

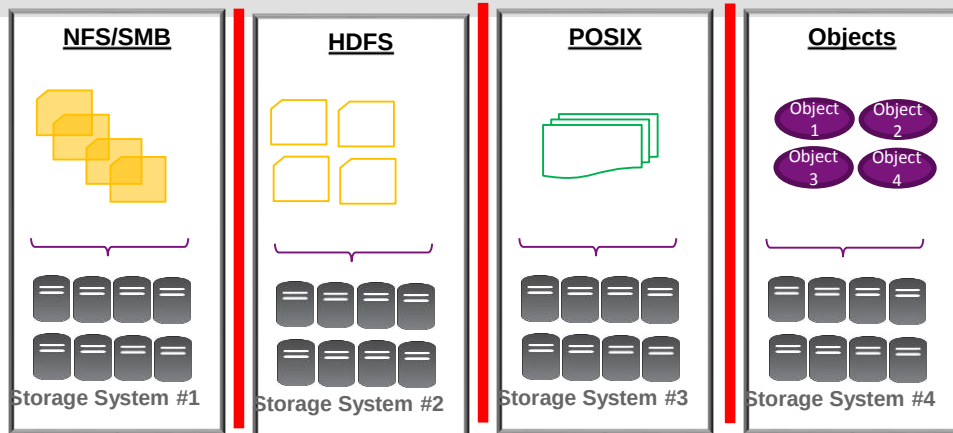
Spinning up a Hadoop Cluster on Demand

Ted Hoover
Program Director

IBM Spectrum Scale Development

Typical Challenges for Customers in Data Analytics

● Inefficient data movements and many copies due to poor interfaces and siloed infrastructure



- ❖ Different analytics workloads might need different interface
 - ✓ Traditional workloads (such as data warehouse, HPC etc) requires POSIX interface
 - ✓ Data ingestion clients need NFS/SMB interface
 - ✓ map/reduce jobs need HDFS interface
- ❖ Poor data accessing interfaces from storage system make customers have to set up siloed infrastructure
- ❖ Multiple data sources, not just data you own, but public or rented data
- ❖ Siloed data lakes bring inefficient data analysis (e.g. long time analysis because of data movement)
- ❖ Siloed data lakes bring inefficient storage space utilization (e.g. many copies on different storage systems)
- ❖ Multiple siloed storage systems bring further issues on cost, management and scaling
- ❖ Lost metadata concerns due to regulations or business requirements

Spectrum Scale Data Intensive Use Cases

IBM Storage & SDI

Instrument Driven Science and Healthcare

- ALS and CryoEM
- Radio Astronomy
- Weather
- Genomics

Data Driven Engineering

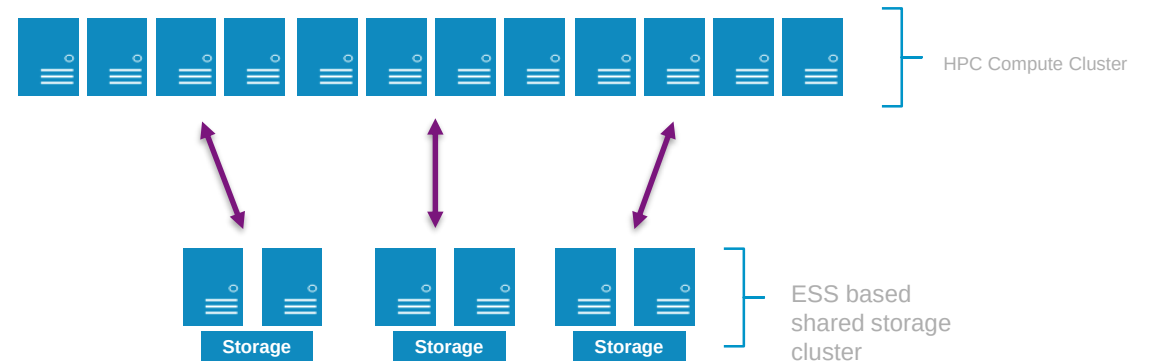
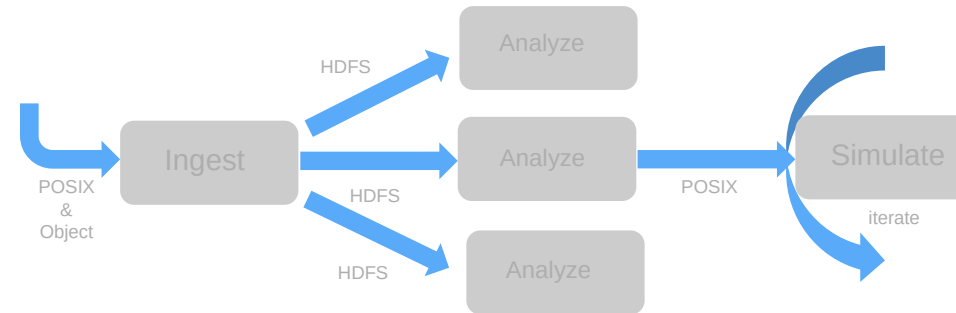
- Modeling and simulation results (i.e. visualization)
- Sensor data analysis
- Financial – market data analysis and reporting
- Supply chain efficiency

Big Data

- Market Insights
- Operational efficiency (including IT)

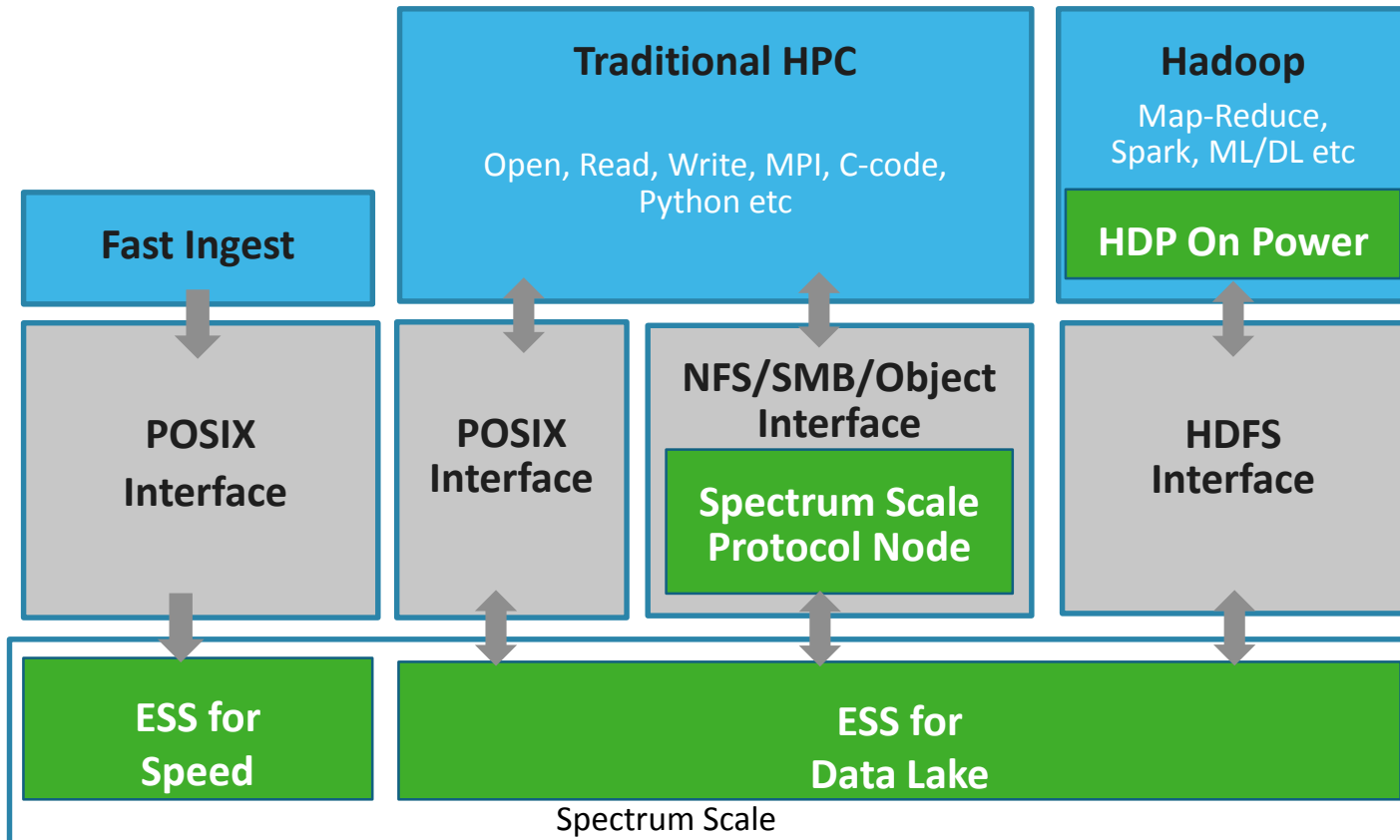
Cognitive

- Personalized Medicine
- Autonomic Driving
- Natural Language Processing



Use Case #1: Integrated HPC and Hadoop

Efficiently transform data into insights with single data lake for HPC & Hadoop



Extend HPC to add modern analytics capabilities

- Efficient movement of data between modern and traditional applications with common namespace
- Spectrum Scale in-place analytics capabilities enable accessing the same data using NFS/SMB/Object/POSIX/HDFS without requiring any modifications to the data
- Improve data reliability and governance with single data lake

Ingest fast and improve time to insight

- POSIX interface combined with ESS Flash storage gives super fast ingest ability
- Common namespace enables running some edge analytics at the ingest layer as well

Control cluster sprawl

- Grow storage independent of compute with ESS
- Up-to 60% less storage footprint
- POWER servers deliver 1.7x throughput compared to Hortonworks on x86

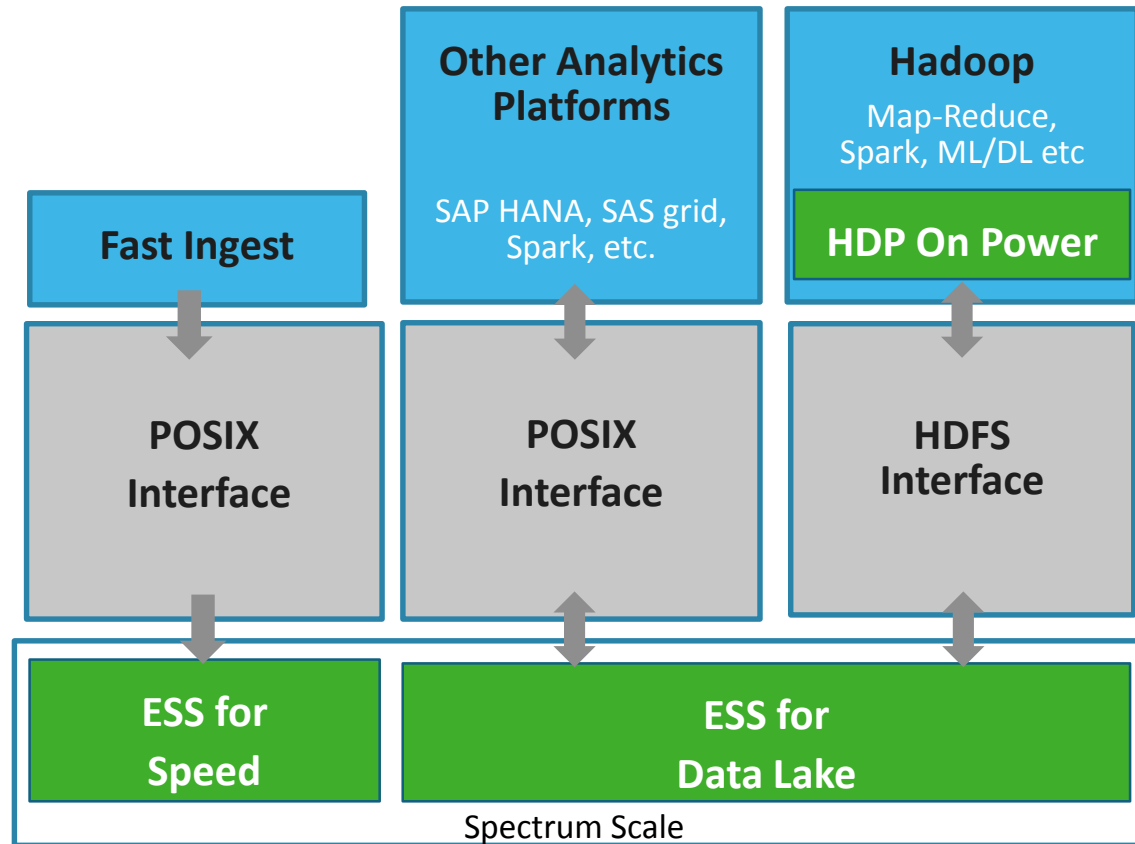
NASA and a Healthcare company from middle east are using common Spectrum Scale data lake to efficiently get insights using traditional HPC and Hadoop analytics.

We are also working with number of existing Spectrum Scale / ESS customers to implement this solution.



Use Case #2 : Integrated Analytics Workflows

Common data lake for all analytics workflows



All analytics workflows on common storage

- Improve data reliability and governance with single data lake
- Build ML/DL workflows that use multiple analytics platforms
- Share data across analytics workflows as appropriate

Ingest fast and improve time to insight

- POSIX interface combined with ESS Flash storage gives super fast ingest ability

Control cluster sprawl

- Grow storage independent of compute with ESS
- Up-to 60% less storage footprint
- POWER servers deliver 1.7x throughput compared to Hortonworks on x86

A bank in South Africa is implementing HDP and SAS grid software on a common ESS based infrastructure.

Step 1: Install HDFS Environment Support

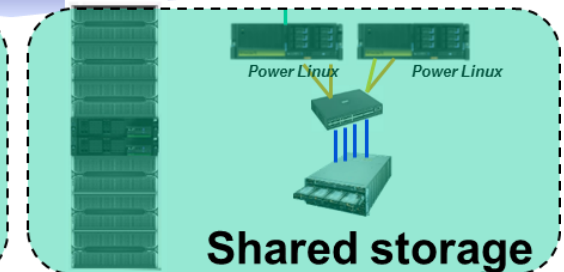
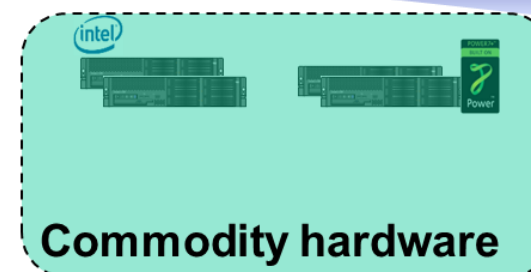
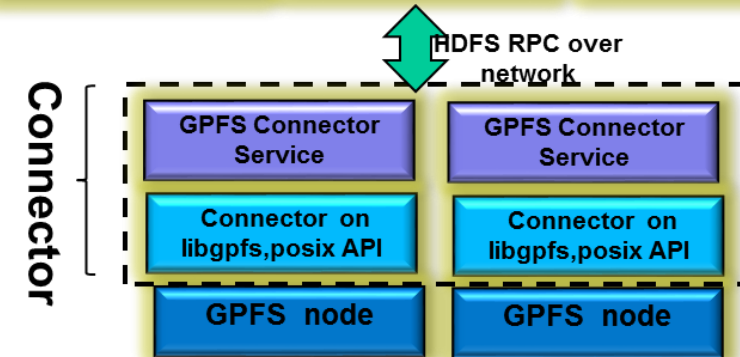
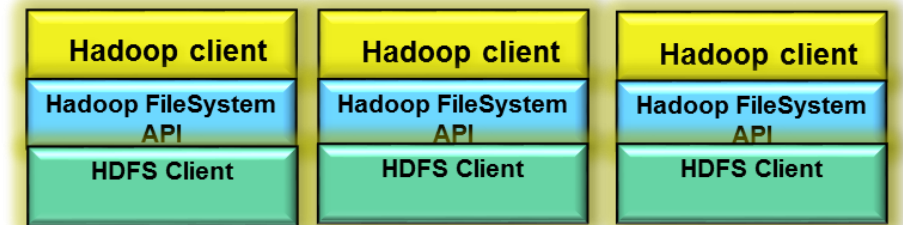
Spectrum Scale completely transparent to Hadoop

- HDFS Transparency makes Spectrum Scale transparent
- Connector Works with 4.1.X/ 4.2.x/5.0.x
- Shipped with Spectrum Scale 4.2.x/5.0.x
 - Download the latest from [IBM developerWorks Spectrum Scale wiki](#)

HDFS Transparency and Spectrum Scale services could be managed from HortonWorks HDP Ambari GUI

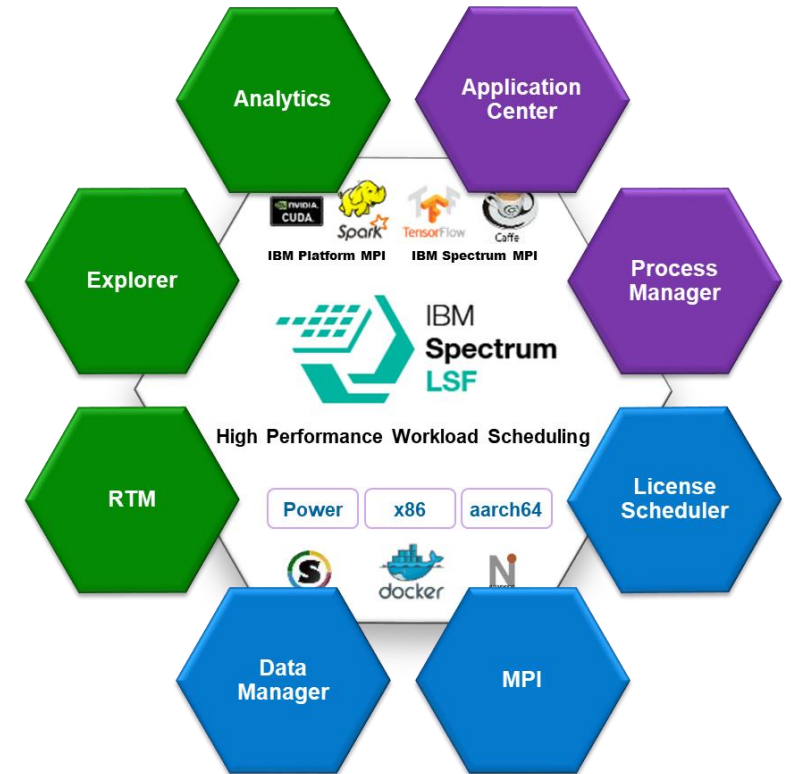


`hdfs://hostnameX:portnumber`



Step 2 : Deploy LSF Hadoop Connector Script

- Use the IBM Spectrum LSF integration with Apache Hadoop to submit Hadoop as regular LSF parallel jobs
- Hadoop connector script (lsfhadoop.sh) automatically provisions an open source Hadoop cluster within LSF allocated resources, then submits actual MapReduce workloads into this Hadoop cluster
- Each LSF Hadoop job has its own resource (cluster), the integration provides a multi-tenancy environment to allow multiple users to share the common pool of HPC cluster resources. LSF is able to collect resource usage of MapReduce workloads as normal LSF parallel jobs and has full control of the job life cycle.
- After the job is complete, LSF shuts down the Hadoop cluster.
- By default, configures the Hadoop cluster with direct access to shared file systems and does not require HDFS.



Step 3: Submit Hadoop as Regular LSF Parallel Job

1. Download and deploy Apache Hadoop. The latest version is available from the official Apache Hadoop site: <http://hadoop.apache.org/releases>
2. Set the **\$HADOOP_HOME** environment variable as the file path to the Hadoop installation directory. If you do not set the **\$HADOOP_HOME** environment before using the integration, you must run the **lsfhadoop.sh** connector script with the **--hadoop-dir** option to specify the file path to the Hadoop installation directory.
3. Set the **\$JAVA_HOME** environment variable as the file path to the Java runtime installation directory. If you do not set the **\$JAVA_HOME** environment before using the integration, you must run the **lsfhadoop.sh** connector script with the **--java-home** option to specify the file path to the Hadoop installation directory.
4. Use the **bsub** command to submit the Hadoop application to LSF through the **lsfhadoop.sh** connector script.

To specify a Hadoop command to run the Hadoop workload, use **hadoop jar** as a subcommand of the **lsfhadoop.sh** connector script:

```
bsub bsub_options lsfhadoop.sh [-h] [--hadoop-dir file_path] [--java-home file_path] [--config-dir file_path] [--user-hdfs] [-debug] hadoop jar jarfile_name jarfile_parameters
```

To specify a script that runs multiple Hadoop commands within a single LSF job, run the script as an argument to the **lsfhadoop.sh** connector script:

```
bsub bsub_options lsfhadoop.sh [-h] [--hadoop-dir file_path] [--java-home file_path] [--config-dir file_path] [--user-hdfs] [-debug] script_name
```

References

- LSF with Apache Hadoop [here](#)
- Download the latest HDFS Transparency package from IBM Spectrum Scale developerworks wiki [here](#)
- Download the latest Ambari integration mpack from IBM Spectrum Scale developerWorks wiki
- Refer the *Big Data and Analytics Guide* from [IBM KC](#)
- Any questions, mail to scale@us.ibm.com

Thank You.
IBM Storage & SDI

