IBM Spectrum Scale

– Running Hadoop on IBM Spectrum Scale –

Rome – Apr 29, 2016 – Olaf Weiser & Frank Kraemer

olaf.weiser@de.ibm.com

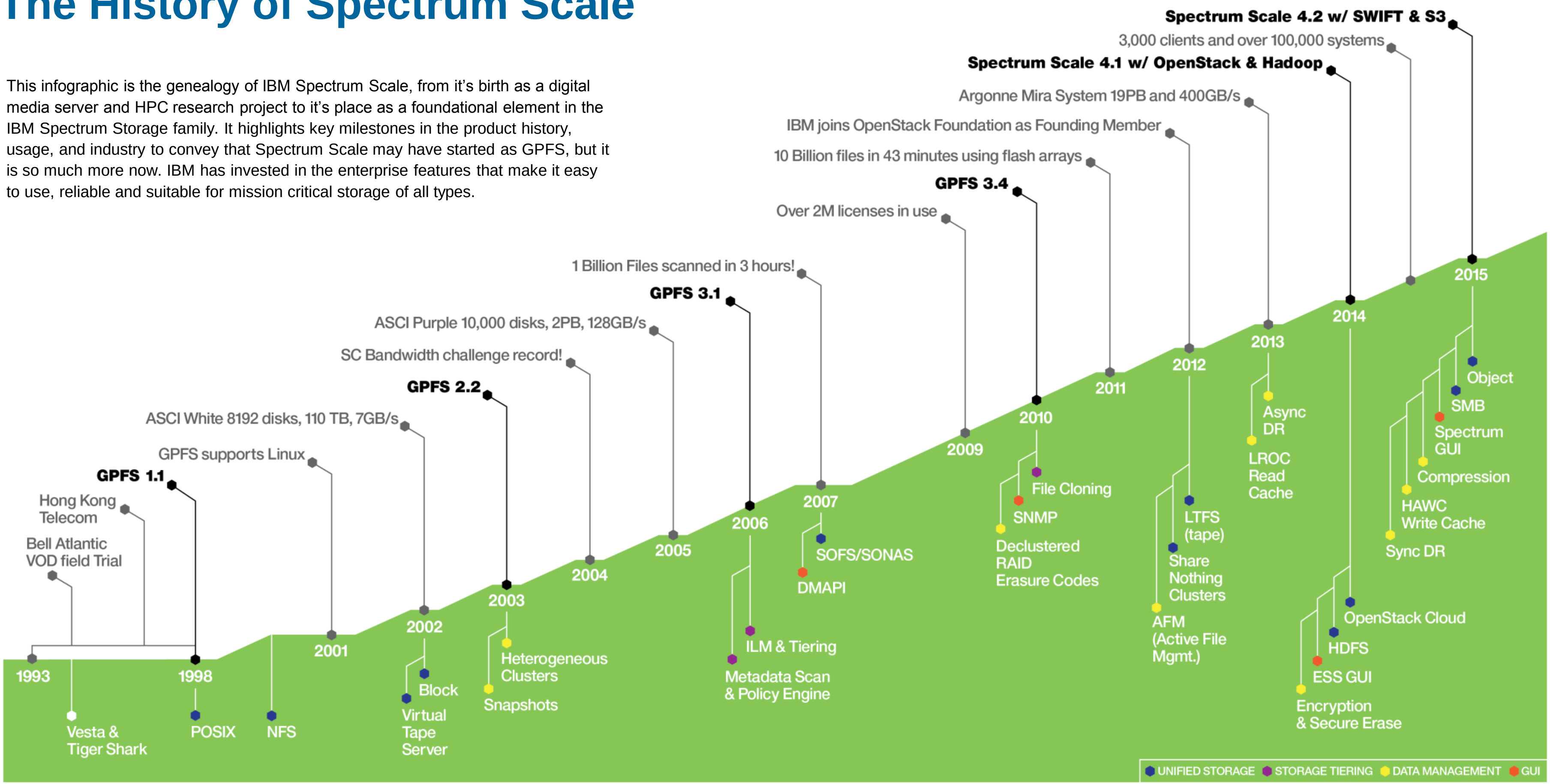
kraemerf@de.ibm.com

Agenda:

- □ Spectrum Scale v4.2
- □ BigInsights
- □ Hadoop Integration
- □ Ambari Integration
- □ Outlook

The History of Spectrum Scale

This infographic is the genealogy of IBM Spectrum Scale, from it’s birth as a digital media server and HPC research project to it’s place as a foundational element in the IBM Spectrum Storage family. It highlights key milestones in the product history, usage, and industry to convey that Spectrum Scale may have started as GPFS, but it is so much more now. IBM has invested in the enterprise features that make it easy to use, reliable and suitable for mission critical storage of all types.



1993
Bell Atlantic VOD field Trial
Hong Kong Telecom
Vesta & Tiger Shark

1998
GPFS 1.1
POSIX

NFS

2001
GPFS supports Linux

2002
ASCI White 8192 disks, 110 TB, 7GB/s
Block
Virtual Tape Server

2003
GPFS 2.2
Heterogeneous Clusters
Snapshots

2004
SC Bandwidth challenge record!

2005
ASCI Purple 10,000 disks, 2PB, 128GB/s

2006
GPFS 3.1
ILM & Tiering
Metadata Scan & Policy Engine

2007
1 Billion Files scanned in 3 hours!
SOFS/SONAS
DMAPI

2009
Over 2M licenses in use

2010
GPFS 3.4
File Cloning
SNMP
Declustered RAID Erasure Codes

2011
10 Billion files in 43 minutes using flash arrays

2012
IBM joins OpenStack Foundation as Founding Member
LTFS (tape)
Share Nothing Clusters
AFM (Active File Mgmt.)

2013
Argonne Mira System 19PB and 400GB/s
Async DR
LROC Read Cache

2014
Spectrum Scale 4.1 w/ OpenStack & Hadoop
OpenStack Cloud
HDFS
ESS GUI
Encryption & Secure Erase

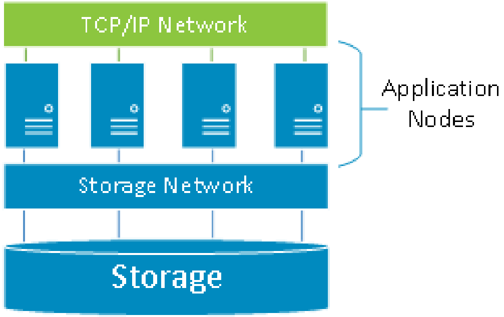
3,000 clients and over 100,000 systems

2015
Spectrum Scale 4.2 w/ SWIFT & S3
Object
SMB
Spectrum GUI
Compression
HAWC Write Cache
Sync DR

● UNIFIED STORAGE ● STORAGE TIERING ● DATA MANAGEMENT ● GUI

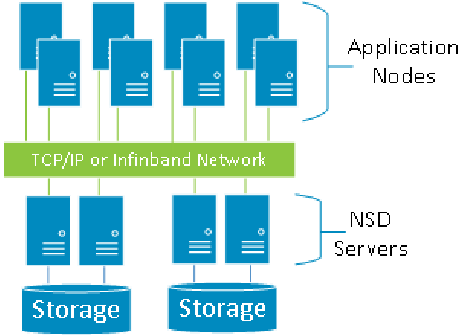
Spectrum Scale deployment models

Enterprise Integrated Model (SAN)



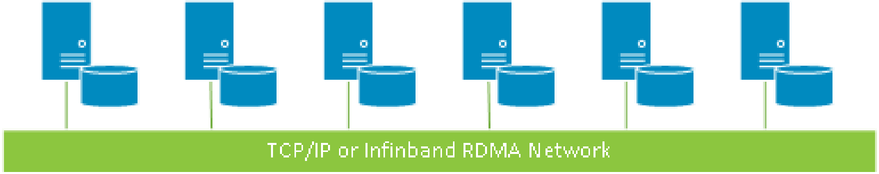
Unify and parallelize storage silos

Network Shared Disk (NSD) Model



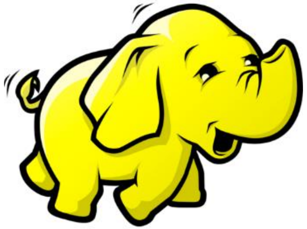
Modular High-Performance Scaling

Shared Nothing Cluster (SNC) Model called FPO

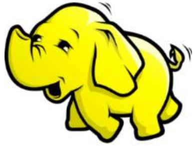


Span storage rich servers for converged architecture or HDFS deployment

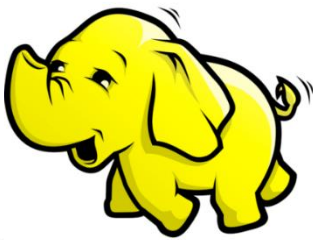
What is HDFS?

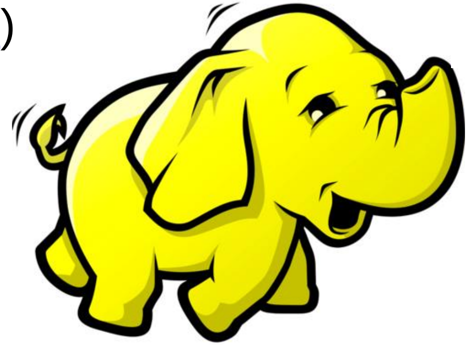


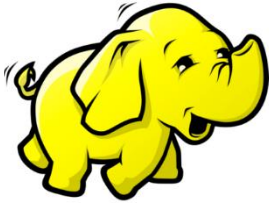
The **Hadoop Distributed File System** **(HDFS)** is a distributed, scalable, and portable file-system written in **Java** for the Hadoop framework.

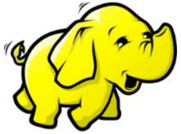


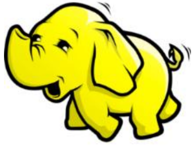
File access can be achieved through the native Java API, the Thrift API to generate a client in the language of the users' choosing (C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, Smalltalk, and OCaml), the command-line interface, or browsed through the HDFS-UI webapp over HTTP. (http://en.wikipedia.org/wiki/Apache_Hadoop)

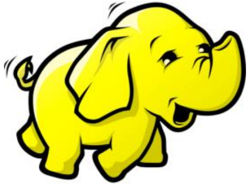












Store everywhere. Run anywhere.

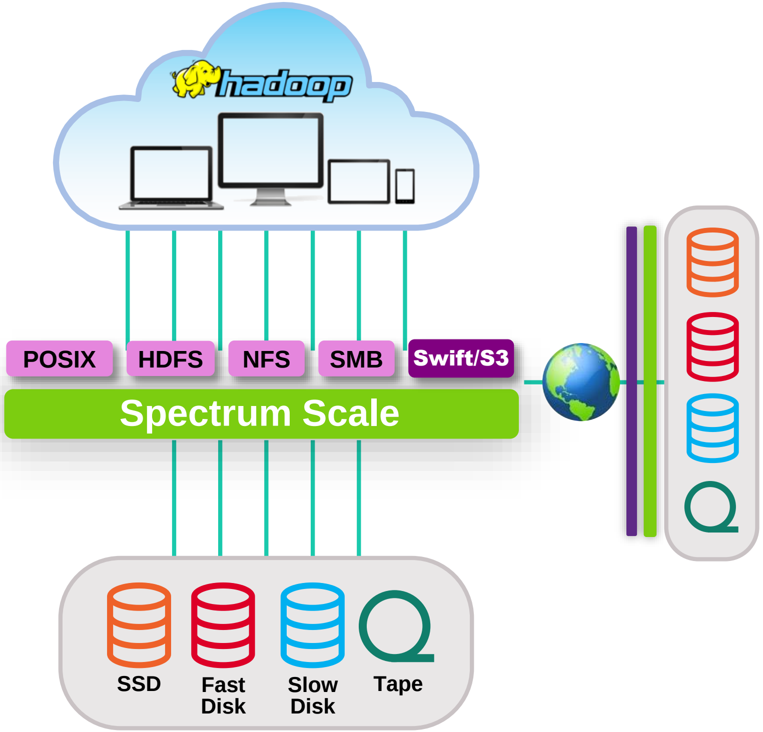
Remove data-related bottlenecks

Challenge

- Managing data growth
  - Lowering data costs
  - Managing data retrieval & app support
  - Protecting business data

Unified Scale-out Data Lake

- File In/Out, Object In/Out; Analytics on demand.
- High-performance native protocols
- Single Management Plane
- Cluster replication & global namespace
- Enterprise storage features across file, object & HDFS



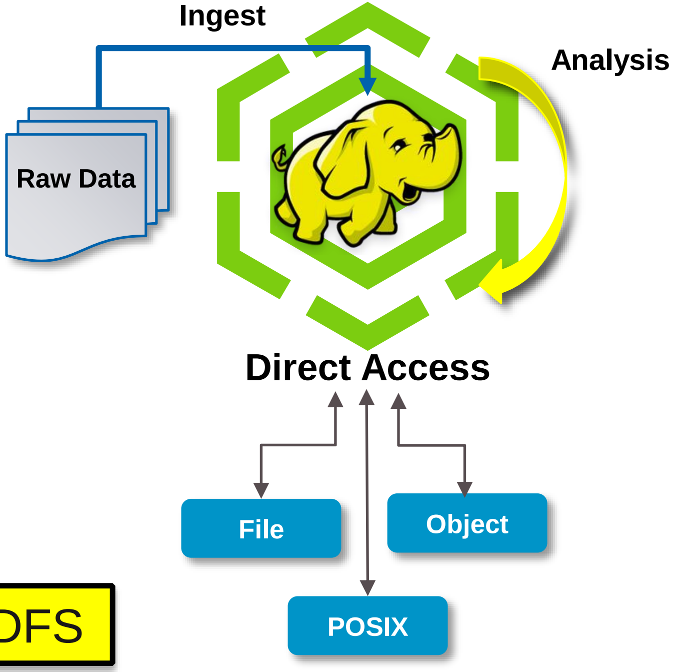
Challenge

Separate storage systems for ingest, analysis, results

- HDFS requires locality aware storage (namenode)
- Data transfer slows time to results
- Different frameworks & analytics tools use data differently

HDFS Transparency

- Map/Reduce on shared, or shared nothing storage
- No waiting for data transfer between storage systems
- Immediately share results
- Single ‘Data Lake’ for all applications
- Enterprise data management
- Archive and Analysis in-place



Analyze object and file data without copying into HDFS

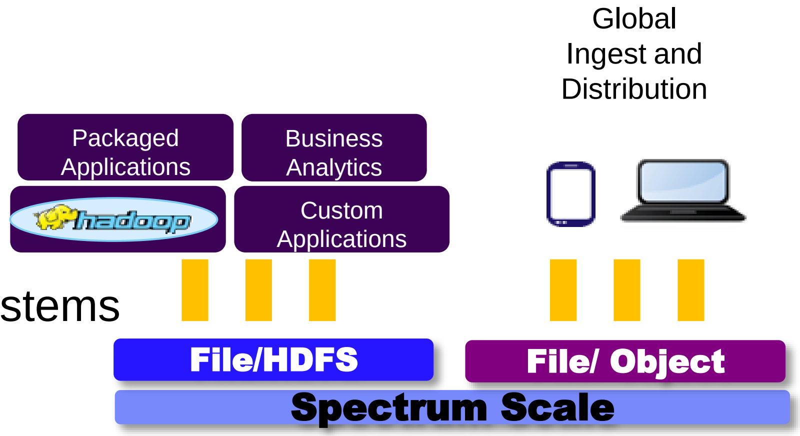
Use Case: Big Data Analytics

Problem: Separate storage systems for ingest/distribution and analysis

- Data movement overhead is a significant part of my time to insight**.**
- Increased cost from data duplication & overhead
- Inconsistent results

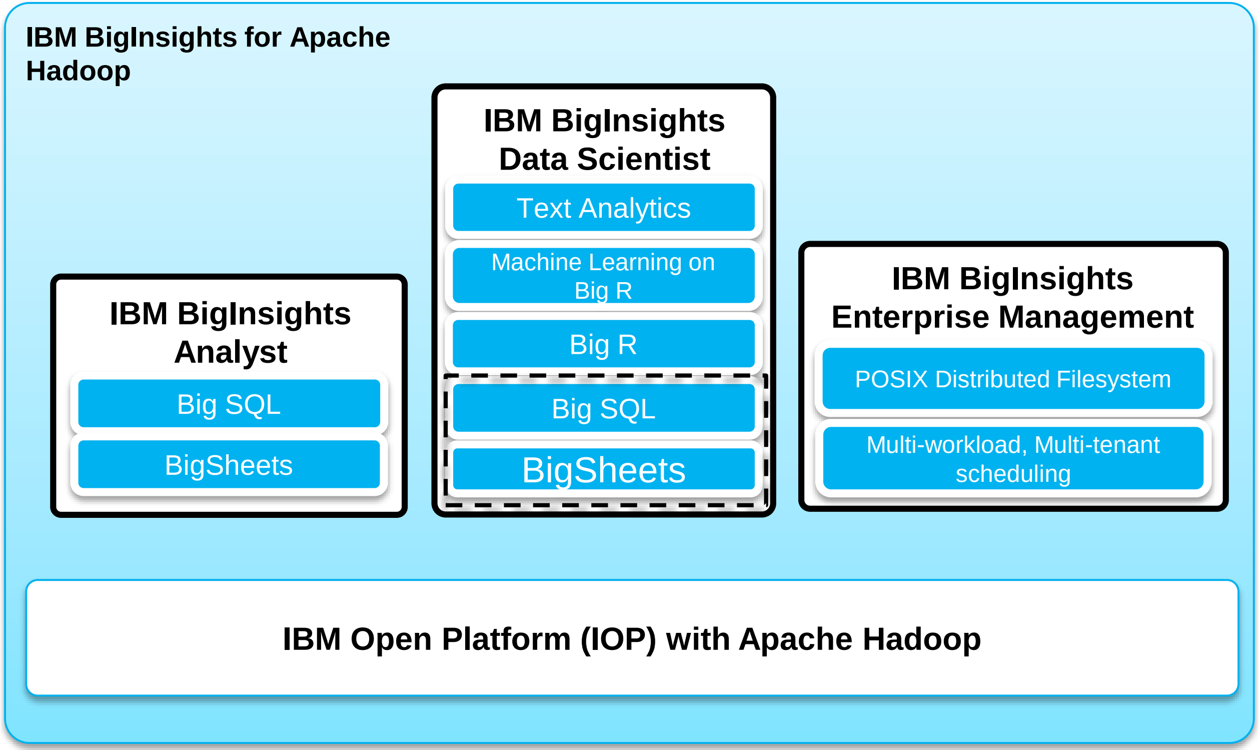
Solution: Native HDFS support

- Decreased time to results
- Run Map/Reduce directly
- No waiting for data transfer between storage systems
- Immediately share results



IBM Open Platform (IOP) with Apache Hadoop

IBM BigInsights



IBM BigInsights for Apache Hadoop

IBM BigInsights Analyst

- Big SQL
- BigSheets

IBM BigInsights Data Scientist

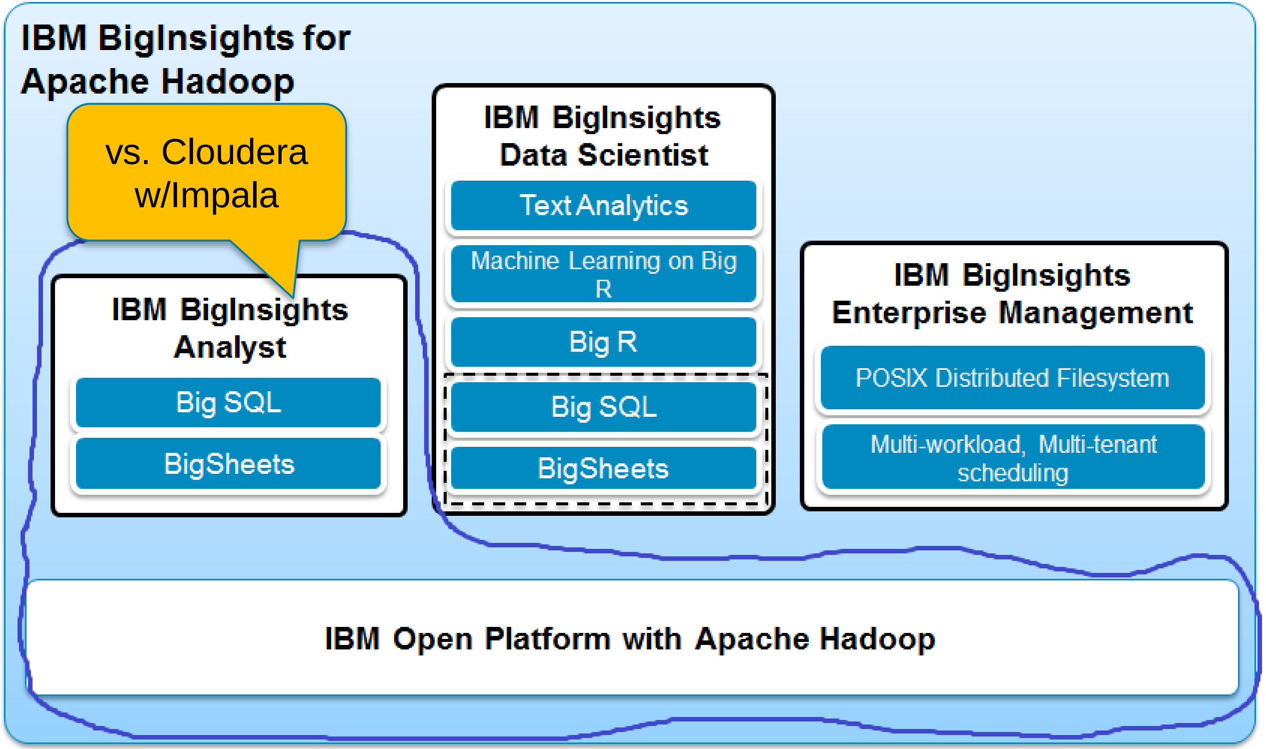
- Text Analytics
- Machine Learning on Big R
- Big R
- Big SQL
- BigSheets

IBM BigInsights Enterprise Management

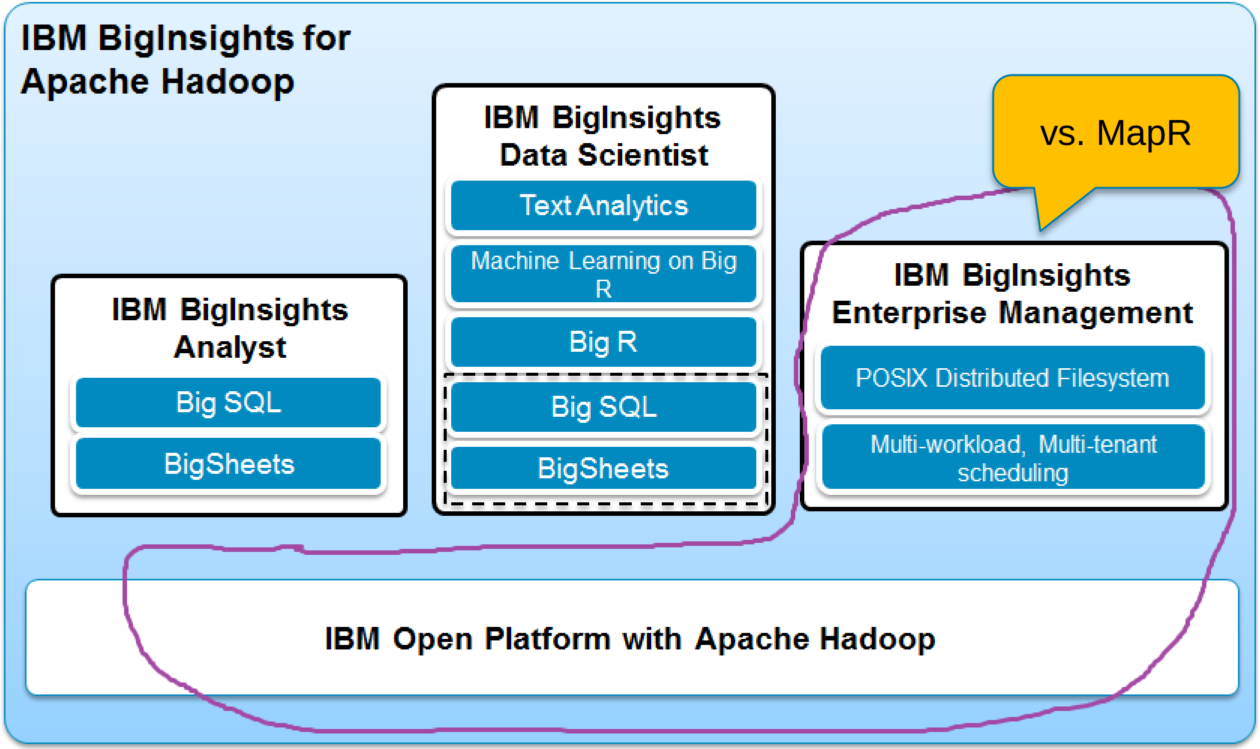
- POSIX Distributed Filesystem
- Multi-workload, Multi-tenant scheduling

IBM Open Platform (IOP) with Apache Hadoop

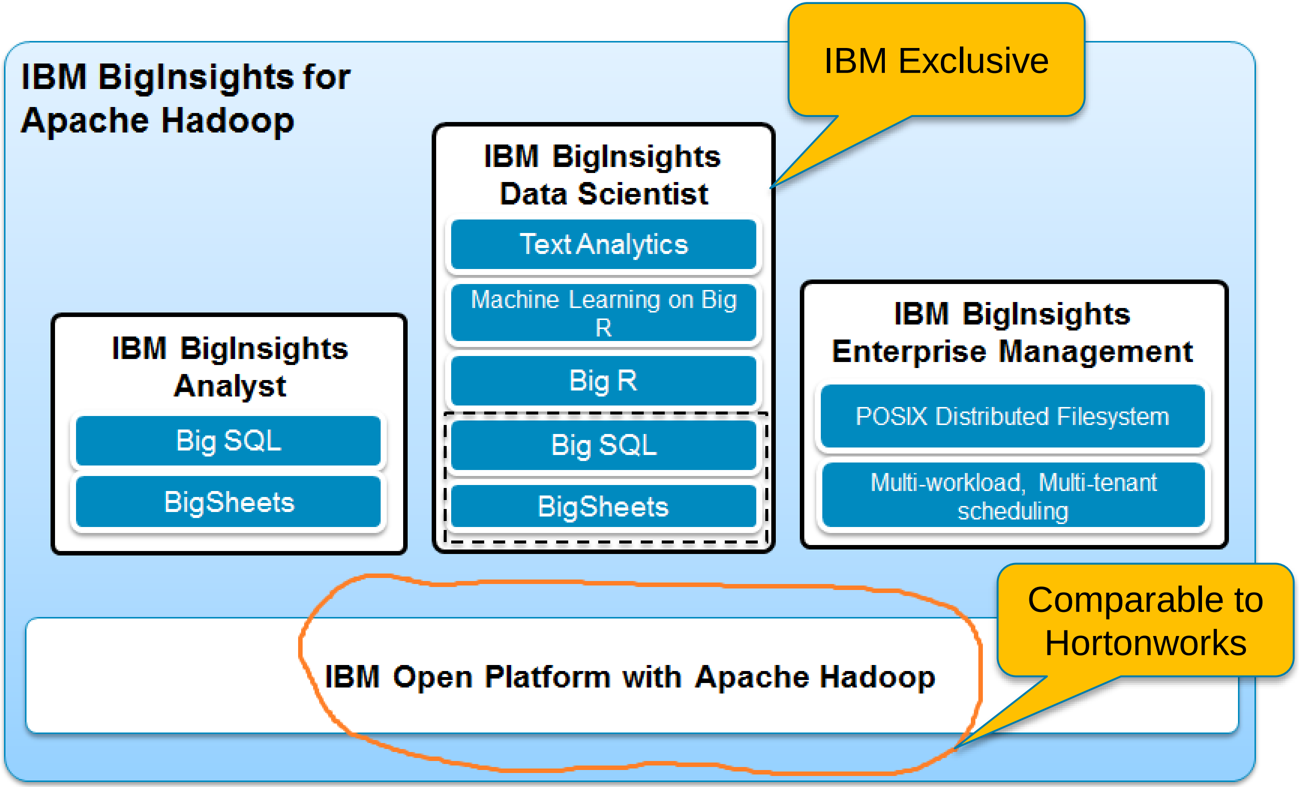
Comparing to Cloudera


IBM BigInsights for Apache Hadoop
vs. Cloudera w/Impala
IBM BigInsights Analyst
Big SQL
BigSheets
IBM BigInsights Data Scientist
Text Analytics
Machine Learning on Big R
Big R
Big SQL
BigSheets
IBM BigInsights Enterprise Management
POSIX Distributed Filesystem
Multi-workload, Multi-tenant scheduling
IBM Open Platform with Apache Hadoop

Comparing to Map R (with POSIX File System)


IBM BigInsights for Apache Hadoop
IBM BigInsights Analyst
Big SQL
BigSheets
IBM BigInsights Data Scientist
Text Analytics
Machine Learning on Big R
Big R
Big SQL
BigSheets
IBM BigInsights Enterprise Management
POSIX Distributed Filesystem
Multi-workload, Multi-tenant scheduling
vs. MapR
IBM Open Platform with Apache Hadoop

IBM Open Platform (alone) is similar to Hortonworks



IBM BigInsights for Apache Hadoop

IBM BigInsights Analyst
- Big SQL
- BigSheets

IBM BigInsights Data Scientist
- Text Analytics
- Machine Learning on Big R
- Big R
- Big SQL
- BigSheets

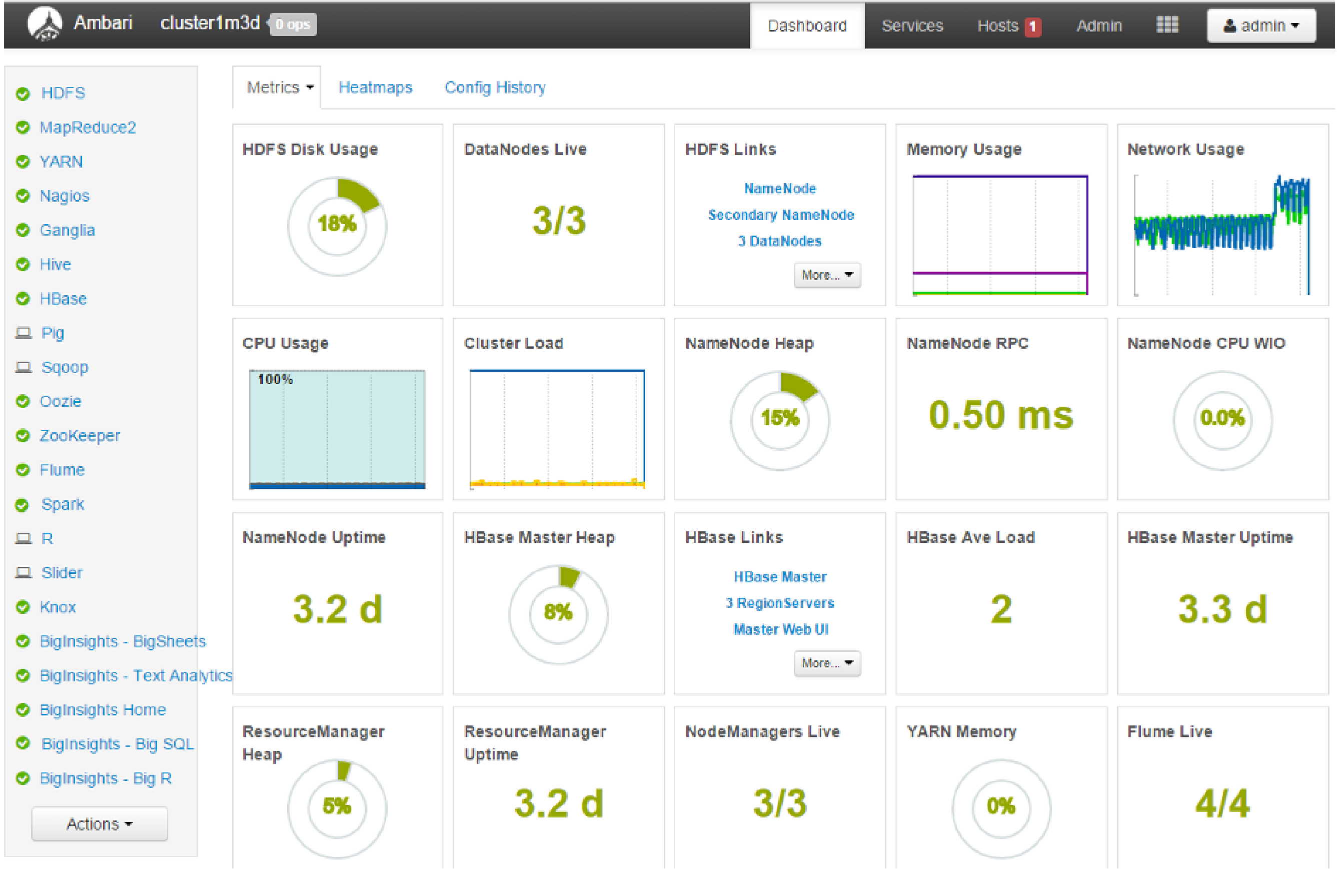
IBM BigInsights Enterprise Management
- POSIX Distributed Filesystem
- Multi-workload, Multi-tenant scheduling

IBM Open Platform with Apache Hadoop

IBM Exclusive

Comparable to Hortonworks

This is Hortonworks…. Oh wait,.. no, it's BigInsights!



Ambari cluster1m3d 0 ops

Dashboard | Services | Hosts 1 | Admin | admin

- HDFS
- MapReduce2
- YARN
- Nagios
- Ganglia
- Hive
- HBase
- Pig
- Sqoop
- Oozie
- ZooKeeper
- Flume
- Spark
- R
- Slider
- Knox
- BigInsights - BigSheets
- BigInsights - Text Analytics
- BigInsights Home
- BigInsights - Big SQL
- BigInsights - Big R

Actions

Metrics | Heatmaps | Config History

HDFS Disk Usage 18%	DataNodes Live 3/3	HDFS Links NameNode Secondary NameNode 3 DataNodes More...	Memory Usage	Network Usage
CPU Usage 100%	Cluster Load	NameNode Heap 15%	NameNode RPC 0.50 ms	NameNode CPU WIO 0.0%
NameNode Uptime 3.2 d	HBase Master Heap 8%	HBase Links HBase Master 3 RegionServers Master Web UI More...	HBase Ave Load 2	HBase Master Uptime 3.3 d
ResourceManager Heap 5%	ResourceManager Uptime 3.2 d	NodeManagers Live 3/3	YARN Memory 0%	Flume Live 4/4

IBM Open Platform (IOP) as of V4.1

Open Data Platform (ODP) benefits and IBM open source project currency commitment

Component	IBM Open Platform V4.1	Hortonworks HDP 2.3	Cloudera CDH 5.4.7
Ambari	2.1.0	2.1.0	N/A
Flume	1.5.2	1.5.2	1.5.0
Hadoop / YARN	2.7.1	2.7.1	2.6.0
Hbase	1.1.1	1.1.1	1.0.0
Hive	1.2.1	1.2.1	1.1.0
Kafka	0.8.2.1	0.8.2	0.8.2
Knox	0.6.0	0.6.0	N/A
Oozie	4.2.1	4.2	4.1.0
Pig	0.15.0	0.15.0	0.12.0
Slider	0.80.0	0.8.0	N/A
Solr	5.1.0	5.2.1	4.10.3
Spark	1.4.1	1.3.1	1.3.0
Sqoop	1.4.6	1.4.6	1.4.5
Zookeeper	3.4.6	3.4.6	3.4.5

IBM Open Platform (IOP) V4.2 (Target GA - 2Q 2016)

Component	IBM Open Platform V4.2 (2Q 2016)	Hortonworks HDP 2.4
Ambari	**2.2**	2.2.1
Flume	**1.6**	1.5.2
Hadoop / YARN, MR, HDFS	**2.7.2**	2.7.1
Hbase	**1.2**	1.1.2
Hive	1.2.1	1.2.1
Kafka	**0.9.0**	0.9.0
Knox	**0.7**	0.6.0
Oozie	4.2	4.2
Pig	0.15.0	0.15.0
Slider	**0.90.2**	0.80
Solr	**5.4**	5.2.1
Spark	**1.6.0**	1.6.0
Sqoop	1.4.6	1.4.6
Zookeeper	3.4.6	3.4.6

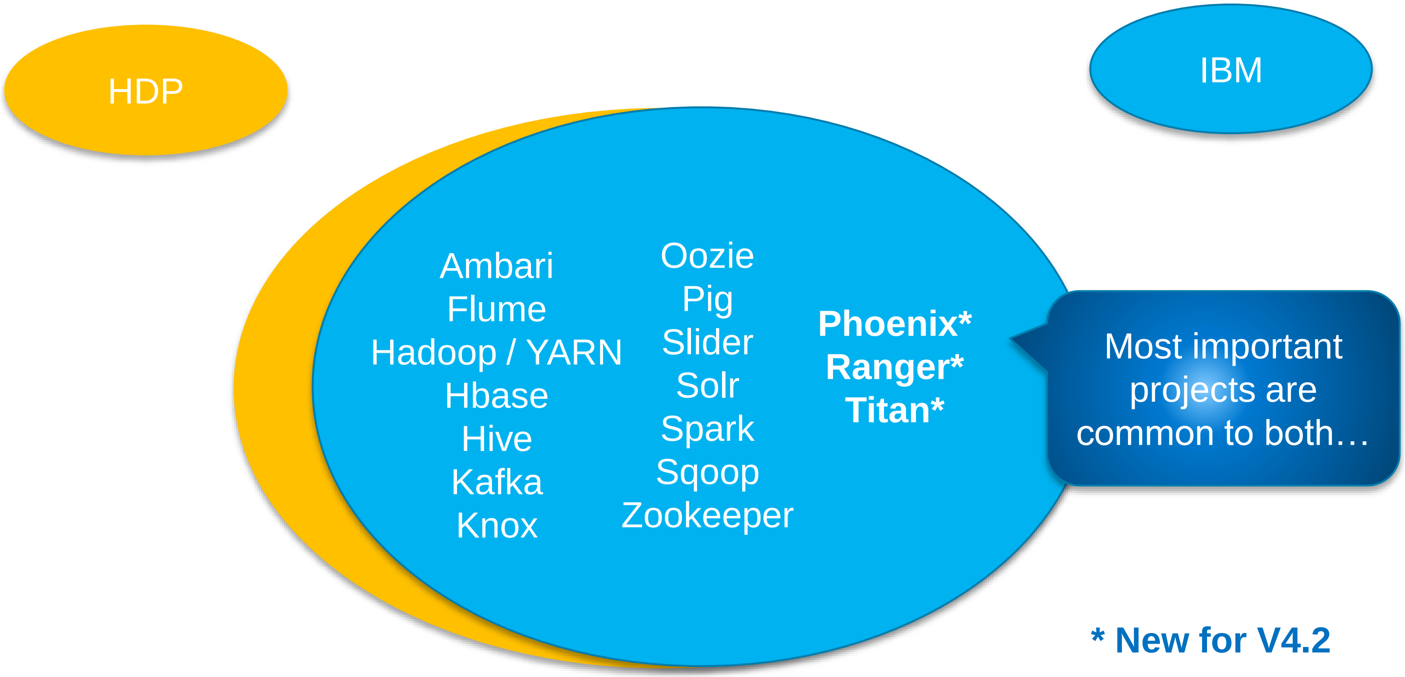
New Additions to IOP:

Phoenix (4.6.1)
Ranger (0.5)
Titan (1.0)
SystemML (0.9)

(bold) = updated

IBM Open Platform vNext (2Q 2016)

Hortonworks and IBM Open Platform have a lot in common…..



*** New for V4.2**

Projects to be supported through Developerworks / Forums:

Hue (v3.9), Storm (v0.10.0), Accumulo (1.7.0), Mahout (0.11), Tez (0.8.2)

Spectrum Scale Hadoop Integration

A Tale of Two Connectors

GPFS Hadoop Connector

- Henceforth known as the “old” connector
- Emulates a Hadoop compatible filesystem
- Replaces HDFS
- Stateless
- Free download – link
- Supports Spectrum Scale 4.1.x, 4.1.1.x and 4.2
- Currently supported with IOP 4.0.x and 4.1.x

Spectrum Scale HDFS Transparency Connector

- Henceforth known as the “new” connector
- Integrates with HDFS – reuses HDFS client and implements NameNode and DataNode RPCs
- Stateless
- Free download – link
- Supports Spectrum Scale 4.1.x, 4.1.1.x and 4.2
- Planned for IOP 4.2

Old GPFS Hadoop Connector Approach

How can we be sure we’re compatible?

Hadoop File System API intended to be open.

```
public abstract class
 org.apache.hadoop.fs.FileSystem
```

Source: *hadoop.apache.org*

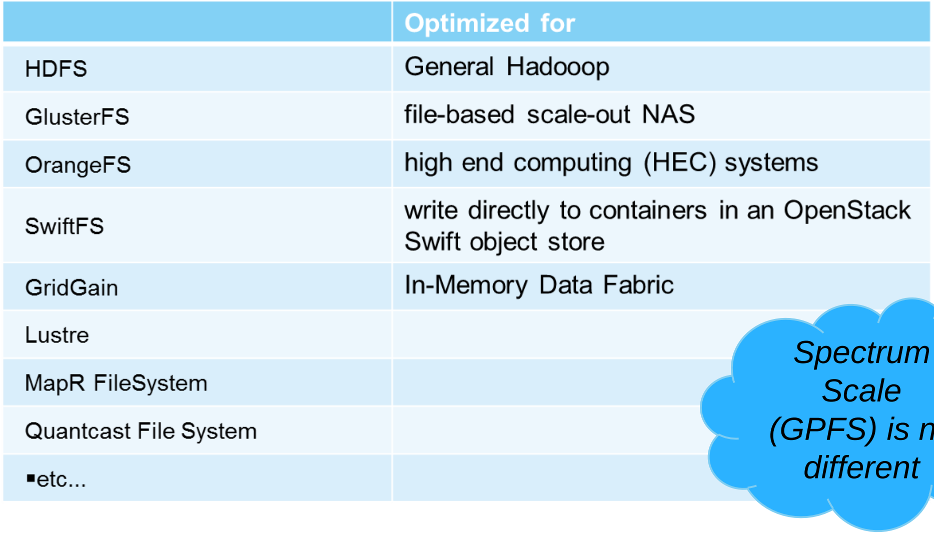
> ***“All user code that may potentially use the Hadoop Distributed File System should be written to use a FileSystem object.”***

Latest File System APIs are described here:

https://hadoop.apache.org/docs/current/api/org/apache/hadoop/fs/FileSystem.html

Old GPFS Hadoop Connector Approach

All based on org.apache.hadoop.fs.FileSystem API



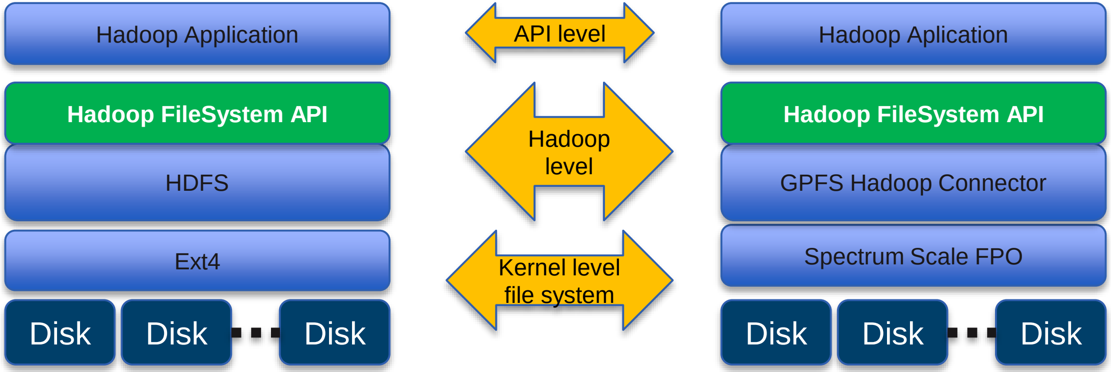
	Optimized for
HDFS	General Hadooop
GlusterFS	file-based scale-out NAS
OrangeFS	high end computing (HEC) systems
SwiftFS	write directly to containers in an OpenStack Swift object store
GridGain	In-Memory Data Fabric
Lustre	
MapR FileSystem	
Quantcast File System	
▪etc...	

Spectrum Scale (GPFS) is no different

Source: https://wiki.apache.org/hadoop/HCFS

Old GPFS Hadoop Connector Approach

Applications communicate with Hadoop using FileSystem API. Therefore, transparency is preserved.



"All user code that may potentially use the Hadoop Distributed File System should be written to use a ***FileSystem*** *object."*

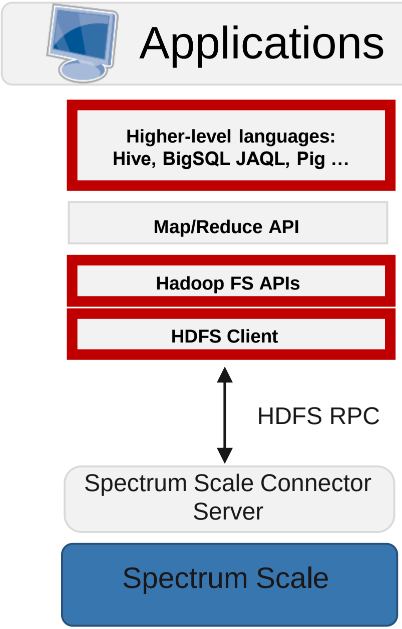
Source: *hadoop.apache.org*

New Spectrum Scale HDFS Transparency Design

- Issues with old Hadoop Connector
  - Some applications and many tools do not use **org.apache.hadoop.fs.FileSystem**
  - Those applications and tools fail with (old) HDFS Connector

- Key Advantages of new **HDFS Transparency Connector**
  - Support workloads that have hard coded HDFS dependencies
  - Simpler integration for currently compatible workloads & components
  - Leverage HDFS Client cache for better performance
  - No need to install Spectrum Scale clients on all nodes
  - Full Kerberos support for Hadoop ecosystem

New Spectrum Scale HDFS Transparency Design



Applications

Higher-level languages: Hive, BigSQL JAQL, Pig …

Map/Reduce API

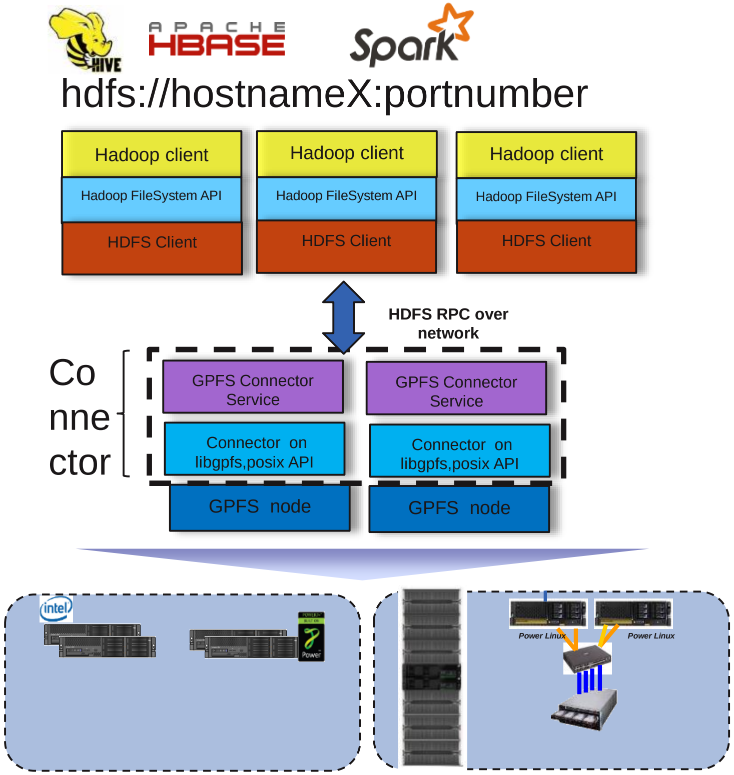
Hadoop FS APIs

HDFS Client

HDFS RPC

Spectrum Scale Connector Server

Spectrum Scale



hdfs://hostnameX:portnumber

Hadoop client	Hadoop client	Hadoop client
Hadoop FileSystem API	Hadoop FileSystem API	Hadoop FileSystem API
HDFS Client	HDFS Client	HDFS Client

HDFS RPC over network

Connector

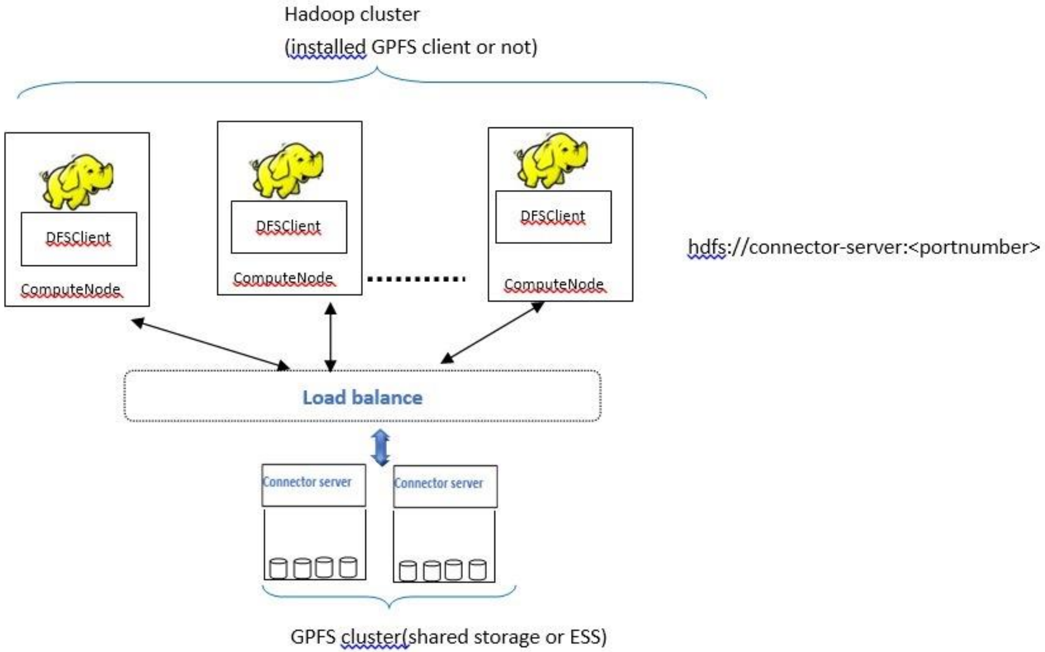
GPFS Connector Service	GPFS Connector Service
Connector on libgpfs,posix API	Connector on libgpfs,posix API
GPFS node	GPFS node

Commodity hardware

Shared storage

New Spectrum Scale HDFS Transparency Design

- Connector servers are installed over limited nodes (ex. GPFS NSD servers)
- GPFS client is not needed over the Hadoop computing nodes
- DNS rotation or CES can be used to load balance for HDFS Client



HDFS Transparency – configuration overview

HDFS Transparency – configuration overview

Having a Hadoop / IOP cluster

- ✓ stop IOP,
- ✓ optionally : recycle hdfs

Having an up n running GPFS cluster / filesystem

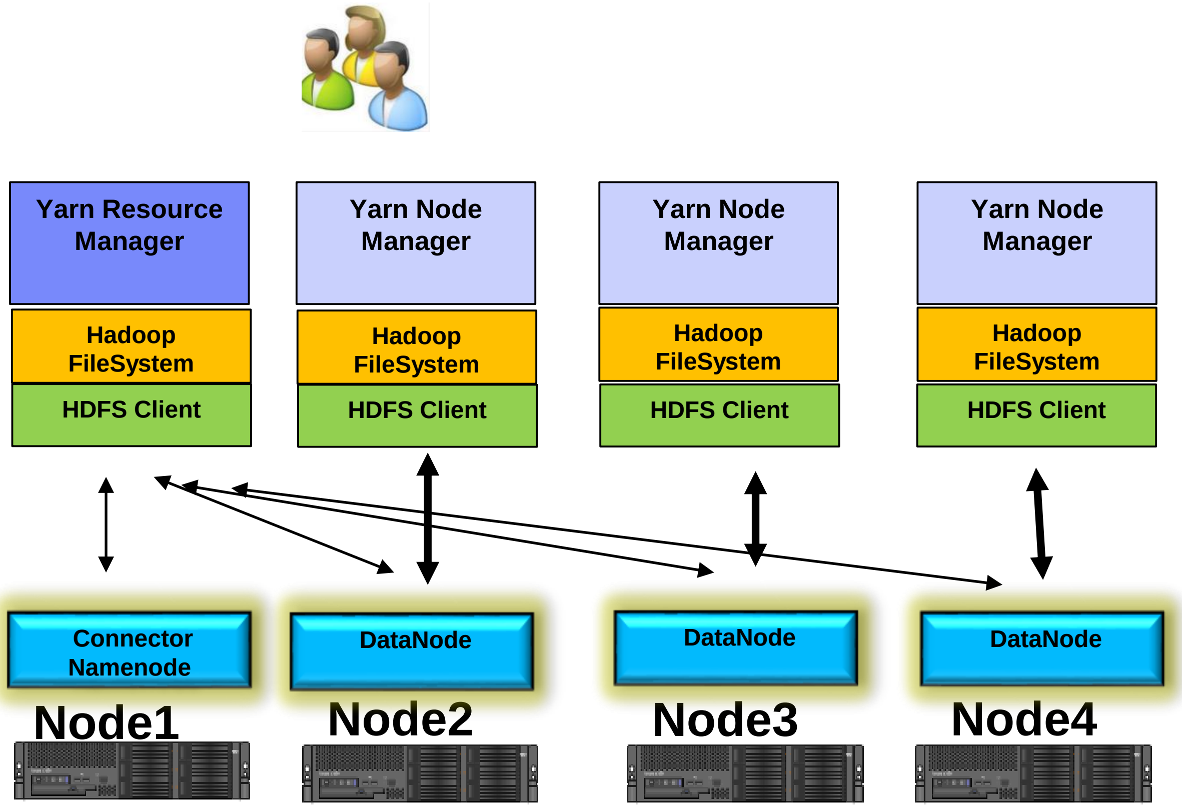
Customize in Ambari GUI / directly

- ✓ hdfs-site.xml
- ✓ core-site.xml

- install **gpfs.hdfs-protocol-2.7.0-0.<arch>.rpm**
- /usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf <your hadoop config dir>
- cd /usr/lpp/mmfs/hadoop/etc/hadoop;
  cp gpfs-site.xml.template gpfs-site.xml
- Modify ***gpfs-site.xml*** according mount point and data directory
- configure slaves
- /usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector start|stop

Note: connector logs are stored under **/usr/lpp/mmfs/hadoop/logs/.**

Hadoop Job Execution with gpfs.hdfs-protocol



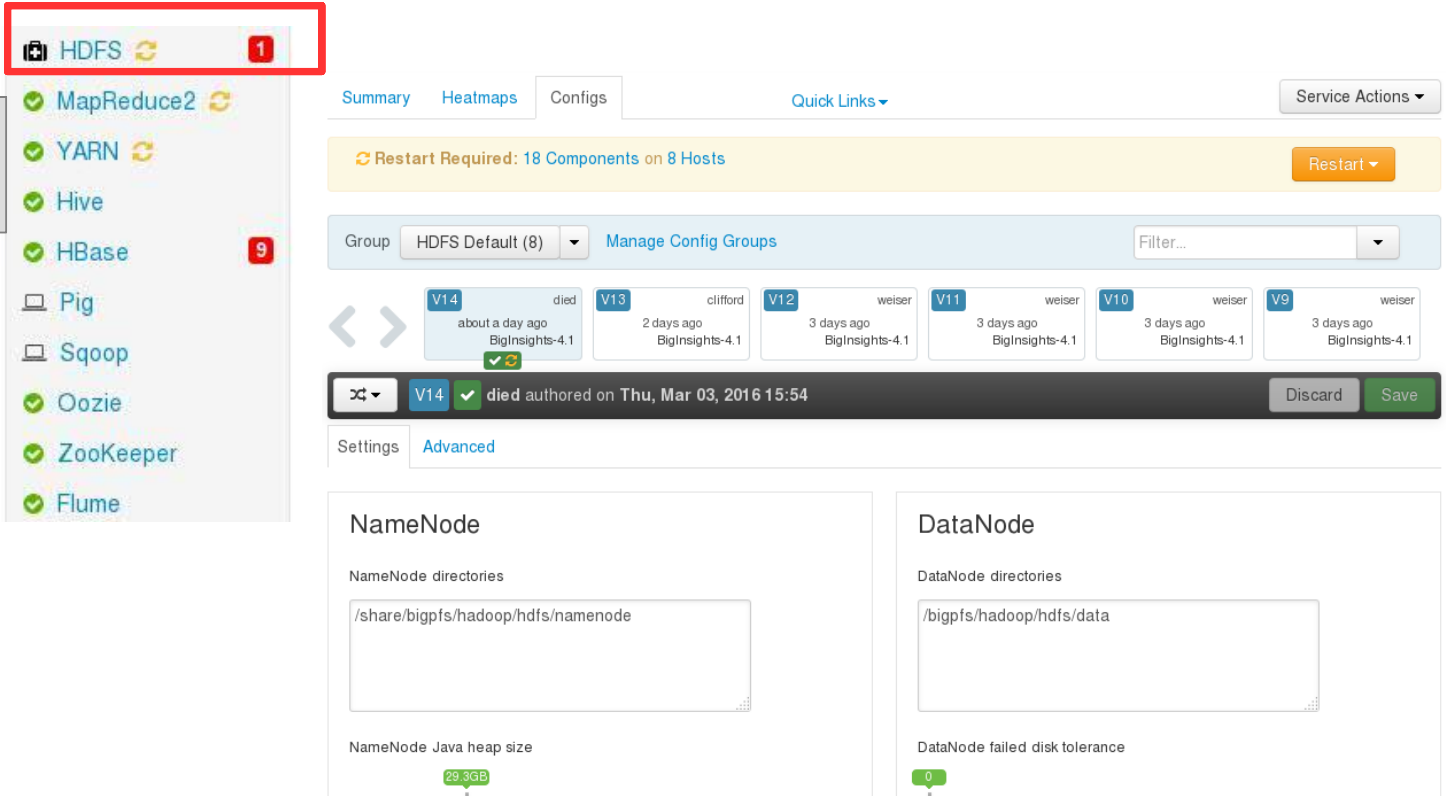
- ○ Configure (connector) name node or nameNodeHA
- ○ Configure data nodes (regular GPFS clients, sharing the same config gpfs-site.xml)
- ○ Syncronize config
- ○ Start connector
- ○ Start applications

HDFS Transparency – configuration overview

Apache Ambari

Current IOP:

Set to maintenance mode



Restart Required: 18 Components on 8 Hosts

HDFS Default (8)

V14	V13	V12	V11	V10	V9
died	clifford	weiser	weiser	weiser	weiser
about a day ago	2 days ago	3 days ago	3 days ago	3 days ago	3 days ago
BigInsights-4.1	BigInsights-4.1	BigInsights-4.1	BigInsights-4.1	BigInsights-4.1	BigInsights-4.1

V14 died authored on Thu, Mar 03, 2016 15:54

NameNode

NameNode directories: /share/bigpfs/hadoop/hdfs/namenode

NameNode Java heap size: 29.3GB

DataNode

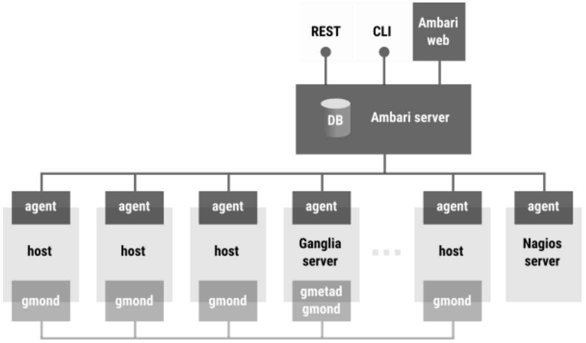
DataNode directories: /bigpfs/hadoop/hdfs/data

DataNode failed disk tolerance: 0

Using Ambari to Deploy and Manage Spectrum Scale

Apache Ambari

- Apache Ambari is an open source tool for deploying and managing a Hadoop cluster.
- Ambari has been developed primarily by Hortonworks, another core member of the Open Data Platform Initiative along with IBM and others.
- Ambari provides a step-by-step install wizard for the initial installation and configuration of Hadoop services across the cluster.
- Ambari provides central management for starting, stopping and reconfiguring Hadoop services.



https://ambari.apache.org/

Apache Ambari (cont.)

- In addition to a web-based user interface, Ambari also provides a RESTful API
- Ambari provides a framework for defining custom services that can also be installed and managed through Ambari's GUI and RESTful API.
- This framework is used to integrate Spectrum Scale, Platform Symphony and other value-add services from BigInsights (such as Big SQL or Big R) into a single interface for installing, configuring and managing services
- Ambari does NOT handle OS provisioning, and is not a replacement for Platform Cluster Manager, xCAT, or similar cluster management tools

https://ambari.apache.org/

Automated Installation of Spectrum Scale

Apache Ambari

Ambari can:

- Install Spectrum Scale across a cluster of any size with minimal input from the administrator.
- Create a new cluster using the File Placement Optimizer (FPO) feature, utilizing the local disks on the Hadoop nodes themselves.
- Install Spectrum Scale on a set of nodes and join an existing shared cluster (such as an ESS-hosted cluster).
- Utilize an existing Spectrum Scale cluster for Hadoop services.
- Optimize Spectrum Scale's configuration for use by Hadoop services.
- Optimize the configurations of the Hadoop services for using Spectrum Scale rather than the Hadoop Distributed File System (HDFS).

https://ambari.apache.org/

Spectrum Scale as an Ambari Service

- A definition for the Spectrum Scale service is installed at the time the Ambari server is installed.
- When the administrator runs the Ambari install wizard, Spectrum Scale is selected, configured and installed just like any of the other Hadoop services.
- The HDFS service has been removed as an option when Spectrum Scale is being used.
- After installation, Spectrum Scale is another service managed by Ambari and can be stopped and started from the UI.
- Configuration changes can be made after the cluster has been started.

https://ambari.apache.org/

Highlights of the Available Features

- Ambari can create a new Spectrum Scale cluster and filesystem through the install wizard.
- A set of configuration values will be used for the Spectrum Scale cluster to optimize use for a Hadoop cluster.
- Configuration values for other Hadoop services will be set to optimize their use of the Spectrum Scale filesystem.
- The Ambari installer will also provide the option to use an existing Spectrum Scale cluster and filesystem and install the Hadoop Connector if it is not already installed.
- Any number of filesystem configurations can be supported from ESS to a filesystem shared with other clusters running other workloads to an existing cluster dedicated to use by the Hadoop services.

https://ambari.apache.org/

Highlights of the Available Features (cont.)

- Spectrum Scale daemons and the Hadoop Connector can be stopped and started on some or all of the nodes in the cluster from Ambari.
- Ambari can provide some basic information about whether or not Spectrum Scale services are available on a given node in the cluster.
- Configuration changes can be made to the Spectrum Scale cluster from Ambari using Ambari's configuration version control feature.
- Nodes can be added to the cluster after the initial deployment.
- Nodes can be removed from the cluster from the Ambari GUI.
- Snap data can be collected from a menu option inside Ambari.
- Spectrum Scale can be upgraded.

https://ambari.apache.org/

Architecture of the Spectrum Scale service

An Ambari cluster consists of a “stack” of services, defined by files found in **/var/lib/ambari-server/resources/stacks**

Ambari stacks support inheritance, so one stack can extend another:

```
[root@dn01-dat ~]# ls -l /var/lib/ambari-server/resources/stacks/BigInsights/
total 12
drwxr-xr-x 8 root root 4096 Oct 27 13:20 4.0
drwxr-xr-x 5 root root 4096 Oct 28 11:38 4.1
drwxr-xr-x 5 root root 4096 Oct 28 11:38 4.1.SpectrumScale
```

Each service is defined in the stack:

```
[root@dn01-dat ~]# ls -l /var/lib/ambari-server/resources/stacks\
/BigInsights/4.1.SpectrumScale/services/
total 72
drwxr-xr-x 2 root root  4096 Oct 28 10:48 FLUME
drwxr-xr-x 5 root root  4096 Oct 29 12:20 GPFS
drwxr-xr-x 3 root root  4096 Oct 28 10:48 HBASE
…
```

Due to stack inheritance, if you can't find the configuration file or script you're looking for in one stack, you may need to look in the parent stack.

https://ambari.apache.org/

Architecture of the Spectrum Scale service (cont.)

Apache Ambari

The service consists of configuration files and python scripts:

```
[root@dn01-dat services]# pwd
/var/lib/ambari-server/resources/stacks/BigInsights/4.1.SpectrumScale/services
[root@dn01-dat services]# ls -l GPFS/configuration/
total 52
-rwxr-xr-x 1 root root 8328 Oct 28 10:44 core-site.xml
-rwxr-xr-x 1 root root 9704 Oct 30 08:48 gpfs-advance.xml
-rwxr-xr-x 1 root root 2473 Oct 28 10:44 gpfs-env.xml
-rwxr-xr-x 1 root root 4668 Oct 29 09:16 gpfs-filesystem.xml
-rwxr-xr-x 1 root root 9483 Oct 28 10:44 hadoop-env.xml
-rwxr-xr-x 1 root root 1419 Oct 28 10:44 hdfs-site.xml

[root@dn01-dat services]# ls -l GPFS/package/scripts/
total 168
-rwxr-xr-x 1 root root 10729 Oct 28 10:44 gpfs-disks-mounts.sh
-rwxr-xr-x 1 root root 40368 Oct 28 10:44 gpfs-disks-partition.sh
-rwxr-xr-x 1 root root 59699 Oct 28 10:44 gpfs.py
-rwxr-xr-x 1 root root 20196 Oct 29 09:41 master.py
-rwxr-xr-x 1 root root  1974 Oct 28 10:44 params.py
-rwxr-xr-x 1 root root  3314 Oct 28 10:44 racks.py
-rwxr-xr-x 1 root root  4607 Oct 28 10:44 service_check.py
-rwxr-xr-x 1 root root  6467 Oct 28 10:44 slave.py
-rwxr-xr-x 1 root root   422 Oct 28 10:44 status_params.py
-rwxr-xr-x 1 root root  4387 Oct 28 10:44 tuning.py
```

https://ambari.apache.org/

Architecture of the Spectrum Scale service (cont.)

- Because the Ambari stack framework and the service scripts are in Python, very detailed debugging is possible.
- The Spectrum Scale service consists of one “Master” component and 2 “Slave” components.
- The GPFS Master service handles most of the installation work (cluster creation, filesystem creation, adding nodes to the cluster, configuration).
- The GPFS Master node is a quorum node by default, but aside from that, it does not have any special role in the GPFS cluster.
- The GPFS Node slave service handles some installation (install rpms, compile portability layer) and starting and stopping GPFS (mmstartup, mmshutdown).
- The GPFS Hadoop Connector service manages the connector daemon separately from the GPFS daemons (mmhadoopctl).

https://ambari.apache.org/

Outlook

- Coming soon
  - BigInsights v4.2 support (plus additional components)
  - HDFS + Spectrum Scale Federation
  - Federate multiple Spectrum Scale clusters
  - Isolate multiple Hadoop clusters on the same filesystem (restrict to sub-directory)

Thank you.
Questions?

ibm.com/systems