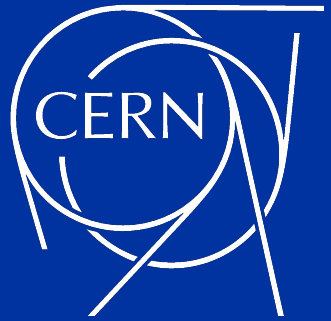


Hadoop service roadmap

Emil Kleszcz @ CERN

04 Dec 2023

Overview

1. Main achievements of 2023
2. Main operational plans for 2024 and beyond
3. ATS-IT engagement project proposal
4. Evaluation of technologies (PoCs)
5. Trends and future

Main achievements of 2023

5 key areas

Achievements of 2023

Regular Operations

Service Improvement

New Feature

1) Transition to OpenJDK

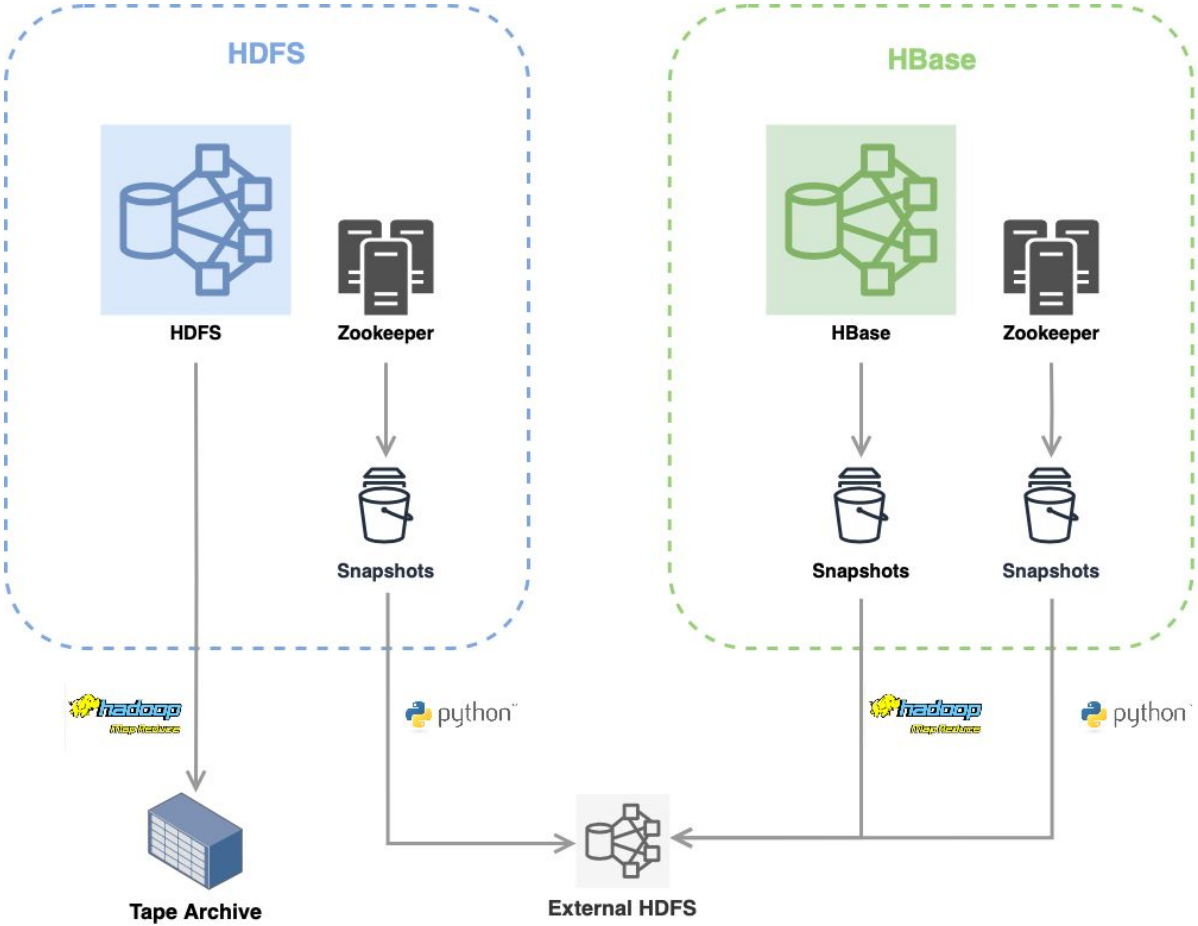
- Retirement of no longer supported Oracle JDK
- Build Hadoop 3.3.6 and Spark 3.5 with OpenJDK 8/11
- Migration of cluster runtime environment

2) Disaster recovery strategy for Hadoop

- Support for HBase DR using built-in snapshots functionality
- Review and redesign of HDFS backup: monitoring, CTA integration
- Replication of Zookeeper snapshots

Hadoop disaster recovery

High-level architecture



Achievements of 2023

3) Isolation of NXCALS HBase instance 

- Installation and migration process, performed during a technical stop
- Two separate clusters, HBase with NVMEs
- Improves service performance, reliability, and operations

4) Support of Alma 9 and RHEL 9 

- Client software packages released
- Hadoop client puppet module adapted (nftables, …)
- Zookeeper Puppet module supported

Achievements of 2023

5) General improvements to the clusters 

- Network configuration (IPv6 disabled, Landb sets mgmt, ...)
- Throughput improvements (multi-threading with salting enabled, up to 750MB/s)
- Automatized regions management (cleanup of empty, merging small adjacent, …)
- 2FA enabled (only NXCALS PRO to do next year)

Main operational plans

for 2024 and beyond

Plans for 2024

1) **Upgrade to the latest versions** 
   - HBase 2.5.6 (current 2.3.4)
   - Hadoop 3.3.6 (current 3.2.1)
   - Spark 4.0 (current 3.5)
   - Deprecate Spark 2.4 and Hadoop 2.7

2) **Hardware renovation** 
   - Replace NXCALS namenodes (2022 order)
   - Replace Analytix and NXCALS datanodes (2023 order)

3) **Business continuity cold tests** 
   - Test steps and time needed to restore the services

Plans for 2024

4) Migration of clusters to Alma 9

- Packages already available

5) Consolidation of monitoring tools

- Collectd plugins: extend phoenix plugin, review existing metrics
- Rundeck: improve logging, migrate cron jobs, move to dedicated deployment
- OpenSearch: move to dedicated deployment

6) Improve security management

- SSO integration with Apache Knox
- Apache Ranger for ACL management

Plans beyond 2024

1) **Improve performance and troubleshooting toolset for HBase**

2) **Introduce aliases on cluster namenodes**
   - Improve the impact of future interventions
   - Investigate other external components such as HAProxy

3) **Deploy client edge nodes in K8s**
   - To improve resource scaling

ATS-IT engagement

Project proposal (PSO)

Overview

NXCALS review, with a purpose to:

- Ensure smooth IT operations of NXCALS through LS3 and the beginning of Run 4 (HL-LHC)
- Enable timely planning and provisioning of the necessary hardware, software and HRs

Two-phase project:

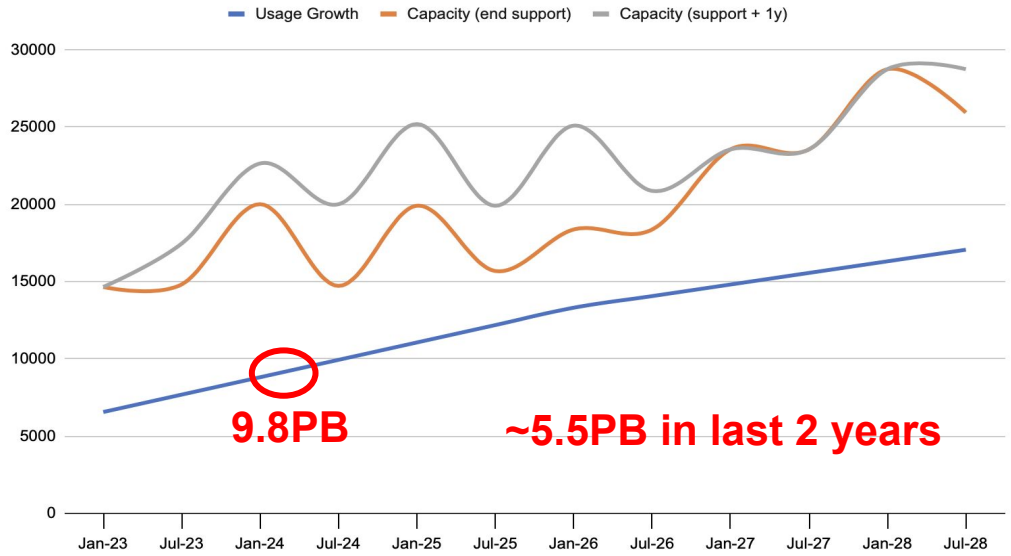
- 1st - evaluation (1 year?)
- 2nd - implementation (2 more years?)

Many thanks to our NXCALS colleagues from BE for their support and initiative :-)

Time window for new deployments

Evaluate storage needs and software requirements

- Growing throughput and storage capacity needs
- Long-term software support and performance capabilities
- We need: **resources** and **time** to evaluate and establish a long-term plan



Project proposal

- **Scope**
  - Review NXCALS IT dependencies such as IT managed software/hardware
  - Plan for the growing NXCALS data throughput and storage needs
  - Test the expected performance and capacity needs in production
  - Estimate and plan the work needed for the maintenance and service evolution for Run 4
  - Evaluate alternative technologies to handle the expected data rate/storage capacity

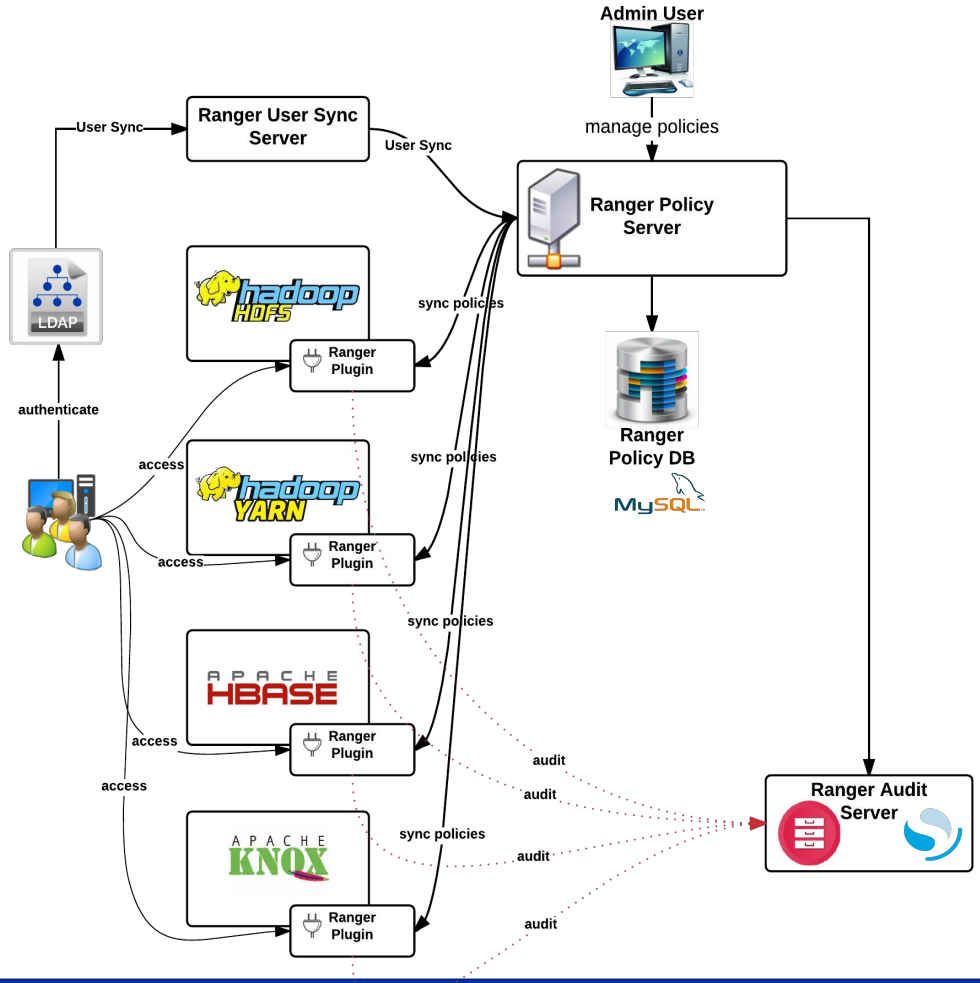
- **Results**
  - Analyse technical risks not related to data growth (obsolescence, maintenance, etc.)
  - Analyse technical risks related to data growth and establish a subsequent mitigation plan
  - Analyse and reach agreement with ATS users on their throughput and storage requirements
  - Prepare PoCs for alternative solutions and test against forecasted data rates
  - Establish a tentative plan to implement the above in a subsequent phase

Evaluation of technologies

Proof of Concepts

Apache Ranger

- **Objective**
  - provide comprehensive security across the CERN Hadoop ecosystem
- **Outcome**
  - Installed Apache Ranger
  - Configured plugins: HDFS, Yarn, HBase
  - LDAP integration
  - Auditing with HDFS backend
  - Example policies
  - Extensive docs
  - Mature project, with poor official docs and not much recent support
  - Presented on ApacheCon23
- **Future**
  - Auditing with OpenSearch backend
  - Outsource mgmt. of policies
  - Explore HA
  - Puppetize and deploy



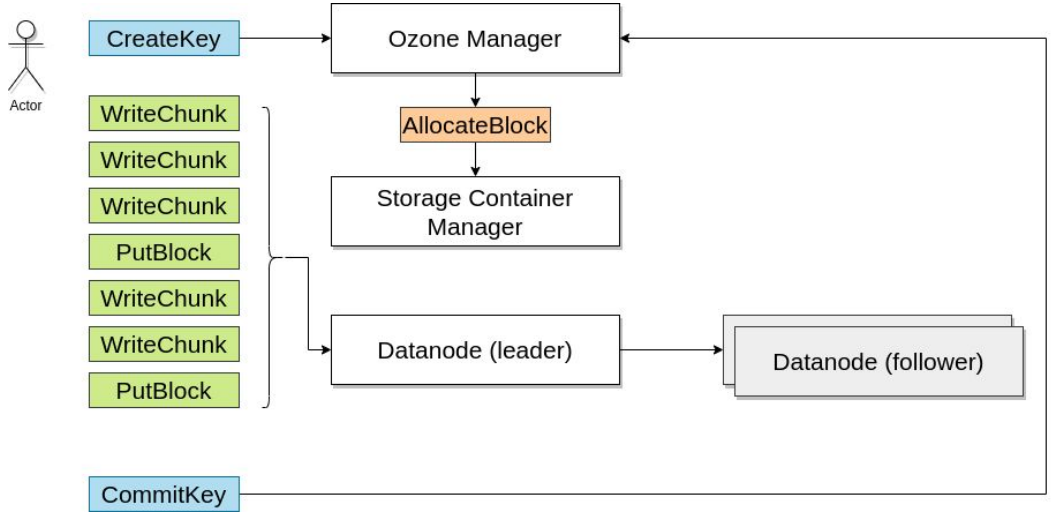
Apache Ozone

- Highly scalable distributed storage
- FS-like ops
- HDFS and S3 compatible
- Support for both Yarn and K8S

- Scales to Exabyte (any size of files)
- Billions of objects (linear scaling)
- Namespace: volume with buckets
- Keys (objects) stored under buckets

Write a file

```
# Create volume and bucket
$ ozone sh volume create /vol1
$ ozone sh bucket create /vol1/buck1

# write a file
$ ozone fs -mkdir -p /vol1/buck1/dir1
$ ozone fs -touch /vol1/buck1/dir1/key1

# Cannot create file under root or volume
$ ozone fs -touch /vol1/key1
```

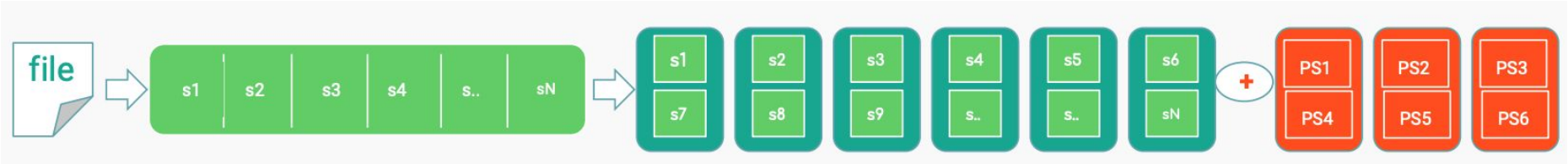


HDFS Erasure Coding

- Why? Replication is expensive (overhead)
- EC provides high level of fault-tolerance
- Requires less storage space
- RS (N,K) = (# data blocks, # parity blocks)
- 6 blocks: 6*3 (replica), 6+3 (EC RS(6,3))
- More details: [IT-DA blog post]

Replication Policy	Additional Storage Cost	Fault Tolerance
RS(3,2)	66%	2
RS(6,3)	50%	3
RS(10,4)	40%	4
3x replication	200%	2

Write a file



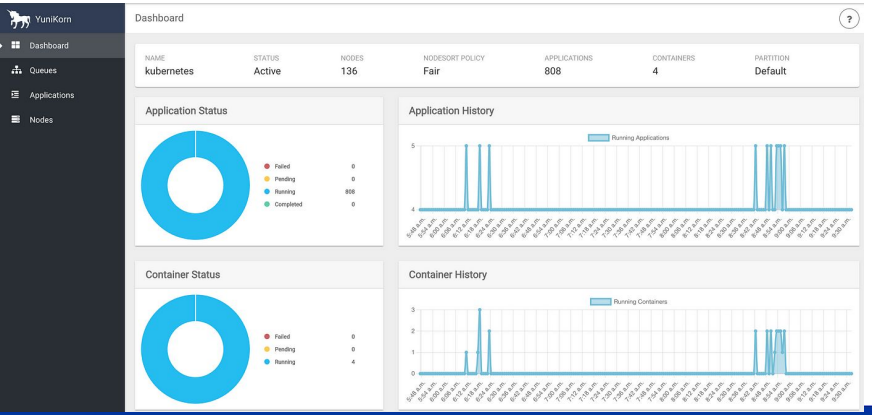
Data is striped

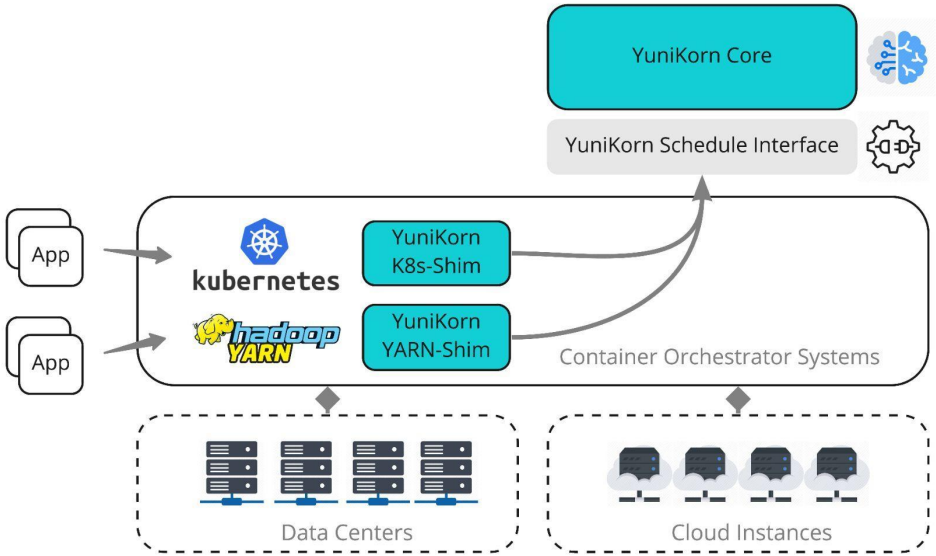
Distributed to N data chunks across cluster nodes

Generate K parity chunks using Reed-Salomon(N,K)

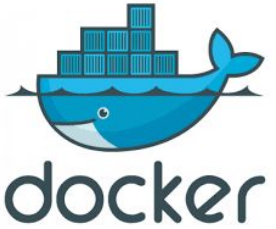
Apache YuniKorn

- Light-weight resource scheduler
- Supports Yarn and K8S (different shim layer)
- Adopts to different RM implementations
- Suitable for batch workloads
- Designed by Yarn developers





Yarn docker containers



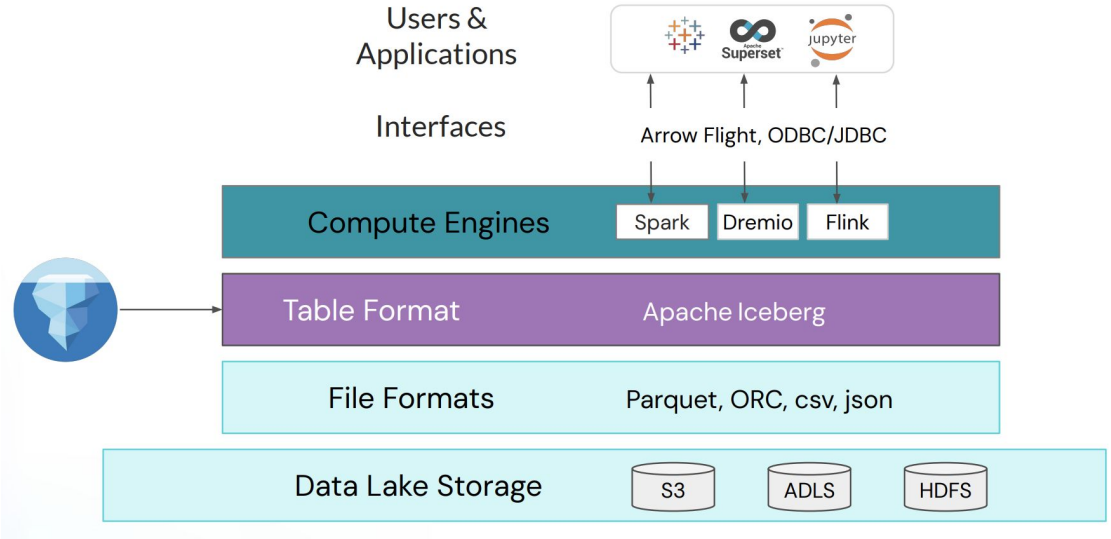
- Idea: NM launches Yarn container inside Docker container instead on the host machine directly - thanks to Linux Container Executor
- Docker client must be installed on all the NMs
- and Docker daemon must be running on all NMs

```
# Pull image (To prevent timeouts load the image in the Docker daemon's cache on the NodeManager hosts
docker pull library/openjdk:8

# Submit spark job
HADOOP_HOME=/usr/hdp/hadoop
SPARK_HOME=/usr/hdp/spark
MOUNTS="$HADOOP_HOME:$HADOOP_HOME:ro,/etc/passwd:/etc/passwd:ro,/etc/group:/etc/group:ro"
IMAGE_ID="library/openjdk:8"

$SPARK_HOME/bin/spark-shell --master yarn \
   --conf spark.yarn.appMasterEnv.YARN_CONTAINER_RUNTIME_TYPE=docker \
   --conf spark.yarn.appMasterEnv.YARN_CONTAINER_RUNTIME_DOCKER_IMAGE=$IMAGE_ID \
   --conf spark.yarn.appMasterEnv.YARN_CONTAINER_RUNTIME_DOCKER_MOUNTS=$MOUNTS \
   --conf spark.executorEnv.YARN_CONTAINER_RUNTIME_TYPE=docker \
   --conf spark.executorEnv.YARN_CONTAINER_RUNTIME_DOCKER_IMAGE=$IMAGE_ID \
   --conf spark.executorEnv.YARN_CONTAINER_RUNTIME_DOCKER_MOUNTS=$MOUNTS
```

Apache Iceberg

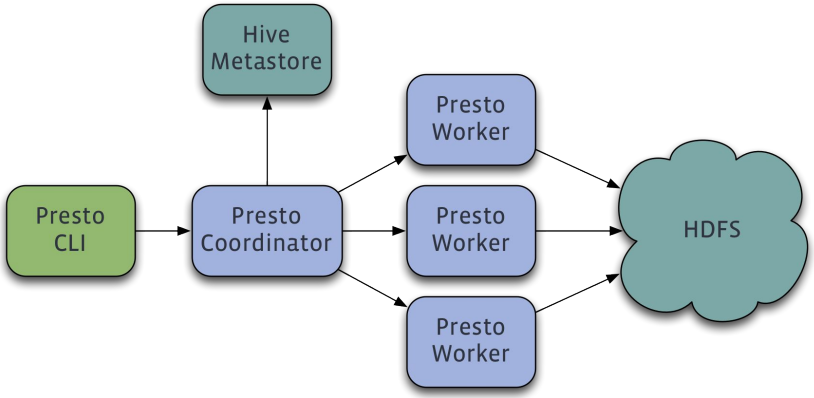
- Table format for large-scale analytics
- Storage agnostic (HDFS, Ozone, S3, …)
- Data in Parquet, meta in Avro
- Scales to petabytes

- Partition evolutions allowed
- ACID transactions supported
- Used with many computing engines
- More performant than Hive

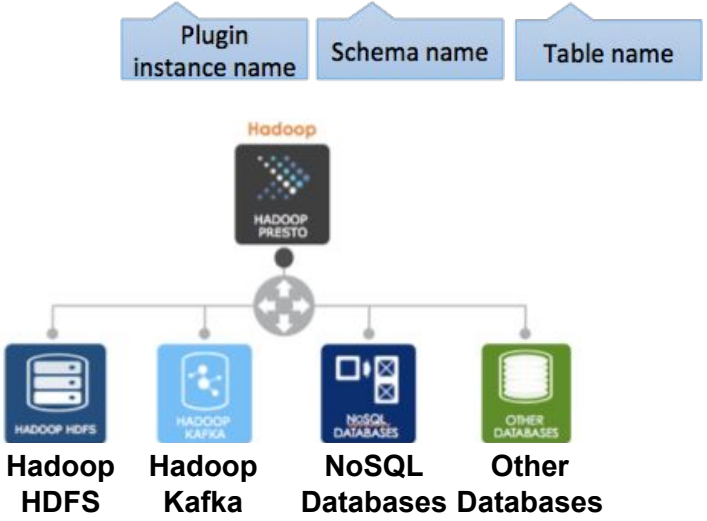


presto as a universal SQL Big Data service

- **SQL on-anything**
  - Massively parallel processing (MPP)
- **Query engine for multiple datastores/DBs**
- **Low latency SQL queries** (<100ms start up)
- **Platform agnostic** - can run anywhere
- Typically runs on top of the HDFS data

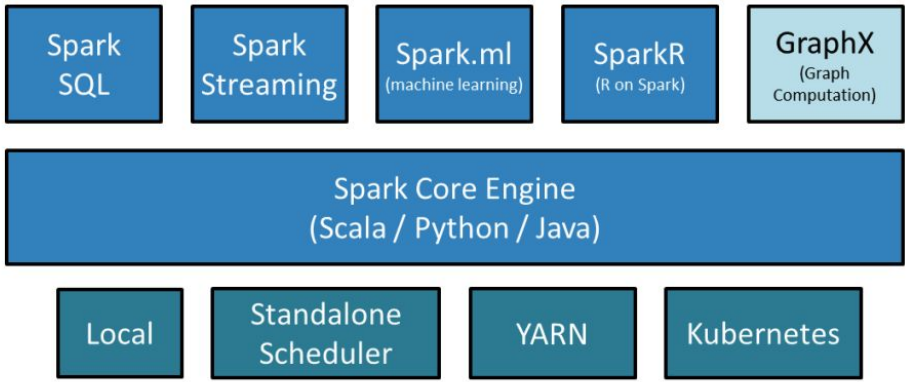


- **Easy-to-use SQL**
- **Multiple connectors**
- **CLI/JDBC/Web interfaces**
- *select * from hive_analytix.schema1.table1*

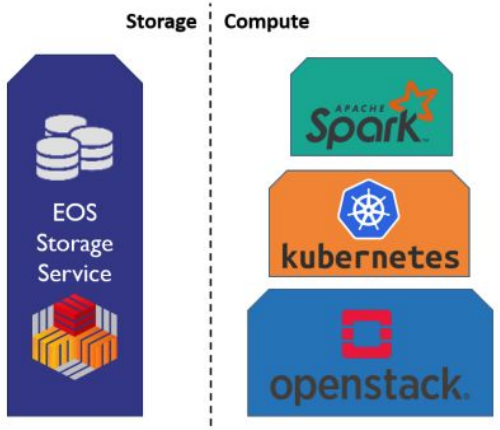


Spark 4.0 and Spark on K8S

- **Spark 4.0.0 highlights**
  - Support Java 17/21, HDP 3.3+, ZK 3.9+…
  - Scala 2.13 (2.12 won't work)
  - Drop mesos support
  - RocksDB as default Shuffle service db
  - Hive 4.0 metastore
  - K8S client upgrade



- **Spark on K8S** (300 cores)
  - Offered from SWAN service
  - May not be suitable for stable workloads
    - Monit, security, nxcals…
  - Suitable for ad-hoc workloads
  - External storage integration (EOS/S3)



We would like to get your feedback on this!

Vote on your preferred technologies

https://www.menti.com/alrxrgo96iqq

Access code: **4620 5218**

Reviewed technologies

- **Apache Ranger**
  - framework to monitor and manage comprehensive data security across the Hadoop platform
  - https://ranger.apache.org
- **Apache Ozone**
  - highly scalable, distributed storage for Analytics, Big data and Cloud Native applications
  - https://ozone.apache.org
- **HDFS Erasure Coding**
  - compress data with less storage overhead and same fault tolerance as the standard replication
  - https://hadoop.apache.org/.../HDFSErasureCoding.html
- **Apache YuniKorn**
  - a light-weight, universal resource scheduler for container orchestrator systems
  - https://yunikorn.apache.org

Reviewed technologies

- **Yarn Docker containers**
  - Docker containers provide a custom execution environment for applications
  - https://hadoop.apache.org/docs/.../DockerContainers.html
- **Apache Iceberg**
  - Table format for large-scale analytics
  - https://iceberg.apache.org
- **Apache Presto**
  - Massively Parallel Processing (MPP) SQL query engine for multiple datasources/DBs
  - https://prestodb.io
- **Spark 4.0 and Spark on K8S**
  - Prepare Spark v4.0.0: SPARK-44111
  - Spark on K8S: https://spark.apache.org/docs/latest/running-on-kubernetes.html

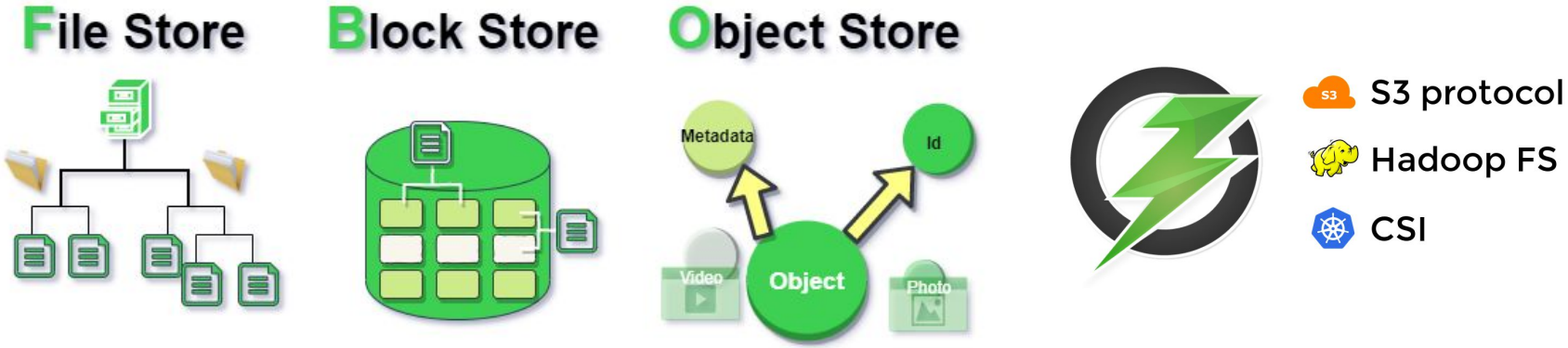
Trends and future

of Hadoop Big Data

Towards object stores

Single-name server file systems have their limitation

- Solution 1: scaling out namespaces by incorporating FS federation
- Solution 2: putting FS namespaces on a distributed cluster of DB servers
- Solution 3: using object stores instead of FSs



Towards Kubernetes

- **YARN is supported by multiple frameworks and works best with HDFS**
  - Lacks some agility in running any resource intensive jobs
  - Difficult to execute jobs with external non-Java deps (like Python)
  - Old-fashion web interfaces
- **Possible options:**
  1. **Yarn v3** offers submitting jobs with **docker images** (custom environment)
  2. run **Spark on K8S** but lacks native support for non-Spark jobs
  3. Apache YuniKorn as a universal resource scheduler in K8S
- **K8S is getting more traction in the Big Data world:**
  - Most of the Hadoop stack is K8S compatible or has corresponding successors

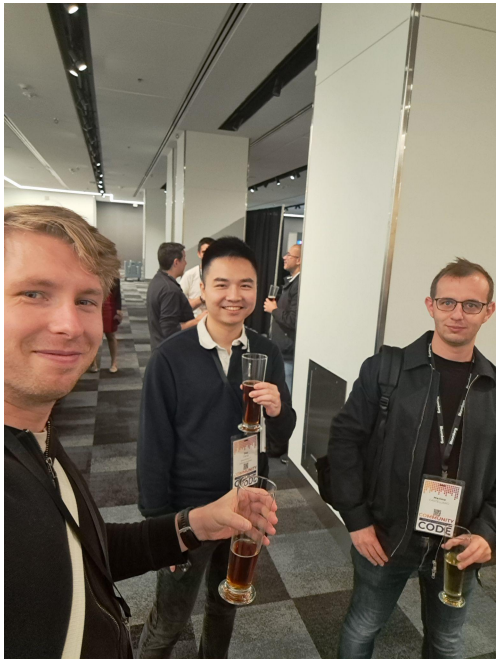
On-premise vs cloud

- **Maintaining clusters for data computing is expensive**
  - Space, power/cooling, people to maintain infra., …
- **A small/medium-sized company cannot afford it**
- **Solution: move to cloud**
  - Pay for what is used, not for what you have
  - Vendors made a lot of effort to support big data systems in the cloud
    - offer HDFS/Yarn, Spark on K8S, object store integrations, …
  - Cloud compromises data locality with scaling agility!
- **Big companies with DCs or with highly confidential data tend to run their own infr.**
  - typically cheaper and more secure

Ideas from the community

- Spark on K8S with YuniKorn
- Iceberg for large-scale analytics (Hive++)
- HBase support for K8S coming soon
- Apache Ozone is gaining traction
- HBase has some competition: Cassandra, Accumulo (backed by NASA), Pinot, TSDB…

- **What are your ideas? :-)**



More about Hadoop @ CERN

Snow service

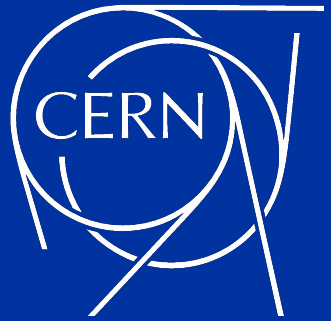
[Service-now Hadoop-Service]

Hadoop Docs

https://hadoop.docs.cern.ch

Mattermost

https://mattermost.web.cern.ch/it-dep/channels/it-hadoop-service



Thank you for your attention

Emil Kleszcz @ CERN

04 Dec 2023