# Cloud Storage: PUT is the new rename()

Steve Loughran
Hortonworks R&D

stevel@hortonworks.com

@steveloughran

June 2018

# Speaker & Talk Overview

## Hadoop committer; busy on object storage

- Why do the cloud stores matter?
- Where do the stores stand today?
- What new & interesting stuff is there?




HORTONWORKS

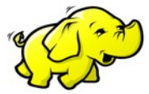
# Why cloud storage?

- Long-lived persistent store for cloud-hosted analytics applications
- HDFS Backup/restore
- Sharing, data exchange, data collection workflows, …





`org.apache.hadoop.fs.FileSystem`



`hdfs`


`wasb`

`s3a`

`adl`

`gs`

`swift`


`abfs`




HORTONWORKS

# All examples in Spark 2.3; "S3 landsat CSV" as source

```
val landsatCsvGZOnS3 = new Path("s3a://landsat-pds/scene_list.gz")

val landsatCsvGZ = new Path("file:///tmp/scene_list.gz")

copyFile(landsatCsvGZOnS3, landsatCsvGZ, conf, false)

val csvSchema = LandsatIO.buildCsvSchema()

val csvDataFrame = LandsatIO.addLandsatColumns(
  spark.read.options(LandsatIO.CsvOptions).
    schema(csvSchema).csv(landsatCsvGZ.toUri.toString))

val filteredCSV = csvDataFrame.
  sample(false, 0.01d).
  filter("cloudCover < 15 and year=2014").cache()
```




HORTONWORKS

# Azure wasb://
# The ~filesystem for Azure


HORTONWORKS®

# Azure wasb:// connector shipping and stable

- REST API
- Tested heavily by Microsoft
- Tested heavily by Hortonworks
- New! adaptive seek() algorithm
- New! more Ranger lockdown

Note: Ranger implemented client-side — needs locked-down clients to work



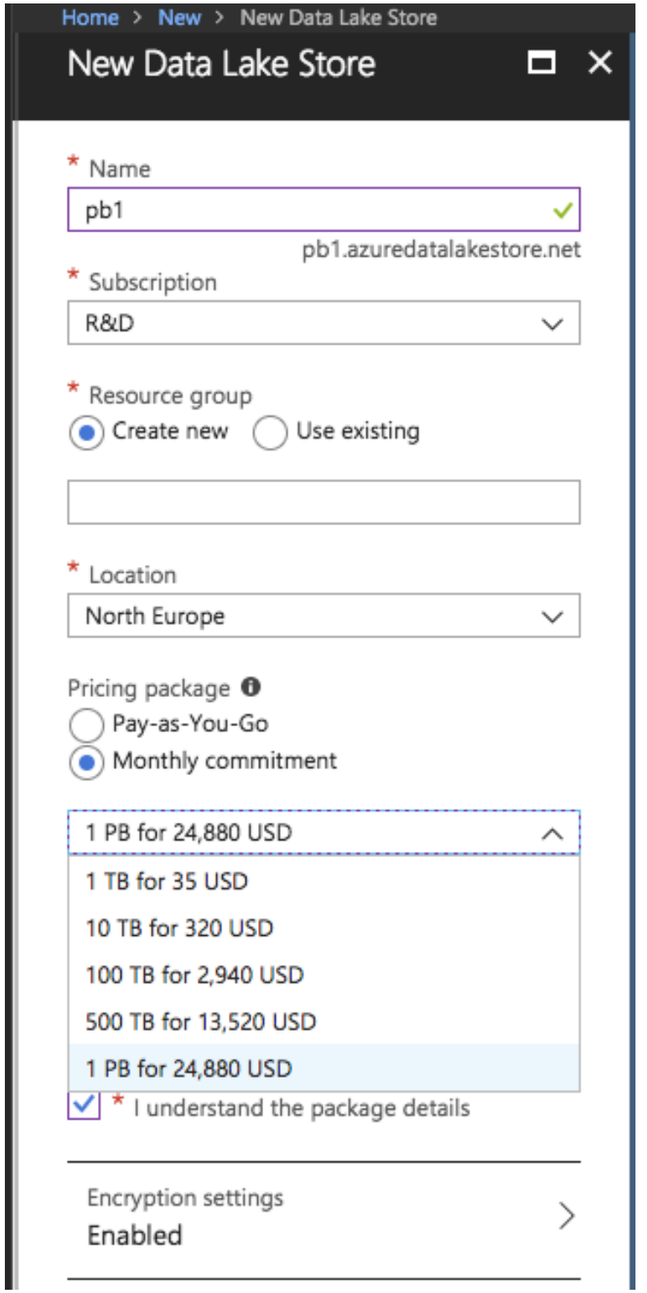

HORTONWORKS


Azure Datalake
adl://
HORTONWORKS

# Azure Datalake: Analytics Store

- API matches Hadoop APIs 1:1
- hadoop-azuredatalake JAR
- Runs on YARN
- Authenticates via OAuth2

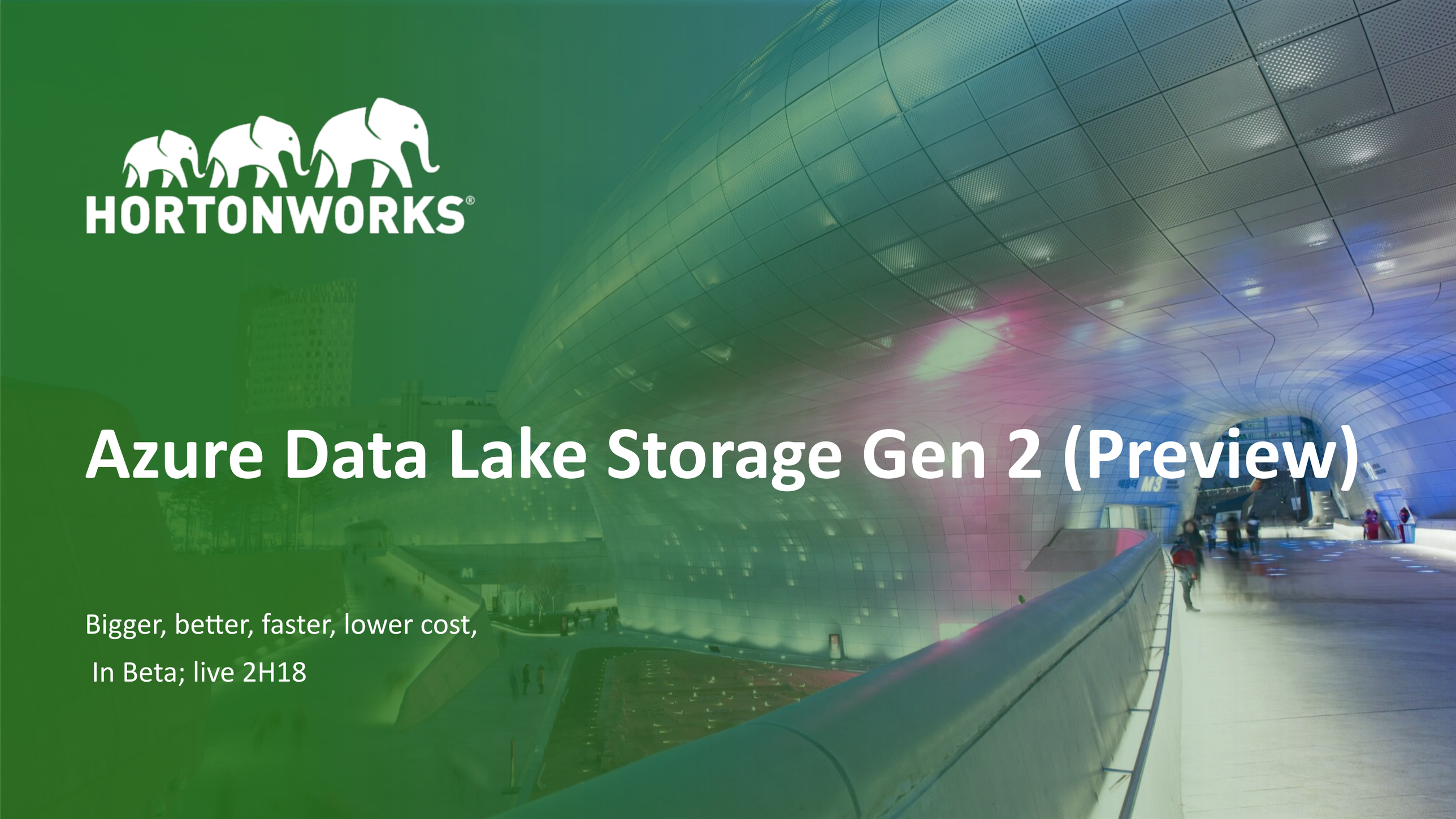
Ongoing : resilience & supportability







HORTONWORKS




# Azure Data Lake Storage Gen 2 (Preview)

Bigger, better, faster, lower cost,

In Beta; live 2H18

# abfs:// is the connector

- New REST API
- HADOOP-15407: add the hadoop connector
- Store and client to succeed wasb:// and adl://
- Old endpoints to remain; wasb:// will continue to work.

abfs:// connector being developed with store; usual stabilization process




HORTONWORKS

# abfs demo:

```
val orcDataAbfs = new Path(abfs, "orcData")

logDuration("write to ABFS ORC") {
  filteredCSV.write.
    partitionBy("year", "month").
    mode(SaveMode.Append).format("orc").
    save(orcDataAbfs.toString)
}
```




HORTONWORKS®

# Amazon S3: Use with Care: inconsistency is its flaw


HORTONWORKS®

# S3Guard: High Performance & Consistent Metadata for S3

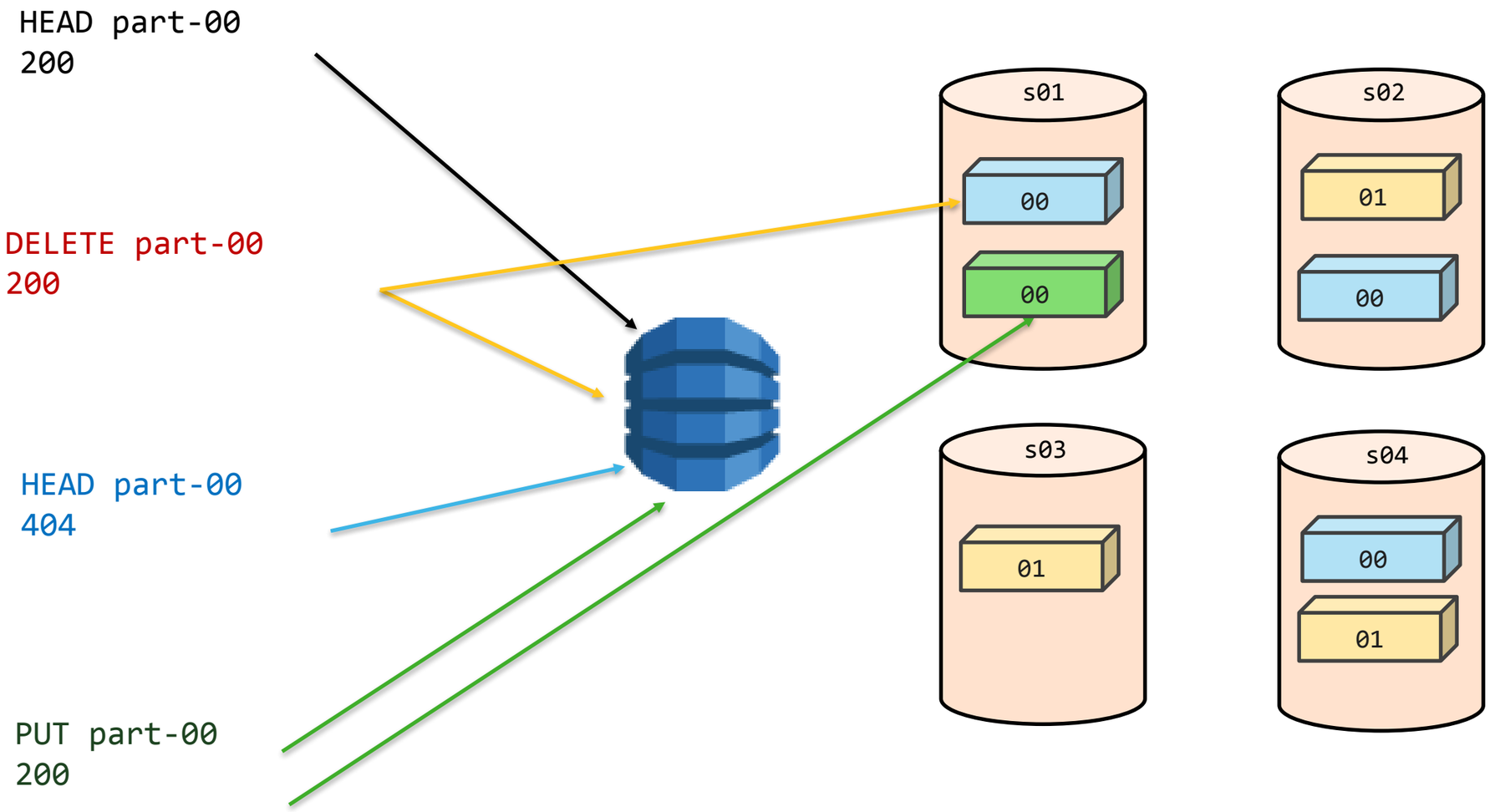
- Uses DynamoDB for a consistent view
- Ensures subsequent operations see new files
- Filters out recently deleted files
- Can speed up `listStatus()` & `getFileStatus()` calls
- Stops `rename()` (and so job commit) missing files

Prevents data loss during Hadoop, Hive, Spark queries




HORTONWORKS

# S3Guard: fast consistent metadata via Dynamo DB







HORTONWORKS

# S3Guard: bind a bucket to a store, initialize the table, use

```
<property>
  <name>fs.s3a.bucket.hwdev-steve-ireland-new.metadatastore.impl</name>
  <value>org.apache.hadoop.fs.s3a.s3guard.DynamoDBMetadataStore</value>
</property>
```

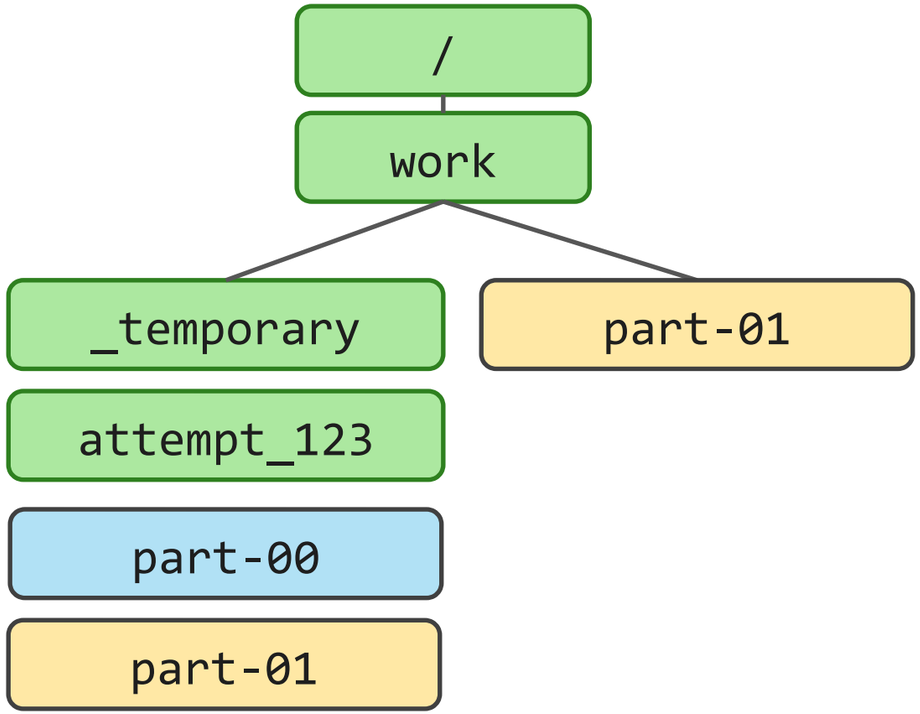
**hadoop s3guard init -read 50 -write 50  s3a://hwdev-steve-ireland-new**

```
…
Metadata Store Diagnostics:
ARN=arn:aws:dynamodb:eu-west-1:980678866538:table/hwdev-steve-ireland-new
description=S3Guard metadata store in DynamoDB
name=hwdev-steve-ireland-new
read-capacity=20
region=eu-west-1
retryPolicy=ExponentialBackoffRetry(maxRetries=9, sleepTime=100 MILLISECONDS)
size=15497
status=ACTIVE
```




HORTONWORKS

# Next: the commit problem



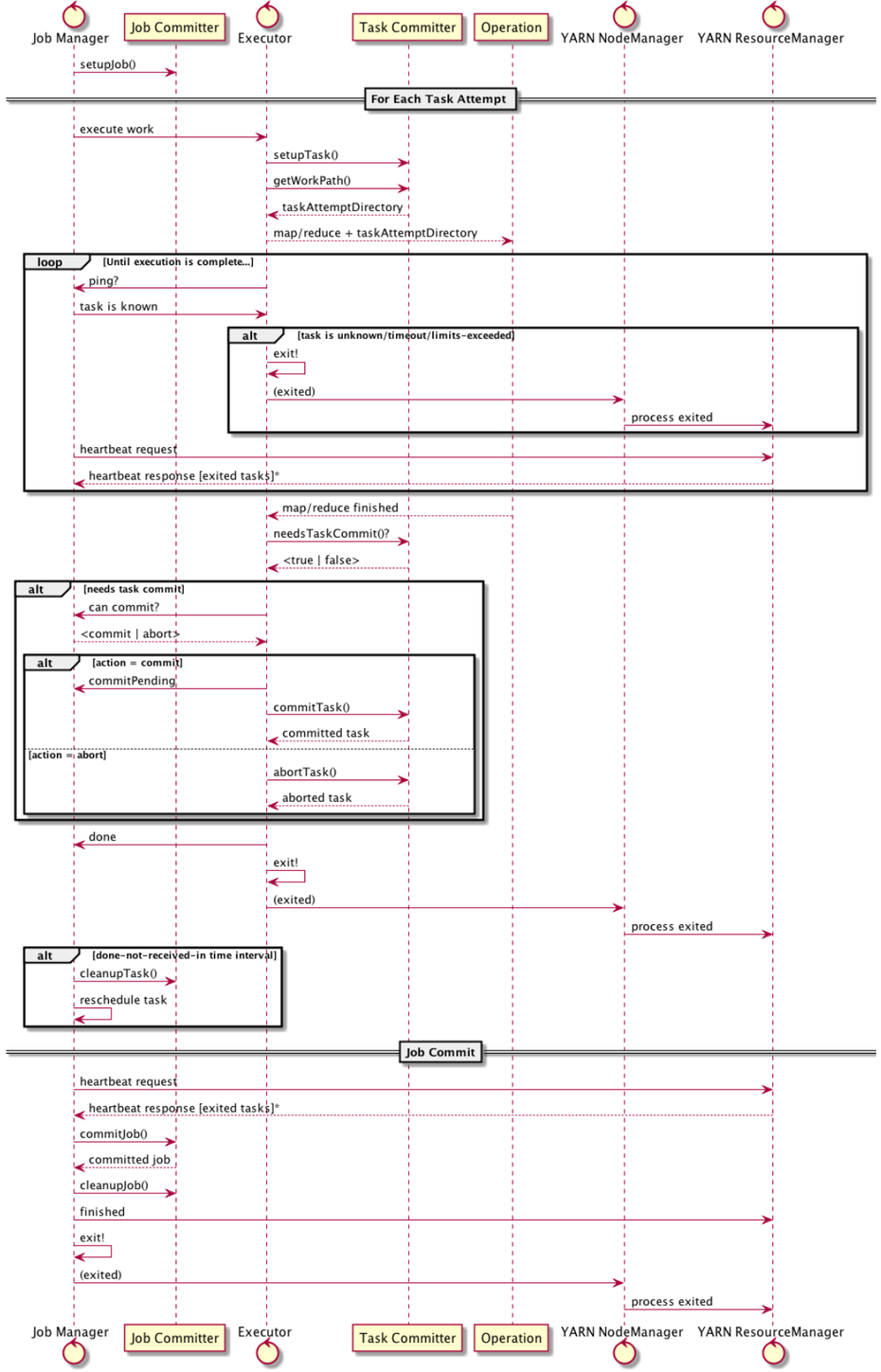

- Task commit: rename to job directory
- Job commit: rename to final directory
- Filesystems & most stores: O(directories)
- S3: COPY operation @6/10 MB/s
- Without S3Guard, risk of loss

```
rename("/work/temporary/attempt_123/part-01", "/work/")
```




HORTONWORKS

# MR
# Commit Protocol

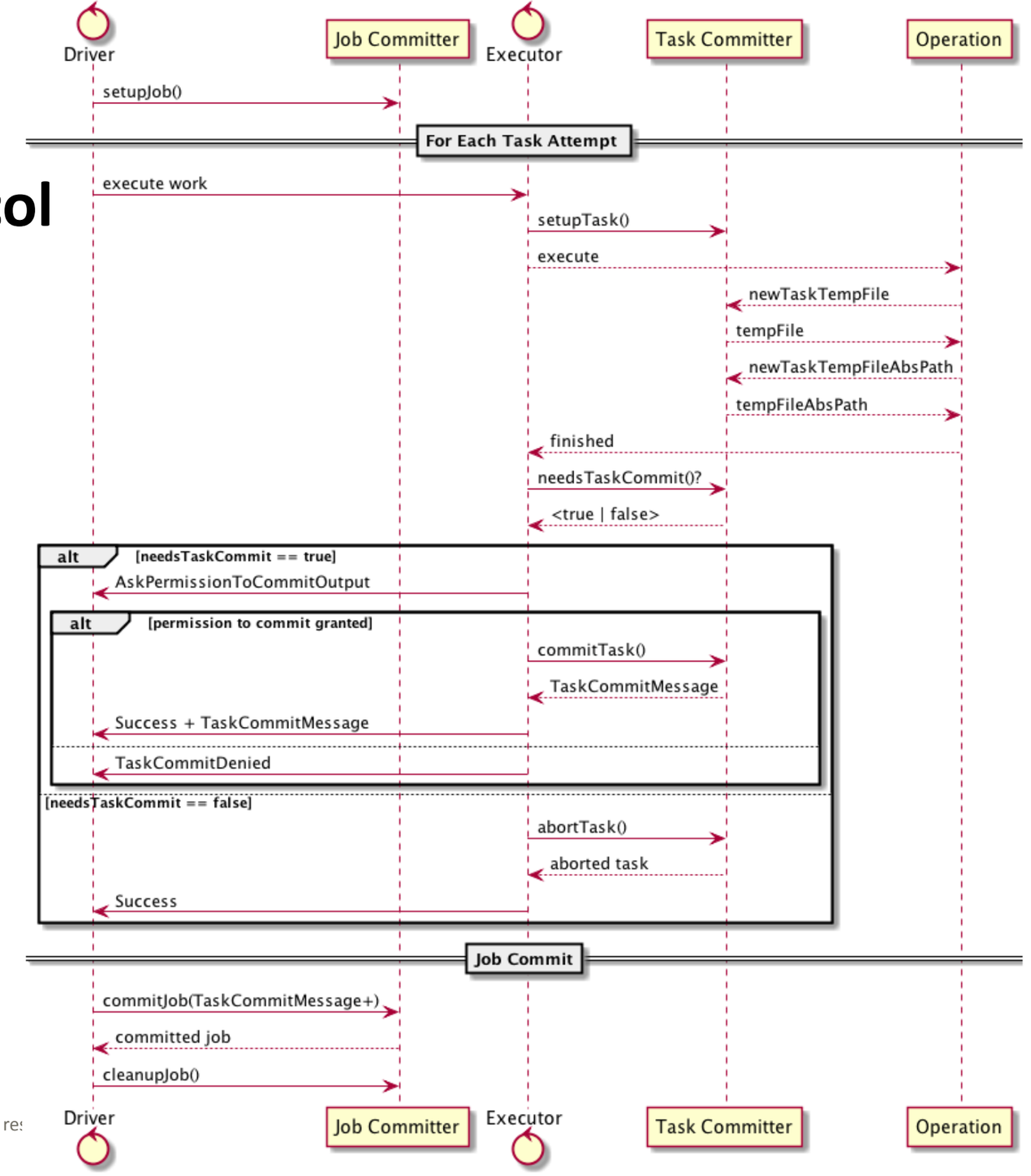






HORTONWORKS

# Spark Commit Protocol







HORTONWORKS

# PUT replaces rename()

Task 00:

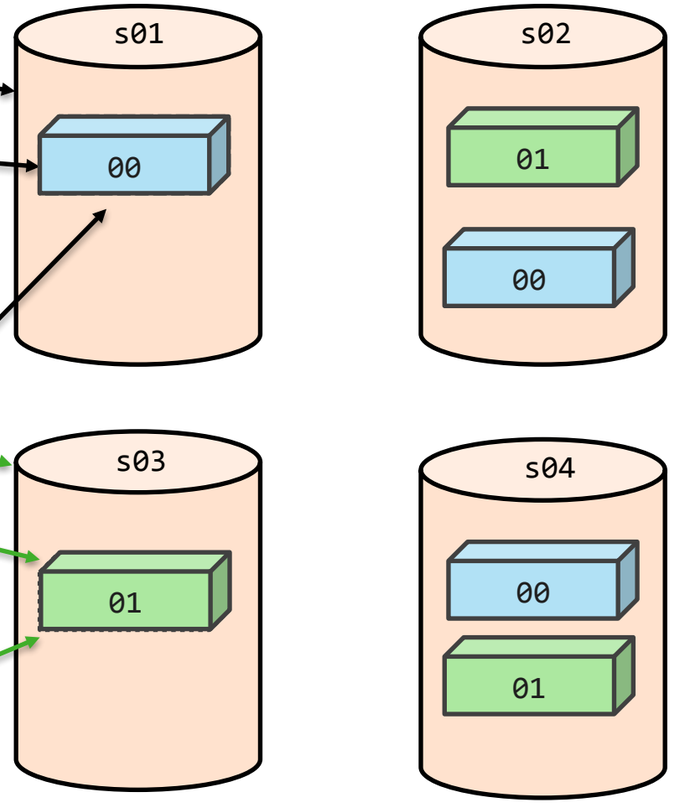
```
put00 = POST "Multipart Upload" /work/part00
POST "Block" put00, /work/part00, data
```

Task 01:

```
put01 = POST "Multipart Upload" /work/part01
POST "Block" put01, /work/part01, data
```

Job Commit

```
POST "Complete" put00, /work/part00
POST "Complete" put01, /work/part01
```







HORTONWORKS

# S3A Committers + Spark binding

```
spark.sql.sources.commitProtocolClass
  org.apache.spark.internal.io.cloud.PathOutputCommitProtocol

spark.sql.parquet.output.committer.class
  org.apache.spark.internal.io.cloud.BindingParquetOutputCommitter

spark.hadoop.fs.s3a.committer.name staging
spark.hadoop.fs.s3a.committer.staging.conflict-mode replace
```

**Also**

```
spark.hadoop.fs.s3a.committer.name partitioned
spark.hadoop.fs.s3a.committer.name magic
```




HORTONWORKS

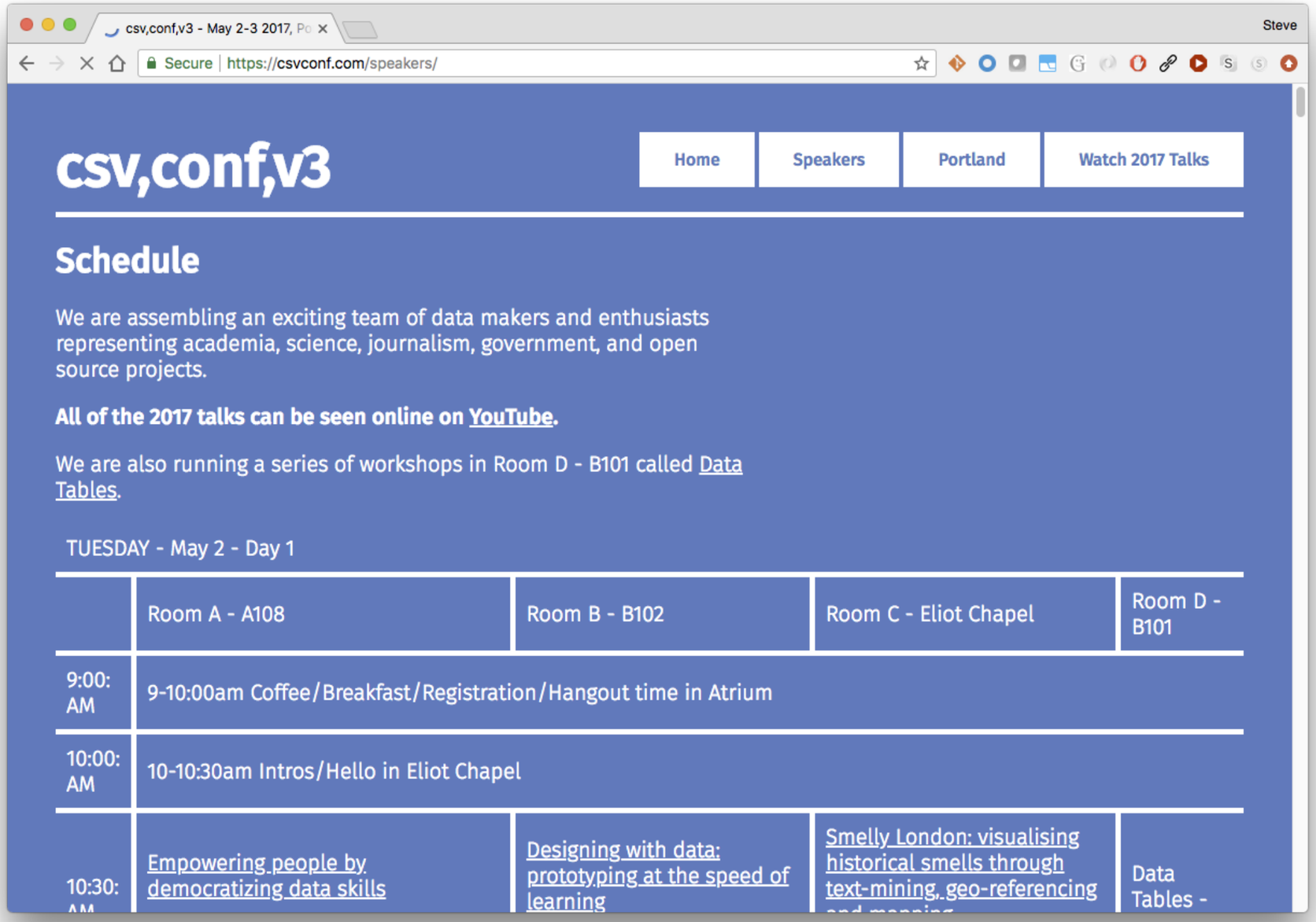
# S3A Committer writing ORC and Parquet to S3A

```
val orcDataS3 = new Path(ireland, "orcData")

orcWasbDataframe.write.
  partitionBy("year", "month").
  mode(SaveMode.Append).orc(orcDataS3.toString)

val orcS3DF = spark.read.orc(orcData.toString).as[LandsatImage]

val parquetS3 = new Path(ireland, "parquetData")

orcDF.write.
  partitionBy("year", "month").
  mode(SaveMode.Append).parquet(parquetData.toString)

cat(new Path(parquetData, "_SUCCESS"))
```




HORTONWORKS

# CSV files: the "Dark Matter" of data







HORTONWORKS

# CSV: less a format, more a mess

- Commas vs tabs? Mixed?
- Single/double/no quotes. Escaping?
- Encoding?
- Schema? At best: header
- missing columns?
- NULL?
- Numbers?
- Date and time?
- Compression?
- CSV injection attacks (cells with = as first char) [Excel, google docs]




HORTONWORKS

# S3 Select: SQL Queries on CSV/JSON In S3 Itself

- SQL Select issued when opening a file to read
- Filter rows from WHERE clause
- Project subset of columns
- Use CSV header for column names
- Cost of GET as normal
- Save on bandwidth




HORTONWORKS

# S3 SELECT against the landsat index dataset

```
# examine data

hadoop s3guard select -header use -compression gzip \
  -limit 10 \
  s3a://landsat-pds/scene_list.gz \
  "SELECT s.entityId FROM S3OBJECT s WHERE s.cloudCover = '0.0'"

# copy subset of data from S3 to ADL

hadoop s3guard select -header use -compression gzip \
  -out adl://steveleu.azuredatalakestore.net/landsat.csv \
  s3a://landsat-pds/scene_list.gz \
  "SELECT * FROM S3OBJECT s WHERE s.cloudCover = '0.0'"
```




HORTONWORKS

# DEMO: remote client, 300MB /1M rows of AWS landsat index

```
time hadoop fs -text $LANDSATGZ  | grep ",0.0,"
 => 44s

// filter in the select
time hadoop s3guard select -header use -compression gzip $LANDSATGZ \
 "SELECT * from S3OBJECT s where s.cloudCover = '0.0'"

 => 12.20s

// select and project

time hadoop s3guard select -header use -compression gzip $LANDSATGZ \
  "SELECT s.entityId FROM S3OBJECT s WHERE s.cloudCover = '0.0' LIMIT 100"

=> 4s
```




HORTONWORKS

# S3 Select Transcoding

Input

- JSON-per-row format
- UTF-8 CSV with configured separator, quote policy, optional header
- Raw or .gz

Output

- JSON-per-row format
- UTF-8 CSV with configured separator, quote policy, NO HEADER
- loss of CSV header hurts Spark CSV Schema inference




HORTONWORKS

# Integration: Work in Progress. Hadoop 3.2?

- Later AWS SDK JAR, upgrades "mildly stressful"
- new `S3AFileSystem.select(path, expression)` method
- Filesystem config options for: compression, separator, quotes
- TODO: open( ) call which takes optional & mandatory properties (HADOOP-15229)
- TODO: adoption by Hive, Spark, …




HORTONWORKS

# Google Cloud Storage

HORTONWORKS®

# Google Cloud: Increasing use drives storage

1. More users (TensorFlow motivator)
2. More tests (i.e. GCS team using ASF test suites, ...)
3. Google starting to contribute optimizations back to the OSS projects (e.g. HDFS-13056 for exposing consistent checksums
4. +Cloud BigTable offers HBase APIs

Attend: Running Apache Hadoop on the Google Cloud Platform
Grand Ballroom 220A, Wednesday, 16:40




HORTONWORKS

# What else?

# Security & Encryption

- Enable everywhere: transparent encryption at rest
- Customer-key-managed encryption available
  — may have cost/performance impact
- Key-with-request encryption: usable with care
- Client side encryption. Troublesome

Permissions:

- ADL, GCS: permissions managed in store
- WASB: Ranger checks in connector: needs locked-down client
- S3: IAM Policies (server side, limited syntax, minimal hadoop support)




HORTONWORKS

# DistCP++

- Old HDFS ⟺ HDFS
- New: HDFS ⟺ cloud, container ⟺ container,  cloud ⟺  cloud,

We are going to have to evolve DistCP for this

1. New: Cloud-optimized -delete option
2. TODO: stop the rename().
3. Need a story for incremental updates across stores




HORTONWORKS

# How do they compare? (excluding abfs)

| | AWS S3 | Azure WASB | Azure ADL | Google GCS |
|---|---|---|---|---|
| **Can run HBase (i.e. "filesystem")** | No | Yes | No | No |
| **Safe destination of work** | With S3Guard or EMRFS consistent view | Yes | Yes | Yes |
| **Fast destination of work** | With S3A Committers | Yes | Yes | Yes |
| **Security** | IAM Roles | Ranger in locked-down client | Ranger | ACLs |
| **Encryption** | Server-side; can use AWS KMS | Yes, keys in Key Vault | Yes, keys in Key Vault | always; can supply key with IO request |




HORTONWORKS

# To Conclude

- All cloud infrastructures have object stores we can use
- Width and depth of product offerings increasing
- Azure teams most engaged with the Hadoop project
- S3 is the least suited as a destination, but we have solutions there.
- Cloud Storage evolving: S3 Select one notable example

**Data analytics is a key cloud workload; everything is adapting**




HORTONWORKS

# Questions?

stevel@hortonworks.com

HORTONWORKS®

# Thank you

stevel@hortonworks.com

# Code used in this talk

- Hadoop trunk + HADOOP-15407 (abfs://) and HADOOP-15364 (S3 Select) patches
- Spark master with SPARK-23977 PR applied
- Hortonworks Spark Cloud integration module
  various utils + schema for Landsat
  https://github.com/hortonworks-spark/cloud-integration
- Cloudstore: diagnostics & utils
  https://github.com/steveloughran/cloudstore




HORTONWORKS