
KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2019

Embracing Big Data Workload in Cloud Native Environment with Data Locality

Sammi Chen & Xiaoyu Yao

About us

- Sammi Chen – Software engineer from Tencent Cloud, Apache Hadoop Committer and PMC, working on Apache Hadoop HDFS/Ozone.

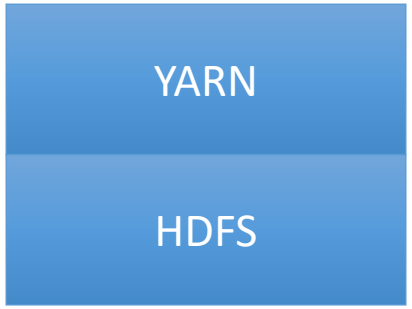
- Xiaoyu Yao –Software engineer from Cloudera, Apache Hadoop Committer and PMC, working on Apache Hadoop/HDFS/Ozone.

Outline

- Big data evolution in cloud native environment
- Data Locality and why it matters
- Data locality in big data storage
- Locality aware big data storage in Kubernetes
- Evaluation

Big Data Evolution in Cloud Environment

- Co-located Compute and Storage



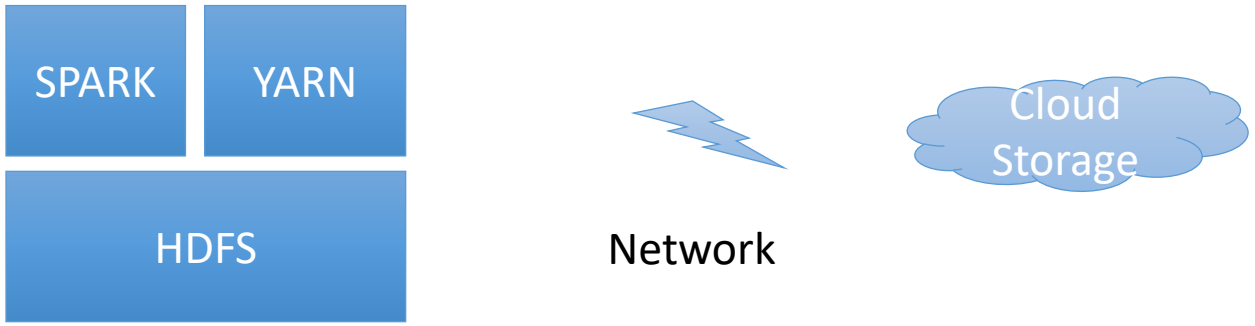
  - Pros
    - Fast storage access with locality
    - Less network traffic
    - Cost-effective for I/O intensive OLAP workloads (MapReduce/Hive/Impala)
  - Cons
    - Limited Elasticity: Requirements for Storage nodes and compute node are different.
    - Elastically scale storage node with compute node is not cost-effective.

Big Data Evolution in Cloud Environment

- Separation of Compute and Storage
  - Pros
    - Elastically scale compute independent of storage
    - Cost-effective for compute-intensive workload(ML)
  - Cons
    - Lose storage locality
    - More network traffic for storage access
    - More CPU cycle (e.g., Erasure Coding)

Big Data Evolution in Cloud Environment

- Hybrid Cloud (On-prem + public cloud)
  - Pros
    - Support locality for I/O intensive workload on-prem
    - Allow agile access, e.g., ML on multi-cloud
  - Cons
    - Cost
    - Compatibility



Challenges of Cloud Native Env for Big Data

Scheduler

- Optimize resource utilization.

Storage

- Optimize external storage access.

Networking

- Optimize bandwidth usage.

Locality in Big Data

- What is Locality?

  Local Node: Data local to the compute node

  Local Rack: Data in the same rack with the compute node

  Local DC: Data in different rack/zone but closer to the compute node

Locality in Big Data - Benefit

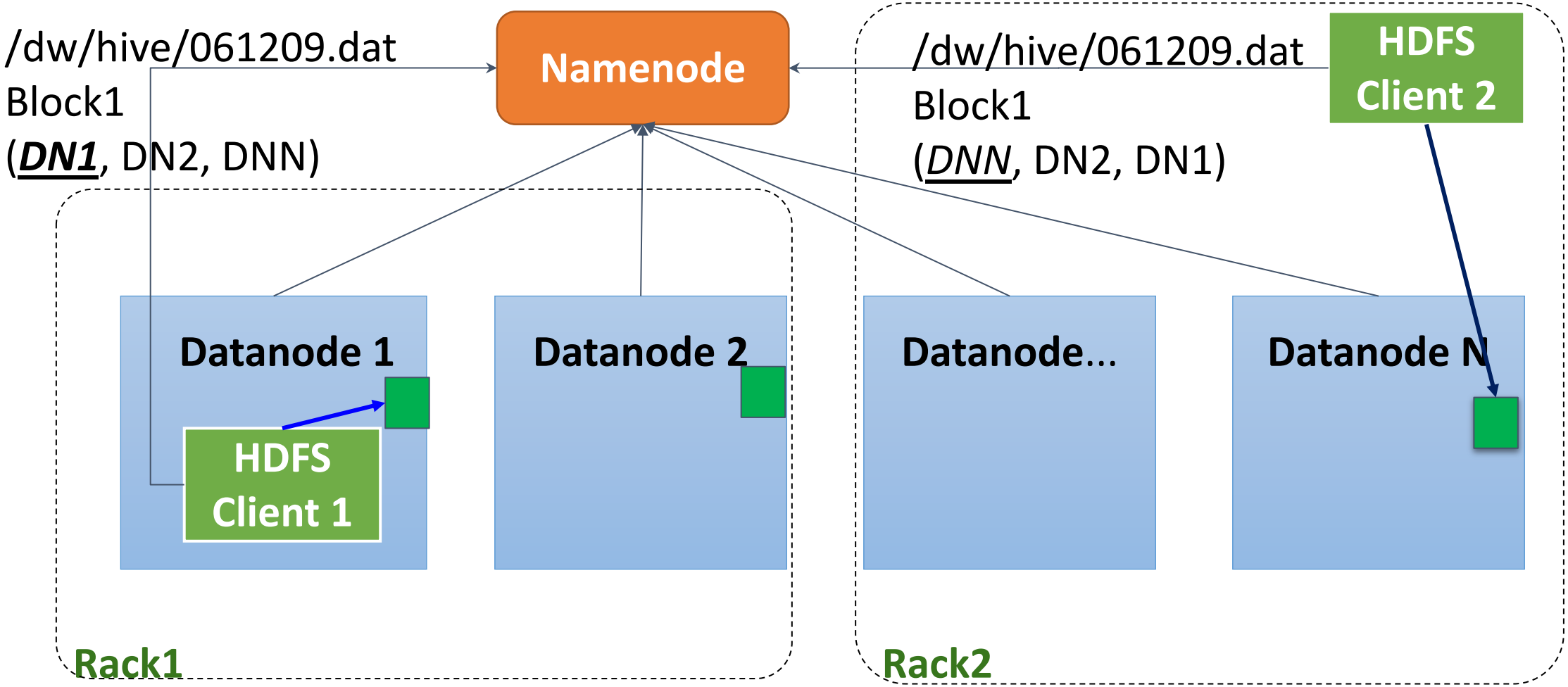
- Higher throughput
- Less network traffic
- Fast job execution
- Better cluster utilization

Locality in Big Data – Storage (HDFS)

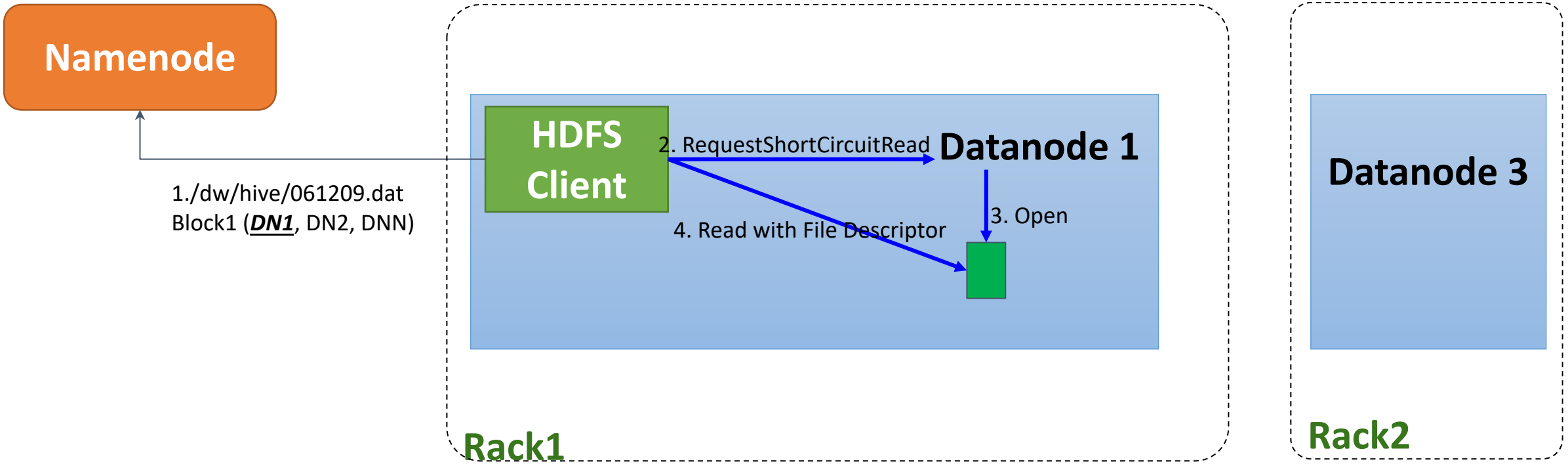
Apache Hadoop HDFS

- Scalable Distributed File System
  - Fast file system metadata access (200K ops/s)
  - Hundreds PBs in capacity
  - Thousands of nodes per cluster
  - Scale horizontally
  - Strong consistency
  - Resilient to failures
  - …

Hadoop HDFS Locality
Rack aware access



Hadoop HDFS Locality – Short Circuit Read



Apache HDFS in Kubernetes

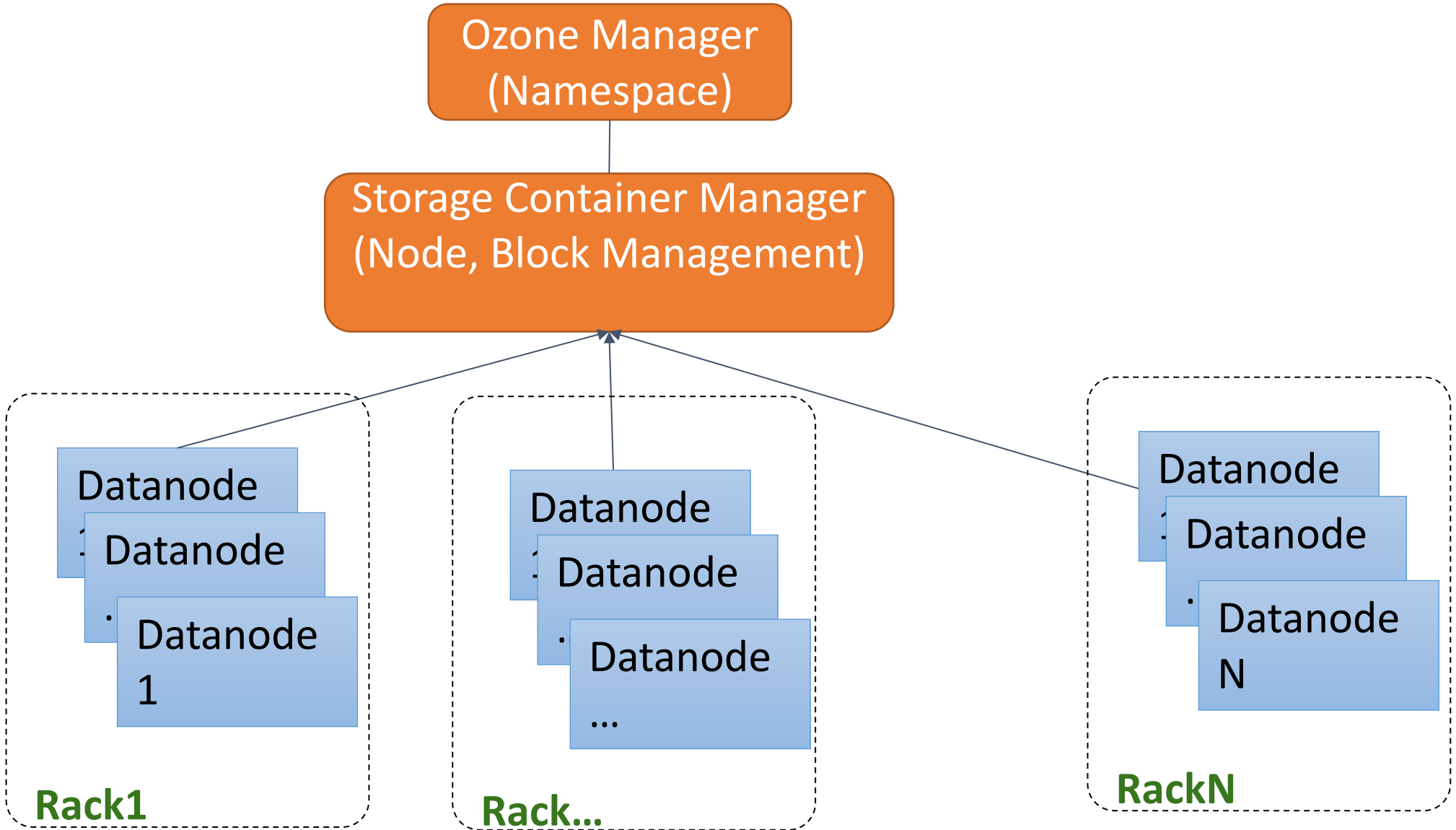
- Challenges
  - Monolithic Namenode
    - Small Files problem
      - 300 million+ files need special GC tuning
      - Take long time to upgrade/restart

- Opportunities
  - Cloud native storage that support
    - Existing big data workload: Analytic/IoT/Streaming, etc.
    - Upgradable from existing HDFS clusters with hundreds or thousands of nodes.

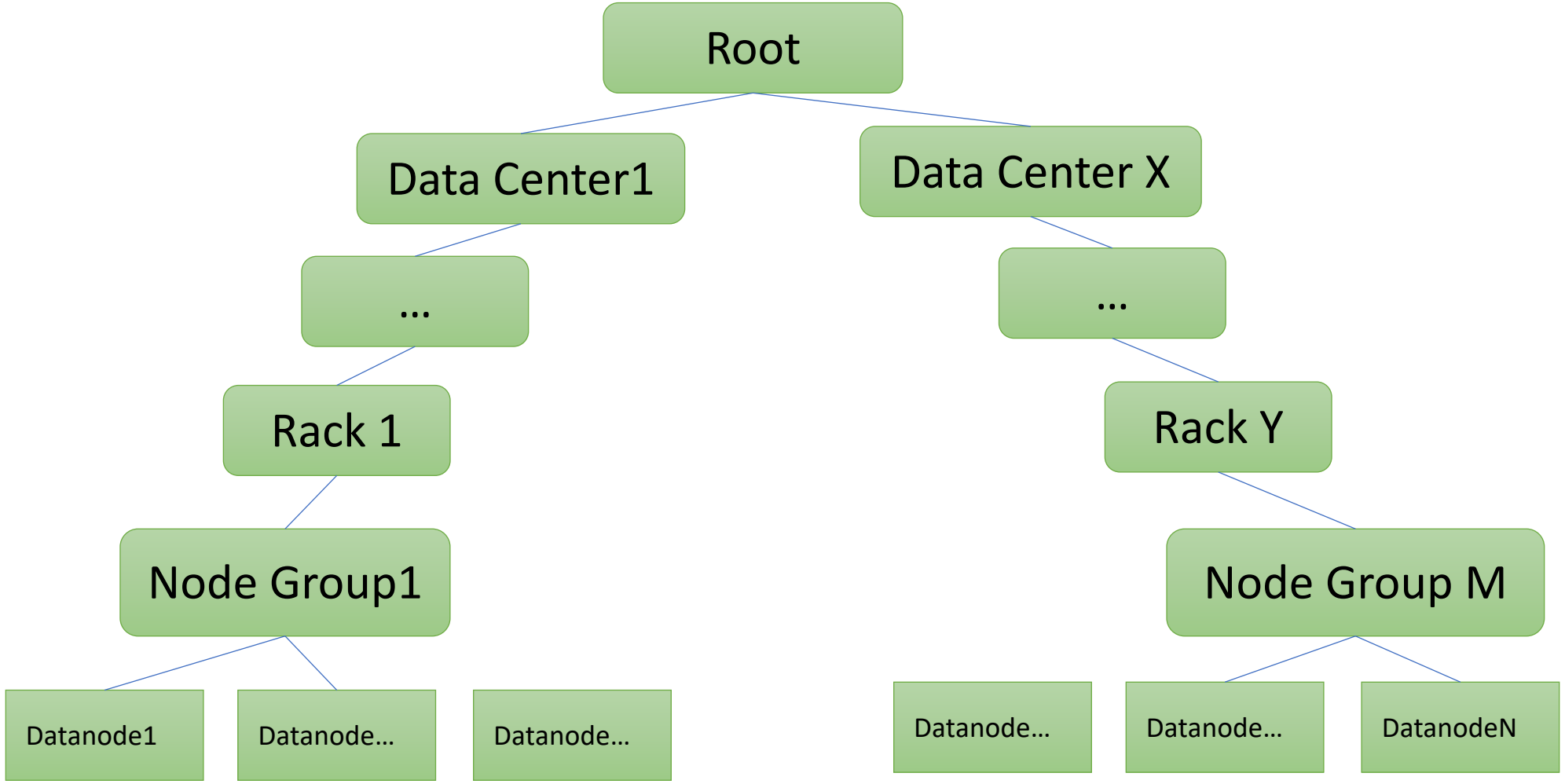
Locality in Big Data – Storage (Apache Ozone)

- Scalable, redundant, and distributed object store
- Scaling to billions of objects of varying sizes.
- Support topology aware data placement and access.
- Support Kubernetes deployment.
- Support S3 access
- Support in-place upgrade from HDFS
- …

Apache Ozone Overview



Apache Ozone Topology



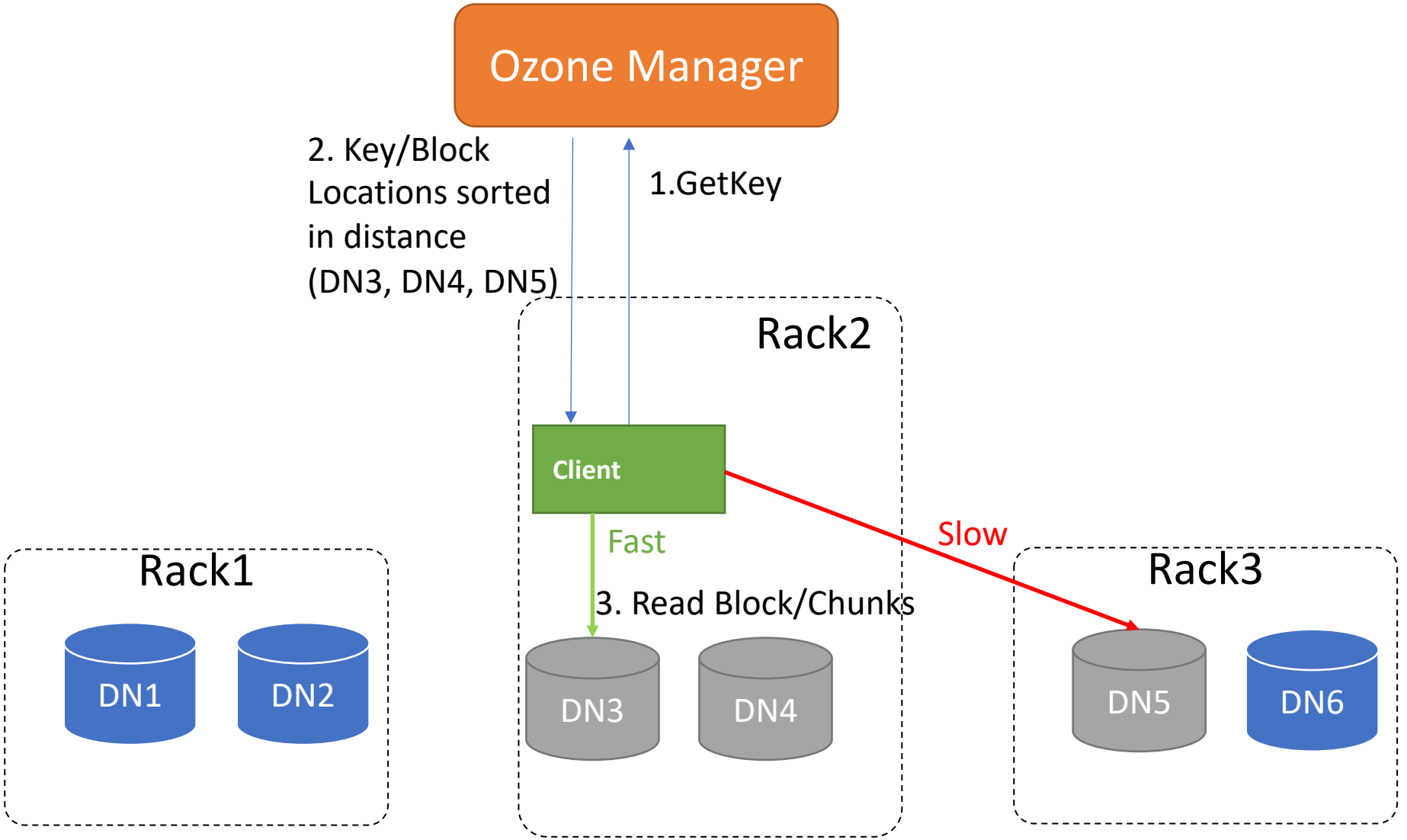
Apache Ozone Locality

- Highly customizable topology schema
  - /DataCenter/NodeGroup/Rack/Datanode.
- Topology aware access policy
  - Ozone manager return topology ordered datanodes list for client
  - Client access block/chunks of the objects to the closest datanodes
- Topology aware placement policy
  - Trade-off between reliability and performance.

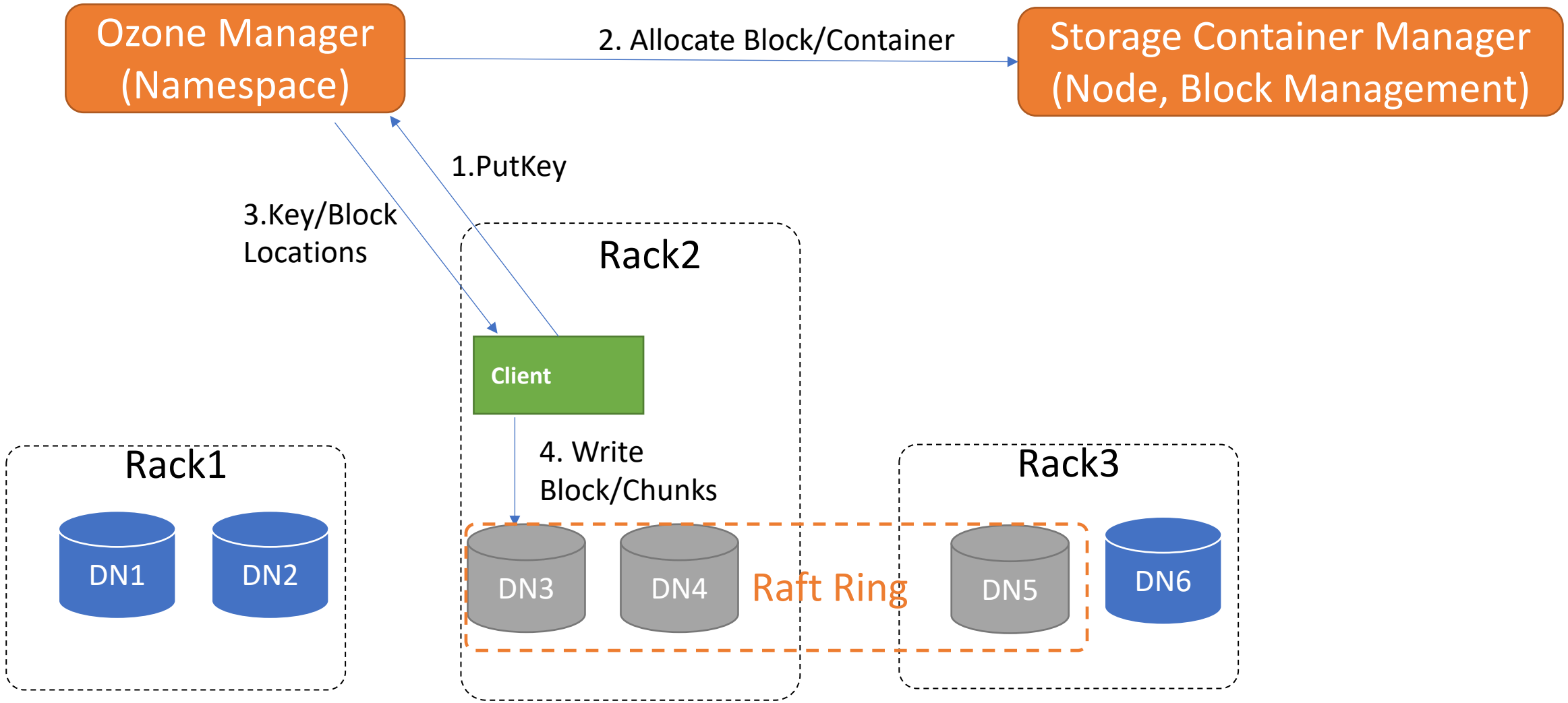
Customize Ozone Topology


```
type: ROOT

# Layer name
defaultName: root

# Sub layer
# The sub layer property defines as a list which can reflect a node tree, though
# in schema template it always has only one child.
sublayer:
  -
    cost: 1
    prefix: dc
    defaultName: datacenter
    type: INNER_NODE
    sublayer:
      -
        cost: 1
        prefix: rack
        defaultName: rack
        type: INNER_NODE
        sublayer:
          -
            cost: 1
            prefix: ng
            defaultName: nodegroup
            type: INNER_NODE
            sublayer:
              -
                defaultName: node
                type: LEAF_NODE
                prefix: node
```

Apache Ozone Topology aware Read



Apache Ozone Topology aware Write



Apache Ozone in Kubernetes

- Stateful Set
  - Ozone Manager
  - Storage Container Manager
- Daemon Set
  - Datanode
    - Datanode -> Pod Mapping
    - Local Persistent Volume
    - Node topology mapping
- Ozone Client
  - Ozone client running in Pod must use K8s node IP to

Apache Ozone in Kubernetes

- Ozone S3 Gateway
  - Access Ozone with S3 API
  - Horizontally scale with RepicaSet behind proxy
- Ozone CSI driver
  - Mount Ozone S3 bucket as CSI volume
- Ozone Operator
  - Rook integration

Locality in Big Data - Scheduling

- Apache YARN
  - Support locality aware task scheduling
- Apache SPARK
  - Elastic executor provision based via K8s with locality awareness

Locality Support from Kubernetes

- Topology aware scheduler
  - Zone based
  - label-based specification (node Selectors and node Affinity)
- Topology-aware volume provisioning
  - Pre-provision volume
  - Dynamic provisioning of volume
- Better support from software defined storage like Apache HDFS/Ozone
  - HostPath volume -> Local Persistent Volume

Evaluation

Spark Tera Sort with Ozone in K8s w/wo topology-awareness

- Finish time
- Network traffic
- Remote Read vs Local Read

Q & A

Thanks!