Big Data Trends and HDFS Evolution

Sanjay Radia
Founder & Architect
Hortonworks Inc

Hello

- **Founder, Hortonworks**

- **Part of the Hadoop team at Yahoo! since 2007**
  - ***Chief Architect*** of ***Hadoop Core*** at Yahoo!
  - Long time Apache Hadoop ***PMC*** and ***Committer***
  - Designed and developed several key Hadoop features

- **Prior**
  - Data center automation, virtualization, Java, HA, OSs, File Systems (Startup, Sun Microsystems, …)
  - Ph.D., University of Waterloo

Overview

- **Hadoop (HDFS, Yarn)**

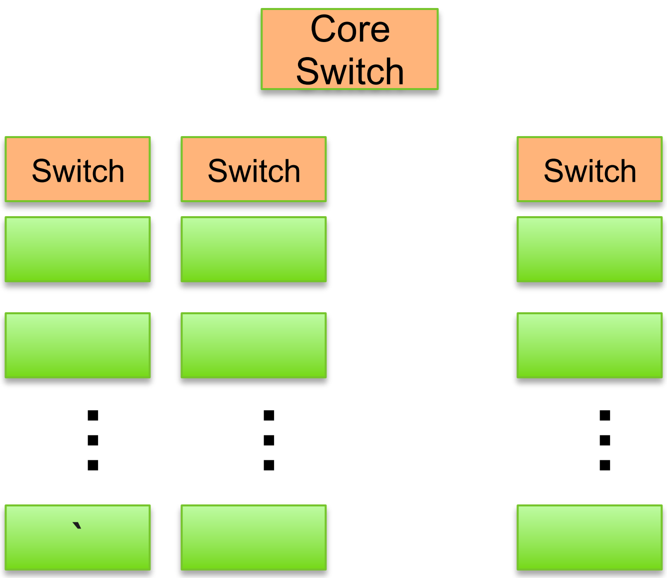
- **Trends**
  - Hardware
  - Applications
  - Deployment Model

- **Addressing the trends**
  - Storage architecture changes
  - Archival
  - Microservers
  - Public Clouds
  - Private Clouds and VMs

Hadoop: Scalable, Reliable, Manageable

Scale IO, Storage, CPU

- Add ***commodity*** servers & ***JBODs***
- 6K nodes in cluster, 120PB



- Fault Tolerant & Easy management
  - Built in redundancy
  - Tolerate disk and node failures
  - Automatically manage addition/ removal of nodes
  - One operator per 3K node!!
- Storage server used for computation
  - Move computation to data
  - High-bandwidth network access to data via Ethernet
- Scalable file system (Not a SAN)
  - Read, Write, rename, append

    No random writes
- YARN: General Execution Engine
  - Not just MapReduce
  - SQL, ETL, Streaming, ML, Services

Simplicity of design

why a small team could build such a large system in the first place

Trends

Cloud usage

- Public
- Private

Applications

- Streaming
- ML, …

Platform Change

- Network Bandwidth
- Cores
- Memory sizes
- Flash storage
- Disk density
- VMs
- Micro-servers

Usage Patterns

- Batch to Interactive
- Data sizes

Heterogeneous Storage

Architectural Change at lower level
that changes how HDFS views storage

Heterogeneous Storage

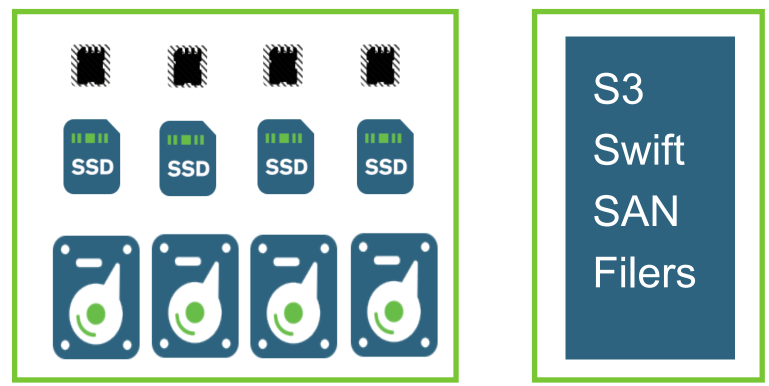
Original HDFS Architecture

- DataNode is a **single storage unit**
- Storage is uniform - Only storage type Disk
- Storage types hidden from the file system

All disks as a single storage

New Architecture

- DataNode is a **collection of storages**
- Storage type exposed to NN and Clients
- Support different types of storages
  - Disk, SSDs, Memory
- Support external storages
  - S3, OpenStack Swift



Collection of tiered storages

Heterogeneous Storage

- **Support new hardware/storage types**
  - DataNodes configured with per-volume storage types
  - Block report per storage – better scalability for large and denser drives
- **Mixed storage type support and Automatic Tiering**
  - Some replicas in SSD and some on disk
  - Some replicas on disk in HDFS and some on S3/OpenStack Swift
  - Move across tiers based on usage
- **Memory is important storage tier**
  - Memory caching for hot files
  - Memory for intermediate files
- **Multi-tenant support – quotas per storage type**
- **APIs to let HDFS manage the data migration across tiers**
  - ***HDFS evolves towards a Data-Fabric***

Archival

- **Hadoop makes it cost effective to store data for several years**
  - Only a portion of the data is hot/warm, but the rest is still online
- **Two models**
  1. Separate cluster with low-power nodes and regular disks
     - But Spindles are the bottleneck
     - Better of putting those spindles on active clusters
  2. Lower-cost slower disks in the same cluster
     - Slower disks become another tier
     - Move cold data to slower disks
       - All replicas
       - One or two of the replicas (if data is lukewarm)
  - Hybrid – some special nodes that are Disk intensive and low powered CPU
- **Other:**
  - Erasure codes for lukewarm and cold data

Micro-servers

- **Large number of small servers**
  - Lower power and unused servers can be shutdown
  - Footprint
  - Storage - Disks or partitions can be moved from one to another
- **Challenges for HDFS/Hadoop**
  - Allocating entire spindles is better
    - Allocating partitions may not be most suitable because of seek contention across DataNodes that share the disk seeks
  - Shutting down a drive does not make sense
  - Some challenges in moving a drive to another “micro DataNode”
    - The OS on the other drive needs to take it over
    - Recent changes due to Heterogeneous storage helps here
      - But multiple replicas on same node will need to be handled
      - Replica allocation will have to take this into account

Public Clouds

Public Clouds – A Spectrum

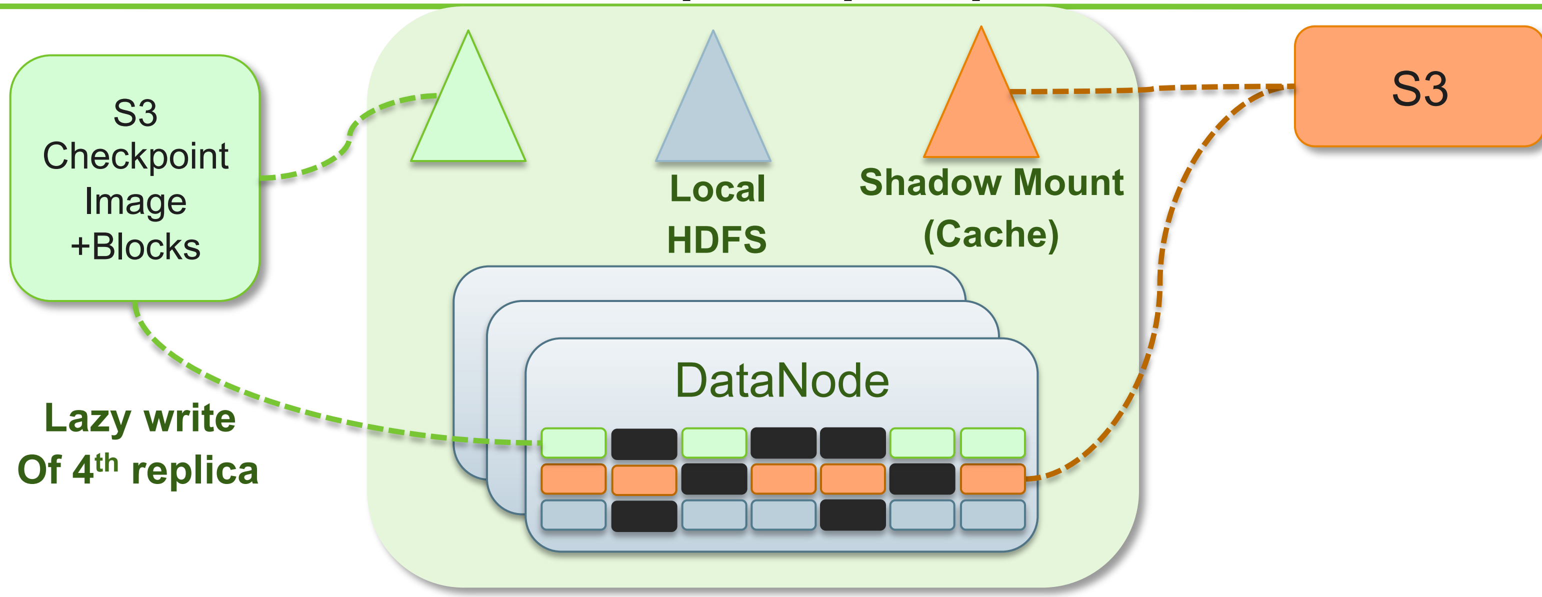
Virtual Machines	Hosted Deployed Managed Hadoop Clusters		Hadoop App as a Service
Run/Manage Your Hadoop Cluster On VMs	*Cluster deployed on VMs or private hosts with easy Cluster Management & Job Management tools*	*Managed "virtual" cluster that uses a shared Hadoop. Just submit jobs/apps*	*Just submit your job/app*
MS Azure AWS EC2 Rackspace Cloud	MS HDInsight (VMs) Rackspace (VMs or Host in cage) ATT	Altiscale	AWS EMR
You Spin up/down cluster Data on external store (S3, Azure storage) across Cluster lifecycles	Hosted: Cluster/Data persists	Shared HDFS but Isolated Namespaces Shared Yarn but isolation via Containers or VMs	Automatic Spin up/down Data on external store (S3 Azure storage) across Cluster lifecycles,

Cluster Lifecycle and Data Persistence

- **For a few: the Cluster and HDFS data persists**
  - Rackspace (Hosted and managed private cluster)
  - Altiscale

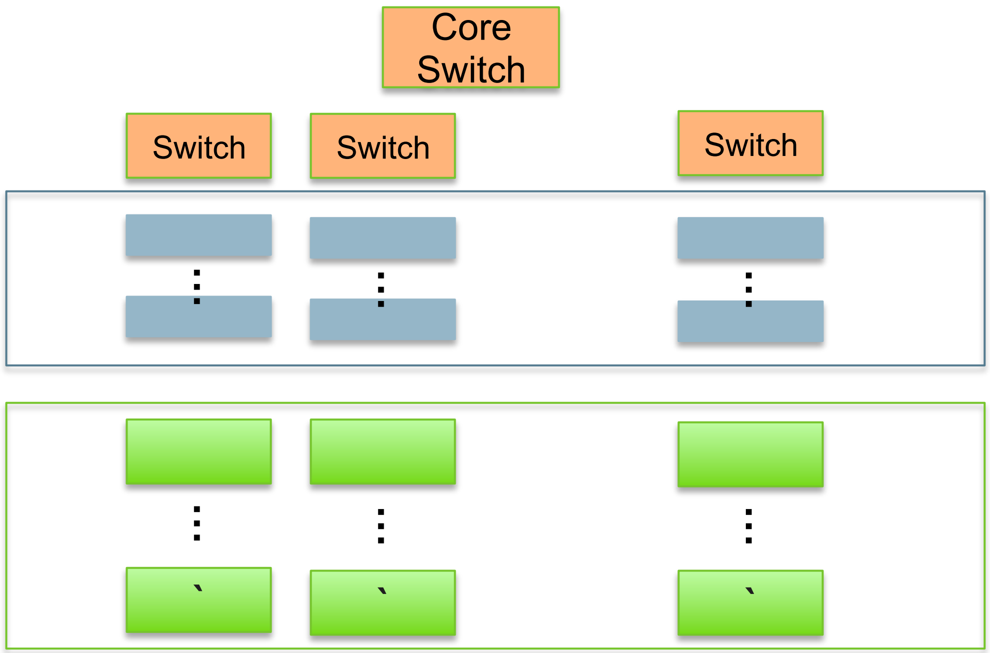
- **For most: clusters spun up and down**
  - Main challenge: need to persist data across cluster lifecycle
    - Local data disappears when cluster is spun down
  - External K-V store
    - Access remote storage (slow)
    - Copy, then process, process, process, spin down (e.g. Netflix)
  - How can we do better?
    - Interleaved storage cluster across rack (an example later)
    - Shadow HDFS that caches K-V Store – great for reading
    - Use the K-V store as 4th lazy HDFS replica – works for writing

Use of External Storage when cluster is Spinup/Spindown



- **Shadow mount external storage (S3) – great for reading**
  - Cache one or two replicas at mount time
    - when missing, fetch from source
  - Hadoop apps access cached S3 data on local HDFS
- **Use S3 to store blocks and check-pointed image – works for writes**
  - Write lazy of 4th replica
  - When cluster is spun down, all block and image must be flushed to S3

An Interesting "Hadoop as service" configuration

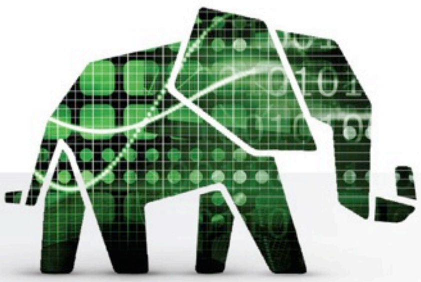


Storage HDFS cluster
- Stores data that persists when compute clusters are spun down

Transient Hadoop cluster
- Clusters (VMs) are spun up and down
- Have transient local storage

- **HDFS Cluster for data external to the compute cluster**
  - but stripped to allow rack level access
- **Persists when the dynamic customer clusters are spun-down**
- **Much better performance than external S3…**

VMs and Hadoop in Private Clouds



Virtual Machines and Hadoop

- **Traditional Motivation for VMs**
  - Easier to install and manage
  - Elasticity – expand and shrink service capacity
    - Relies on expensive shared storage so that data can be accessed from any VM
  - Infrastructure consolidation and improved utilization

- **Some of the VM benefits require shared storage**
  - But external shared storage (SAN/NAS) does not make sense for Hadoop

- **Potential Motivation for Hadoop on VMs**
  - Public clouds ***mostly*** use VMs for isolation reasons
  - Motivation for VMs for private clouds
    - Test, dev clusters-on-fly to allow independent version and cluster management
    - Share infrastructure with other non-Hadoop VMs
    - Production clusters …
      - Elasticity ?
      - Multi-tenancy isolation?

Virtual Machines and Hadoop

- **Elasticity in Hadoop via VMs**
  - Elastic DataNodes? - ***Not a good idea –lots of data to be copied***
  - Elastic Compute Nodes – ***works but some challenges***
    - Need to allocate local disk partition for shuffle data
    - VM per-tenant will fragment your local storage
      - This can be fixed by moving tmp storage to HDFS –will be enabled in the future
    - Local read optimization (short circuit read) is not available to VM

- **Resource management & Isolation excellent in Hadoop**
  - Yarn has flexible resource model (CPU, Memory, others coming)
    - It manages across the nodes
  - Yarn's Capacity scheduler gives capacity guarantees
  - Isolation supported via Linux Containers
  - Docker helps with image management

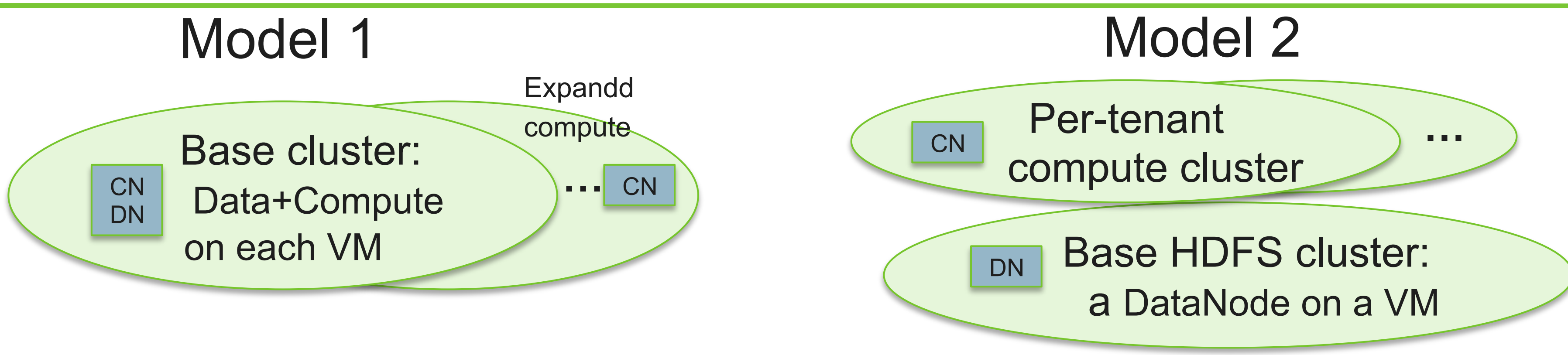
Guideline for VMs for Hadoop

- **VMs for Test, Dev, Poc, Staging Clusters works well**
  - Can share Host with non-Hadoop VMs
  - For such use cases having independent clusters is fine
    - Do not have lots of data or storage fragmentation is less concern
    - Independent clusters allow each to have a different Hadoop version
  - Some customers: a native shared Hadoop cluster for testing apps

- **Production Hadoop**
  - VMs *generally* less attractive

    But if you do, *consider motivation, and configure accordingly (next slide)*
  - Hadoop has built-in
    - elasticity for CPU and Storage
    - resource management
    - capacity guarantees for multi-tenancy
    - isolation for multi-tenancy

Configurations for Hadoop VMs



- Share the cluster *unless* you need per-tenant Hadoop-version or more isolation

- **Model 1 is generally better**
  - ComputeNode/DataNode shortcut optimization
  - Less storage fragmentation (intermediate shuffle data)
  - Share cluster: Hadoop has excellent Tenant/Resource isolation
  - Use Model 2 if you need need additional tenant isolation but shared data

Hortonworks

Summary

- **Hadoop Virtualizes the Cluster's HW Resources across Tenants**
- **Hadoop for the Public Cloud**
  - Most based on VMs due to strong complete isolation requirement
    - Altiscale, Rackspace offer additional models
  - Main challenge is data persistence as clusters are spun up/down
    - We describe best practices and what is coming
- **Hadoop for the Private Cloud**
  - For test, dev, Poc clusters
    - Both Hadoop on raw machines and VMs are a good idea
      - VMs allow "cluster-on-the-fly" plus ability to control version for each user
  - For production clusters
    - Hadoop natively provides excellent elasticity, resource management and multi-tenancy
    - If you must use VMs
      - Understand your motivations well and configure accordingly
  - For VMs, follow the best practices we provided

Closing Remarks

- **Storage is fundamental to Big Data**
  - HDFS: Central storage for an enterprise's data – Data Lake
  - Buil your Analytics, ETL, etc. around the centrally stored data
- **HDFS provides a proven, rock-solid file system**
  - Some create FUD around HDFS but …
  - Proven reliability, scalability
  - Has the key enterprise features
- **HDFS has evolved and continues to evolve**
  - Improvements on features, scalability, performance will continue
  - Evolve to a ***Data-fabric*** rather than mere file storage
    - Tiering of storage type: memory, flash, disks, archival, S3
    - Data moves across tiers according to application needs
    - Hooks for upper layers to influence how Hadoop handles data

Hortonworks

Q & A

Thank You