Hadoop Operations

Jennifer Davis

LISA 2014 Tutorial

Establishing Expectations

- NOT downloading hadoop to laptops.
  - Repo you can do later.
  - Pseudo-distributed deployment
- Intended Audience: Beginners to Hadoop

Too Basic? Other Options

— SRE Classroom Practical Large Scale Design
— Statistics for Ops
— Tatical Capacity Planning for Sysadmins
— Working like a Team
— R for Sysadmins
— Intro to Ceph

Agenda

- Overview
- Intro to Hadoop
- Security
- Hadoop Deployments
- Configuration

Agenda

— Monitoring
— Backups
— Troubleshooting
— Operability Issues
— Resources

BREAKS!

10:30-11am - Mid-morning
12:30-1:30 - Lunch
3-3:30 - Mid-afternoon

Restrooms

Out the door and on the opposite wall.

The elephant in the room...

power.

First 3 rows have power!
Please try to recharge if needed during breaks/lunch.

Introductions

— Who are you?

— Why are you here?

— Previous experience?

Verify your system

Find your username:

`https://raw.githubusercontent.com/iennae/hadoopops.org/master/class_name_ip`

LINK

ssh chef@IP
password chef

Discussion: Big Data

— What is Data?

— What is Big?

Why does "Big Data" Matter?

— Storing data
— Working with data
— Transformation of data into meaningful information

Distributed Systems

Distributed File System

Types of Data

— Structured
— Semi-structured
— Unstructured

Characteristics of Data

— Volume

— Variety

— Velocity

Why this tutorial

— Not enough Operations minded folks within the Hadoop Community

— Not enough beginner resources for Sys Admins

Intro to Hadoop

Hadoop

— store & compute
— distributed system/ distributed filesystem
— parallel execution of programs
— HDFS, MapReduce, YARN

Hadoop Common?

— set of common utilities that support other Hadoop modules

Limitations

— Availability

— Security

— Multi-tenancy

— Multiple Data Centers

— Configuration

XML

Limitations - Recommendation 1

Automate Configuration management.

— Don't hand craft on servers.

— This is #1 reason why folks choose vendor packages!

Limitations - HDFS

— Inefficiency in small files

— Nontransparent compression

— No random writes

Limitations - MapReduce

— Not real time

— No global synchronization

Limitations - YARN

— long running processes YARN-896
— complex - container management and fault tolerance
— gang scheduling YARN-624

Hadoop Users

— Web Scale

— Enterprise

— Government

— Health Care

The list goes on …

Hadoop Users

— Data Scientists
— Data Engineers
— Developers
— Research
— Business
— System Admins

Hadoop Contributors

— Cloudera
— Hortonworks
— Yahoo
— Facebook
— Twitter
— You?

A very brief history ...

History

— Doug Cutting Scalability with Nutch

— Google GFS, MapReduce papers

— Apache 1st class project

Hadoop Ecosystem

Ambari	Chukwa	Pig
Avro	Hbase	Spark
Cassandra	Hive	Tez
Mahout	Hcatalog	Zookeeper

Hadoop Ecosystem

— rich

— diverse

— overlapping

— growing

Evaluating new technology?

— know where to find info
— know current resources and support strategies
— introduction of new systems, impact and supportability

Terminology

— Job

— Task

— Task attempt

Terminology

— Slot

— Container

— ETL

Hadoop Intro Questions

— what is hadoop?
— what are the components of core hadoop?
— name examples of other software that is part of the hadoop ecosystem.


Security

Security: Layers

- authentication
- authorization
- accounting
- data protection

Security: Authentication

— simple authentication and kerberos

— hadoop.security.authentication

LDAP or Active Directory

Security: Authorization

- HDFS - file permissions
- MapReduce - resource level
- HBase - table/column family ACL
- Hive - table

Authorization - Recommendation

— set up gateways for users

— firewall your hadoop cluster

Security: Accounting

— logging

Accounting - Recommendation

— store hadoop logs in hadoop

— know how to "ask" who did what and when

Accounting - Technology

— Flume - streaming data
— Splunk Hadoop Connect

Security: Data Protection

— operating system level encryption

Governance

Governance

— who decides policy?

Governance

— what kind of data can be loaded?
— who can have access to data?
— how long can it be stored?
— who controls schema changes?
— when can you cleanse stale data?
— …

Governance - Recommendation

— Explicitly state policies.

— Ensure technology implemented to adhere to policy.

Governance - Technology

— Configuration management
— Event handers
— Oozie - workflow scheduler
— Sqoop - bulk data transfer
— Flume - streaming data

Security and Governance Questions

— What are some of the questions you want to ask when figuring out your hadoop deployment?
— What are the 4 layers of security we discussed?
— What is available to you to meet those 4 layers?

Hadoop Deployments

Hardware Selection

Hardware Selection

- good quality commodity hardware
- more memory is better, error correcting!
- pilot clusters to learn about load patterns
  - establish processes

Master nodes

— more robust hardware
— run critical cluster services
— service disruption if fail
— dual power supplies
— OS separate from data

Master nodes

namenode

- RAM - metadata must fit in physical memory
- dedicated disk
- RAID-1

~1GB per 1 million blocks

Master nodes

secondary namenode

— generally identical to namenode

— with HA NN - standby NN performs the checkpoint work the secondary namenode would do

Worker nodes

— no RAID on the DN

— MapReduce - large datasets, parallel, sequential I/O access pattern

— disk failure on DN

Tune configurations

Java Virtual Machine (jvm)

— We'll dig into this more later!

Disks

— ensure enough for NN

— verbose logging

— hard to debug issues if logs are gone

Plan hardware replacement strategy

— always decommision datanodes before removing from cluster!

Sizing of cluster

— storage

— compute

— mixed job types with conflicting resource requirements

Network sizing

Types of traffic

- "housekeeping" - heartbeats, block reports, admin commands
- data transfer in and out
- size of cluster determines size of connections
- clients need to be able to talk to all of the nodes
- Really large and complex topic.

Choosing a filesystem

— don't use LVM

— mount data partitions with noatime, nodiratime

Filesystem layout

— Hadoop Home

— Datanode data directories

— Namenode directories

Filesystem layout

- MapReduce local directories
- Hadoop log directory
- Hadoop pid directory
- Hadoop temp directory

DNS

— host identification

— discovery

Hadoop Deployments

Deployment Modes

— Standalone

— Pseudo-distributed

— Fully Distributed

Standalone

— Not running separate daemons
— Single JVM (Java Virtual Machine)
— Does not use HDFS -- standard file system
— Can test MapReduce code

TLDR: Limited usefulness

Pseudo-Distributed

— All daemons running on single computer

— Most common development deployment

— HDFS backed

This is what we are going to use today!

Fully-Distributed

— Multiple nodes

Software dependencies

— Java

— ssh

— ntp

Distributions

- Apache source and binary
- Cloudera
- HDP (Hortonworks)
- MapR
- Cloud based offerings ...

Versioning

— 0.20 -> 1.0

— 0.23 -> 2.0

Lab: Format filesystem

```
$  sudo -u hdfs hdfs namenode -format
```

Lab: Start up hdfs processes

```
$ for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do sudo service $x start ; done
```

Lab: Browse the NameNode web interface

`http://namenode:50070/`

Lab: Set up /tmp

```
$   sudo -u hdfs hadoop fs -mkdir -p /tmp
$   sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
```

Lab: Create the YARN system directories:

```
$ sudo -u hdfs hadoop fs -mkdir -p /tmp/hadoop-yarn/staging/history/done_intermediate
$ sudo -u hdfs hadoop fs -chown -R mapred:mapred /tmp/hadoop-yarn/staging
$ sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
$ sudo -u hdfs hadoop fs -mkdir -p /var/log/hadoop-yarn
$ sudo -u hdfs hadoop fs -chown yarn:mapred /var/log/hadoop-yarn
```

Lab: Verify the HDFS file structure

```
$ sudo -u hdfs hadoop fs -ls -R /
```

Lab: Verify the HDFS file structure

```
$ sudo service hadoop-yarn-resourcemanager start
$ sudo service hadoop-yarn-nodemanager start
$ sudo service hadoop-mapreduce-historyserver start
```

Lab: Create /user on HDFS

```
$ sudo -u hdfs hdfs dfs -mkdir /user
```

Lab: Browse the ResourceManager web interface

`http://namenode:8088/`

Lab: Enter safe mode

```
$ sudo -u hdfs hdfs dfsadmin -safemode enter
```

Lab: Browse the NameNode web interface

`http://namenode:50070/`

Lab: Stop the hadoop processes

```
$ for x in `cd /etc/init.d ; ls hadoop-*` ; do sudo service $x stop ; done
```

Lab

— Start up the hadoop services

— Verify hadoop pseudo-distributed setup is running

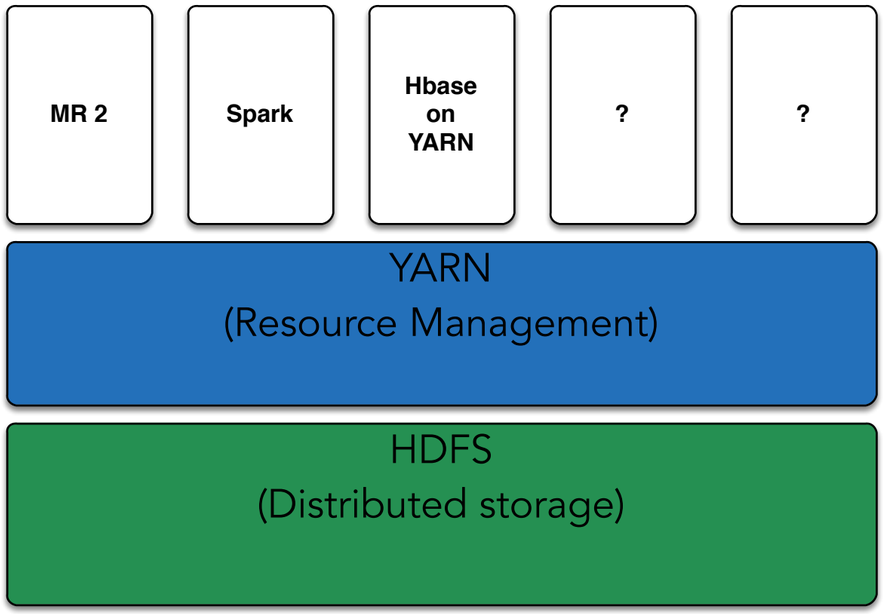
Hadoop Deployment Questions

— What are the different types of deployment modes and how are they used?

— Is RAID good for Hadoop?

— How do you verify that the namenode has started?

Architecture



Hadoop Architecture

— master/slave
— HDFS - storage partitions, data metadata
— YARN - scheduler of work
— MapReduce - compute work scheduler

HDFS Architecture

— Hadoop Distributed File System

— Primary storage system for Hadoop

— Fast and reliable

HDFS Characteristics

- Persistence
- Replicated
- Linearly scalable
- Applications sequentially stream reads
- Optimized for reads

HDFS Characteristics

- Write once and read many times
- Data stored in blocks
  - Distributed over many nodes
  - 128 MB to 1GB Block size

Goals of HDFS

— high throughput
— reading and writing of large files >GB
— scalability
— availability

HDFS

- data replication
  - # of times - configuration
  - re-replication of data automated
  - blocks on failed nodes

HDFS - Data Replication

- automatic recovery
  - parallel and fast

HDFS

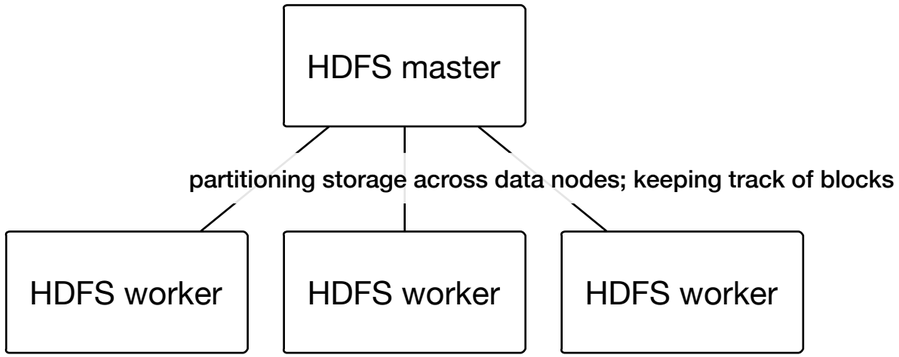
- Large Block sizes
- Data Locality Optimization - reduce network I/O
- Scalability and Availability
  - through data replication and fault tolerance

HDFS

— namenode - memory metadata

— datanode - storage

HDFS diagram



Hadoop 2

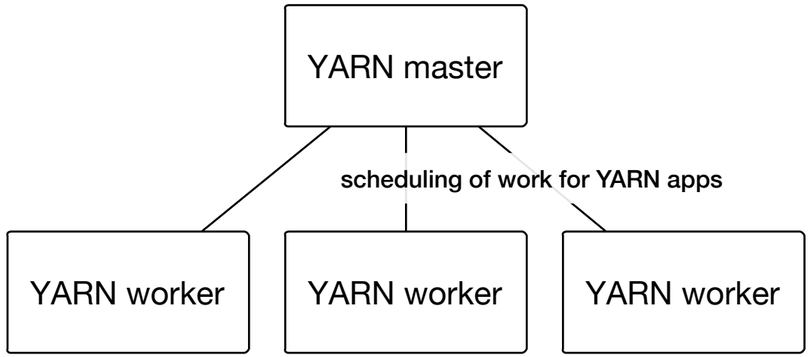
— 🆕 Federation - HDFS metadata shared across NN

— 🆕 High availability

YARN

- scheduler
- introduced with Hadoop 2
  - > 4000 node scalabity
  - added workload types stream, graph

YARN


YARN master
scheduling of work for YARN apps
YARN worker
YARN worker
YARN worker

YARN

— resource manager - "master process"

— node manager - "worker process"

YARN application

— client

— application manager

— container

Hadoop 2

— slot => container

— different memory and CPU traits

— elimination of wasted cluster resources

For people joining us

https://github.com/iennae/hadoopops.org

PDF in this directory current slides.
Modified as we go along during breaks
Also AWS allocation
user:chef
password:chef

Lab: YARN

DistributedShell application
* reference YARN application
* utility for running parallel command

```
$ sudo -u hdfs hadoop org.apache.hadoop.yarn.applications.distributedshell.Client \
--jar /usr/lib/hadoop-yarn/hadoop-yarn-applications-distributedshell.jar \
--shell_command date --num_containers 1
```

Lab: YARN

CLI to examine yarn applications

```
$ sudo -u hdfs yarn application -list
```

Lab: YARN

CLI to view logs for an application

```
$ sudo -u hdfs yarn logs -applicationId APPLICATION_ID
```

— application must be completed

— log aggregation must be enabled

YARN

Accessing logs using the YARN GUI

On pseudo-distributed setup:

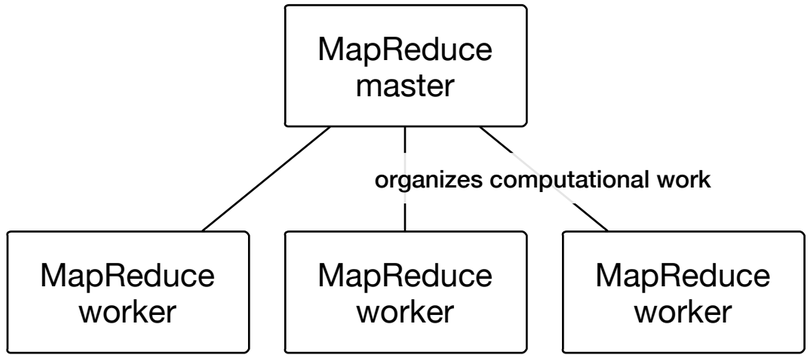
`http://YOURNAMENODE:8088/cluster`

MapReduce

— batch based distributed computing framework modeled after Google paper

Use cases:
* web logs and data to model user's behavior

MapReduce


MapReduce master
organizes computational work
MapReduce worker
MapReduce worker
MapReduce worker

MapReduce

— Hadoop 2 - MRv2
— elimination of parallel execution interdependencies
— no synchronization
— no state sharing

HDFS Ports

— RPC communication between servers

— HTTP/S

File System shell

— cli to interact with HDFS and other supported file systems

```
$ hdfs dfs <args>
```

URI

`scheme://authority/path`

HDFS:

`hdfs://namenodehost/parent/child`

or if config is set to hdfs://namenodehost

`/parent/child`

File System shell

— Example

```
$ sudo -u hdfs hdfs dfs -ls /user
```

Lab: Copy local file to HDFS

```
$ sudo -u hdfs hdfs dfs -copyFromLocal /var/log/hadoop-hdfs /user/hdfs/hadoop-hdfs
$ sudo -u hdfs hdfs dfs -ls /user/hdfs/hadoop-hdfs
```

Lab: Display size of file

```
$ hdfs dfs -du -h /user/hdfs/hadoop-hdfs
```

Lab: Try out the following FS Shell commands

```
cat         copyFromLocal   mv
chgrp       copyToLocal     rm
chmod       count           rmr
chown       cp
```

Users

User Accounts

— OS Username is hadoop username

Lab: Create a home directory for alice in HDFS

```
$ sudo -u hdfs hadoop fs -mkdir /user/alice
$ sudo -u hdfs hadoop fs -chown alice:users /user/alice
```

Lab: Verify home directory for alice created

```
$ sudo -u hdfs hadoop fs -ls /user
```

Quotas

— Name Quotas

— Space Quotas

Name Quotas

— newly created directory has no quota
— hard limit on number of file and directory names
— file and dir creation fail if quota would be exceeded
— set/remove creates a journal entry

Name quota gotchas

— Set quota lower than current standards

— Quota set to 1 forces empty directory

— Persistent with fsimage

Space Quotas

— newly created directory has no quota
— hard limit on number of *bytes*
— replicas included!
— file metadata not counted against the quota

Lab: Set the space quota for alice

```
$ sudo -u hdfs hdfs dfsadmin -setSpaceQuota 1G /user/alice
```

Lab: Verify the space quota for alice

```
$ sudo -u hdfs hadoop fs -count -q /user/alice
```

Lab: Verify the space quota for /user

```
$ sudo -u hdfs hadoop fs -count -q /user
```

dfsadmin

Administrative command

Quota Options:

— setQuota
— clrQuota
— setSpaceQuota
— clrSpaceQuota

dfsadmin - HDFS status

Lab: Get a report on the HDFS status

```
$ sudo -u hdfs hdfs dfsadmin -report
```

Why Name Quota?

— prevent too many files

Metadata stored on namenode in memory!

Trash

— "undo"
— prevents accidental deletion of a file
— deleted files moved to the /trash directory
— remain for a minimum amount of time
— each user has their own trash
— by default trash is disabled

Trash Gotchas

— User deleting file does not mean free space in HDFS immediately

Trash: Configuration

File: core-site.xml

```
<property>
  <name>fs.trash.interval</name>
  <value>DELAY_INTERVAL_IN_MINUTES</value>
</property>
```

Lab: Enable Trash

— Shutdown Hadoop processes.

— Edit file: /etc/hadoop/conf/core-site.xml

— Add a property value within the <configuration> directive.

```
<property>
  <name>fs.trash.interval</name>
  <value>1440</value>
</property>
```

— Start Hadoop processes.

Lab: New user Bob

— create user account for bob

— set bob up with a home directory

— set up quotas for bob

— copy file to bobs directory

— delete file

User Account Questions

— How do you create a user in hadoop?

— How do you set the space quota on hadoop?

— What does fs.trash.interval of 480 do?

Configuring Hadoop

Hadoop Configuration Environment

— `$HADOOP_CONF_DIR` - env configuration file location

— hadoop-env.sh - bash script

— yarn-env.sh - bash script

Configuration Environment

- `JAVA_HOME` (minimum)
- `HADOOP_PID_DIR`
- `HADOOP_SECURE_DN_PID_DIR`

Hadoop Configuration Files

Site Specific:

— core-site.xml
— mapred-site.xml
— hdfs-site.xml
— yarn-site.xml

Java Configuration Files

— log4j.properties
hadoop-metrics.properties

core-site.xml

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://namenode:8020</value>
</property>
```

Default filesystem name

Default port is 8020

core-site.xml

```
<property>
  <name>io.file.buffer.size</name>
  <value>131072</value>
</property>
```

Size of read/write buffer SequenceFiles

data buffered during read/write operations

hdfs-site.xml - NN

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/var/hdfs/name</value>
</property>
```

Directory where NameNode stores metadata

namespace and transaction logs

Can be multi-directory through comma separation

Recommendation hdfs-site.xml - NN

- multiple redundant dirs
- NFS mounted
  - `tcp,soft,intr,timeo=20,retrans=5`

hdfs-site.xml - NN

```
<property>
  <name>dfs.namenode.name.dir.restore</name>
  <value>true</value>
</property>
```

restores NN storage directory during checkpointing

hdfs-site.xml - NN

```
<property>
  <name>dfs.namenode.hosts</name>
  <value>FILENAME</value>
</property>
```

Explicitly allowed datanodes that can connect

hdfs-site.xml - NN

```
<property>
  <name>dfs.namenode.hosts.exclude</name>
  <value>FILENAME</value>
</property>
```

Explicitly excluded datanodes

hdfs-site.xml - NN Block size

```
<property>
  <name>dfs.blocksize</name>
  <value>128m</value>
</property>
```

default 64MB

configuration is in bytes or based on suffix

hdfs-site.xml - SNN

```
<property>
  <name>dfs.namenode.checkpoint.dir</name>
  <value>file://${hadoop.tmp.dir}/dfs/namesecondary</value>
</property>
```

Directory where Secondary NN stores checkpoints of metadata

hdfs-site.xml - DataNode

```
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file://{hadoop.tmp.dir}/dfs/data</value>
</property>
```

List of directories for the DataNode to store blocks

Comma delimited

Each drive

NO RAID

hdfs-site.xml - Reserving space

```
<property>
  <name>dfs.datanode.du.reserved</name>
  <value>107374182400</value>
</property>
```

default 0

reserve space on storage for non dfs use

hdfs-site.xml - Replication factor

```
<property>
  <name>dfs.replication </name>
  <value>1</value>
</property>
```

default is 3.

yarn-site.xml - RM and NM

```
<property>
  <name>yarn.acl.enable</name>
  <value>BOOLEAN</value>
</property>
```

default false

enable acls

yarn-site.xml - RM and NM

```
<property>
  <name>yarn.admin.acl</name>
  <value>*</value>
</property>
```

default * anyone

comma-separated-usersspacecomma-separated-group

yarn-site.xml - RM and NM

```
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>false</value>
</property>
```

log aggregation

JVM configutrations

- Java heap size
  - set max and starting heapsize
  - configure NewSize and MaxNewSize

System configuration

limits.conf

- open fd ulimit
  - *tune for your environment*
- 16K for DN is not too much
- 64K for Master nodes not too much

System configuration

ntp

— time synchronization is critical

System configuration

kernel tuning

- vm.swappiness = 0
- vm.overcommit_memory = 1
  - vm.overcommit_ratio - percentage added to the amount of RAM

Version Control

— store your config files in version control
— create artifacts based off config files
— deploy versioned artifacts

Configuration Management

— generally changes made to all nodes
— recommend using automation to manage
— different types of machines may require different configs

Lab: Configuration File Changes

- Place HDFS in safemode
- Stop all daemons
- Modify configurations
- Start up daemons
- Verify HDFS out of safemode

Spark

spark.apache.org - speedy general engine for large-scale data processing.
Languages: Java, Scala, Python

Monitoring

Monitoring

— cluster storage

— nodes

— files

— blocks

Monitoring Hardware

Distributed system

- single system failure not page worthy
  - unless it's the namenode 😄
- track dead datanodes handle regularly

Monitoring System

— Disk I/O

— Network I/O

— CPU

— memory

Critical for capacity planning

Monitoring

- JVM Metrics
  - GC times
  - Memory used
  - Thread status

Monitoring

- RPC metrics
  - latency

Track slowdowns!

Monitoring

- HDFS metrics
  - Used storage
  - # of files and blocks
  - Total load on the cluster
  - File system operations

Monitoring

Track cluster storage utilization

— if < 85% slow downs

Monitoring

Track rate of change cluster storage utilization

— plan regular capacity adds based on past performance

Monitoring

Corrupt blocks

— delete tmp files with replication factor of 1

— missing blocks

Monitoring

— MapReduce Metrics

Monitoring

— YARN Metrics

Logging

- $HADOOP*LOG*DIR/hadoop-$HADOOP*IDENT*STRING-<server>.log
- audit logging on the namenode

Monitoring Recommendation

— tweak during upgrades and maintenance

— change one thing at a time

Monitoring Recommendation

Cluster validation testing

— run on restart, upgrade, config changes

Backups

Backups

— Metadata

— Data?

Metadata

Namenode

— periodic backups

Data

DistCp

Hadoop Distributed Copy

Copy:
* between clusters
* within cluster

DistCp

— Uses MapReduce

— Expands file and directory listing into map tasks

Using DistCp

```
$ hadoop distcp hdfs://nn1:8020/source hdfs://nn2:8020/destination
```

What does it do

- Expands the name space under /source on namenode nn1 into a temporary file
- Partition contents among set of map tasks
- Copy from namenode nn1 to namenode nn2

DistCp with multiple sources

```
$ hadoop distcp hdfs://nn1:8020/source/a hdfs://nn1:8020/source/b hdfs://nn2:8020/destination
```

Challenges with DistCp

- Cross-check source and destination to verify successful copy
  - Problems with MapReduce, FileSystem API can impact whether it was true copy
- Source collisions errors
- Destination collisions depends on options
- If source is currently being written to…

More Challenges with DistCp

- Security
  - Copying between secure clusters - kerberos realms, principal
  - Copying from secure to insecure

DistCp: Recommendation 1

— Don't do this manually
— Set up Jenkins job and configure as needed
— Test validation of copy
— use WebHDFS

DistCp: Recommendation 2

Lots of (small) files in a directory?

— memory constraint

— set the JVM heap-size

```
$ export HADOOP_CLIENT_OPTS="-Xms64m -Xmx1024m"
```

Troubleshooting

What is trouble?

— Inaccessible?

— Exceptions?

— Slow?

What is normal?

— Always measure

— Logs

Large number of datanodes fail

Put the namenode in safemode

— avoid unnecessary replication

— bring the datanodes/rack back online

GC

— GC logging on the name node

— GCViewer

— memory usage patterns, GC pauses, misconfiguration

HDFS FsImage

— persistent checkpoint
— metadata about files and directories
— binary file
— user/group name
— candidates for duplicate datasets
— inefficient mapreduce jobs

HDFS Offline Image viewer

— converts FSImage to text format

Twitter's HDFS_DU

HDFS Capacity Planning

- analyze FSImage to learn how fast HDFS grows
- combine with info about external datasets
  - growth of dataset
  - size of dataset
  - time of dataset

Physical Hardware Challenges

- System aging
- Inconsistent hardware over time
- BIOS updates
- Multiple services installed within same rack
- Dataset Retention

BIOS updates

— physical change - rack wise pattern
— soft change - spread change across rack
— validate hardware bios prior to install

Service rack awareness

— PDU usage

what will happen if all servers get turned off

— in a rack?

— in a row?

— in a datacenter?

Performance

Performance: OS

Brendan Gregg's USE method

USE Linux Checklist

Performance: Java

- Heap
  - keeping ref to old objects

Lab: Install java-1.7.0-openjdk-devel

```
$ sudo yum install java-1.7.0-openjdk-devel
```

Performance: Java Tooling

jps

— comes with JDK (aka free)

— list all java process ids

```
$ jps -l
```

Performance: Java Tooling

jps

Specify user that java process is running as:

```
$ sudo -u USER jps -l
```

Performance: Java Tooling

jps

Example Output:

```
$ jps -l
 24154 org.apache.hadoop.hdfs.server.namenode.NameNode
 24251 org.apache.hadoop.hdfs.server.datanode.DataNode
 27520 sun.tools.jps.Jps
```

Performance: Java Tooling

JMap

— comes with JDK (aka free)

— prints memory map of a java process

```
$ jmap -heap JAVA_ID
$ jmap -dumpfile
```

Performance: Java Tooling

jhat

— comes with JDK (aka free)

— read heap dump

```
$ jhat /tmp/DUMPFILE
```

Performance: Java Tooling

jhat

— java.lang.OutofMemoryError
— uses a large amount of memory compared to dump
— starts http server on port 7000 by default
— trace instance counts for all classes
— see number of times in object spaces

Performance: Java Tooling

jhat

```
$ jhat -J-Xmx1024m /tmp/DUMPFILE
```

Performance: Java Tooling

jhat

— costly!!

— don't run in production

Performance: Java Tooling

visualvm

- free
- jvm monitoring, troubleshooting, profiling tool
- GUI
- heap dumps, memory leak info,
- garbage collection monitoring, memory profiling, CPU profiling

Performance: Java Tooling

Java Management Extensions (jmx)

Lab: Investigate jmx

In a browser or from the command line:

```
http://YOURSYSTEM:50070/jmx
http://YOURSYSTEM:8088/jmx
```

Performance: Java Tooling

jstack

— java stack traces of threads for a given java process or core file.

— lists of threads

— order of creation

```
$ jstack -l pid
```

Options

Performance: Java Tooling

jconsole

— view the Hadoop metrics available via JMX

— *-Dcom.sun.management.jmxremote*

```
bash
$ kill -3 PID
```

Operability Issues

OI: Configuration

— versioning

— clients

OI: Port Usage

— Troubleshooting

— Firewall rules

OI: Security

— kerberos

OI: Capacity Planning

— network requirements

— manual

OI: User Tuning

— Documentation focuses on the individual

— Impact to team

User Recommendation

- train users on best practices in your org
  - storage management
  - replication # for important data
  - replication factor of 1 to save space

Document best practices

User Recommendation

— provide testing cluster

— avoid debugging in production

User Recommendation

- create announce list
  - auto-subscribe new cluster users
  - automate sending out cluster management
- create mailing list for information exchange
  - facilitate users helping each other

OI: SPOF

— network outages

— users

— configuration errors

— hardware errors

OI: Debugging

— multi-threaded, multi-process, multi-host

Hadoop Issues

Hadoop Issues

- Jira
- Hadoop Common
- Hadoop HDFS
- Hadoop Map/Reduce
- Hadoop YARN
- HBase
- Hive

Create JIRAs

— help identify issues

— follow through on bug fixes

Community Sites

— Apache project site

Project Members

Hadoop project members

- Project Management Committee (PMC)
  - determines general direction of project and releases
- Project Committers

Our Journey

— Overview
— Intro to Hadoop
— Security
— Hadoop Deployments
— Configuration

Our Journey

— Monitoring
— Backups
— Troubleshooting
— Operability Issues

Other Hadoop events at LISA 2014

— Taming Operations in the Apache Hadoop Ecosystem LISA 2014

— Cloudera BOF

— Hadoop Operations BOF

Next Steps

— Subscribe to mailing lists

— Follow Allen Wittenauer `@_a__w_` on twitter

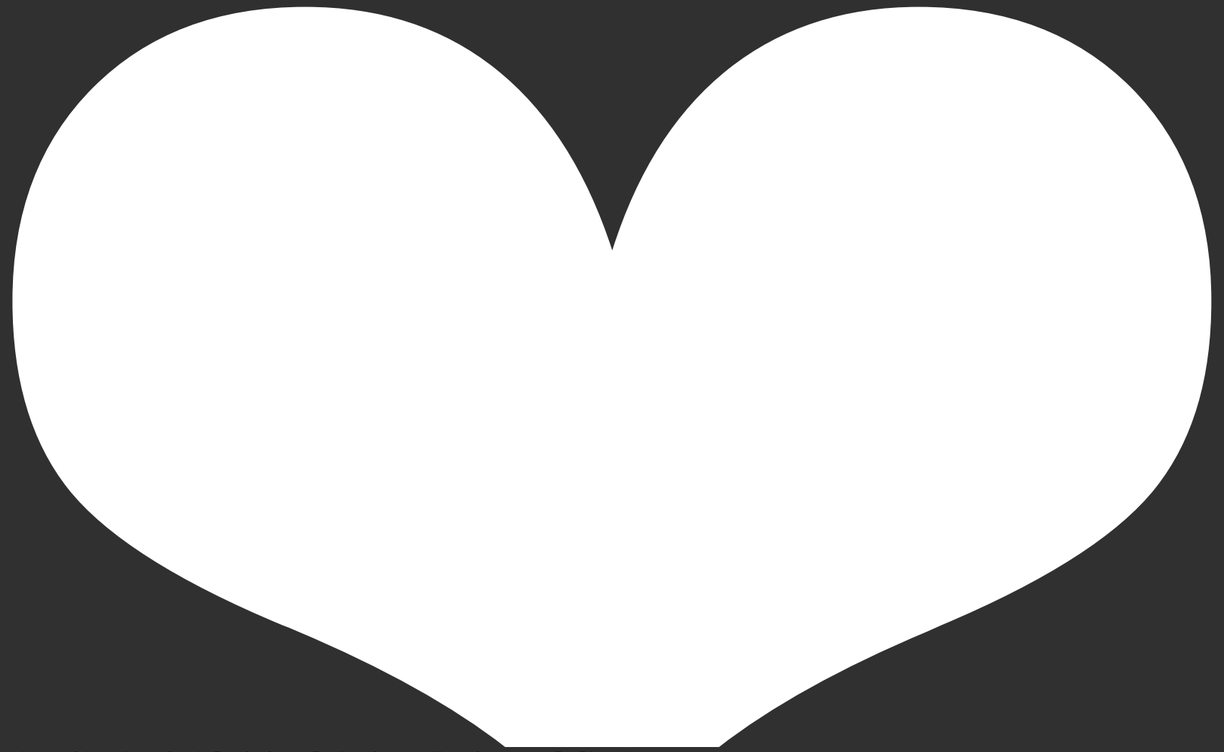
— Help me improve the Hadoop Chef Cookbook

Hadoop Operations

Other Hadoop Ops folks:

```
Allen Wittenauer @_a__w_
Charles Wimmer @cwimmer
Travis Thompson @_trthomps_
Rajive @rajivec
Travis @hcoyote
Laurie Denness @lozzd
```

Thank You!



Questions?

References

Books

— Hadoop Operations by Eric Sammer

— Hadoop the Definitive Guide by Tom White

References

Conferences

- Hadoop Summit
  - Developer focused
- Strata
  - Data Scientist focused

References

Presentations

Top 10 things to get the most out of your Hadoop Cluster
Apache Hadoop for System Administrators- Allen Wittenauer

References

Websites

- Hadoop wiki
- Search Hadoop
- Cloudera Blog
- Hortonworks Blog

Schedulers

BFQ

— Budget Fair Queueing (BFQ) Storage-I/O Scheduler

— BFQ website

— Low latency for interactive applications

— Low latency for soft real-time applications

— High throughput

— Strong fairness guarantees