66.7%</td>
<td>- / -</td>
<td>-</td>
<td>DRF</td>
</tr>
</tbody>
</table>
Static Resource Management

Static Service Pools: Status

Step 1 of 4: Basic Allocation Setup

<table>
<thead>
<tr>
<th>Service</th>
<th>Allocation %</th>
</tr>
</thead>
<tbody>
<tr>
<td>HBase</td>
<td>30 %</td>
</tr>
<tr>
<td>HDFS</td>
<td>10 %</td>
</tr>
<tr>
<td>Impala</td>
<td></td>
</tr>
<tr>
<td>Solr</td>
<td>10 %</td>
</tr>
<tr>
<td>YARN (MR2 Included)</td>
<td>50 %</td>
</tr>
<tr>
<td>Total</td>
<td>100 %</td>
</tr>
</tbody>
</table>

Help

Static service pools serve to isolate services from each other, so that high load on one service has bounded impact on other services.

To start, subdivide the cluster using percentages on the left. Cloudera Manager will then suggest settings for the worker roles of the services that correspond to the percentages. You will have a chance to review the changes and restart the cluster. Learn more

This is the coarser-grained, more static partner of Dynamic Resource Pool Configuration.
Sanity Testing

- Use the basic sanity tests provided by each service
  - Submit example jobs: pi, sleep, teragen/terasort
  - Work with sample tables in Hive/Impala
  - Use Hue to do these things if your users will
- Repeat these tests when turning on security/authentication/authorization mechanisms
  - Make sure they succeed for the expected users and fail for others
- Cloudera Manager provides some ‘canary’ tests for certain services
  - HDFS create/read/write/delete
  - Hbase, Zookeeper, etc
HDFS Health tests

- NameNode summary: philip03-2.vpc.cloudera.com (Availability: Active, Health: Good)
- Space free in the cluster: 188.0 GiB, Capacity of the cluster: 188.9 GiB, Percentage of capacity free: 99.52%.
- 0 blocks with corrupt replicas in the cluster, 299 total blocks in the cluster. Percentage blocks with corrupt replicas: 0.00%.
- 0 missing blocks in the cluster, 299 total blocks in the cluster. Percentage missing blocks: 0.00%.
- 0 under replicated blocks in the cluster, 299 total blocks in the cluster. Percentage under replicated blocks: 0.00%.
- Canary test of file create, write, read and delete operations succeeded.
- Healthy DataNode: 7, Concerning DataNode: 0. Total DataNode: 7, Percent healthy: 100.00%. Percent healthy or concerning: 100.00%.

Health History
- 11:42:02 AM  7 Became Good
- 11:41:42 AM  7 Became Disabled
### Hue Examples

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fs</td>
<td>Example of Fs action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Hive</td>
<td>Example of Hive action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Ssh</td>
<td>Example of Ssh action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Sqoop</td>
<td>Example of Sqoop action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>DistCp</td>
<td>Example of DistCp action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>MapReduce</td>
<td>Example of MapReduce action that sleeps</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Shell</td>
<td>Example of Shell action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Pig</td>
<td>Example of Pig action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Email</td>
<td>Example of Email action</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>DailyAnalytics</td>
<td>Run daily a workflow with a date range of input data</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>DailySleep</td>
<td>Run a daily sleep job</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Aggregate</td>
<td>Aggregate together and batch 2 coordinators</td>
<td>02/17/15 12:21:06</td>
</tr>
<tr>
<td>Sample: Job loss</td>
<td>Job loss among the top earners 2007-08</td>
<td>02/17/15 12:21:09</td>
</tr>
<tr>
<td>Sample: Salary growth</td>
<td>Salary growth (sorted) from 2007-08</td>
<td>02/17/15 12:21:09</td>
</tr>
<tr>
<td>Sample: Top salary</td>
<td>Top salary 2007 above $100k</td>
<td>02/17/15 12:21:09</td>
</tr>
</tbody>
</table>
Appendix