Automatic-Hot HA for HDFS NameNode

Konstantin V Shvachko *EBay*

Ari Flink *Cisco*

Timothy Coulter *Aisle Five*

November 11, 2011

About Authors

- *Konstantin Shvachko*
  - Hadoop Architect, eBay;   Hadoop Committer
- *Ari Flink*
  - Operations Architect, Cisco
- *Tim Coulter*
  - Cloud Architect, Aisle Five Consulting
- Thanks to *Rajesh Balasa*

Agenda

- What is HA
  - Why HA is important for Hadoop
  - And why it is not
- HDFS architecture principles
  - How HDFS HA is different from traditional
- HA design choices
- One simple design: Details in HDFS-2064
- Comparisons
- Hadoop 0.22 progress

Why High Availability is Important?

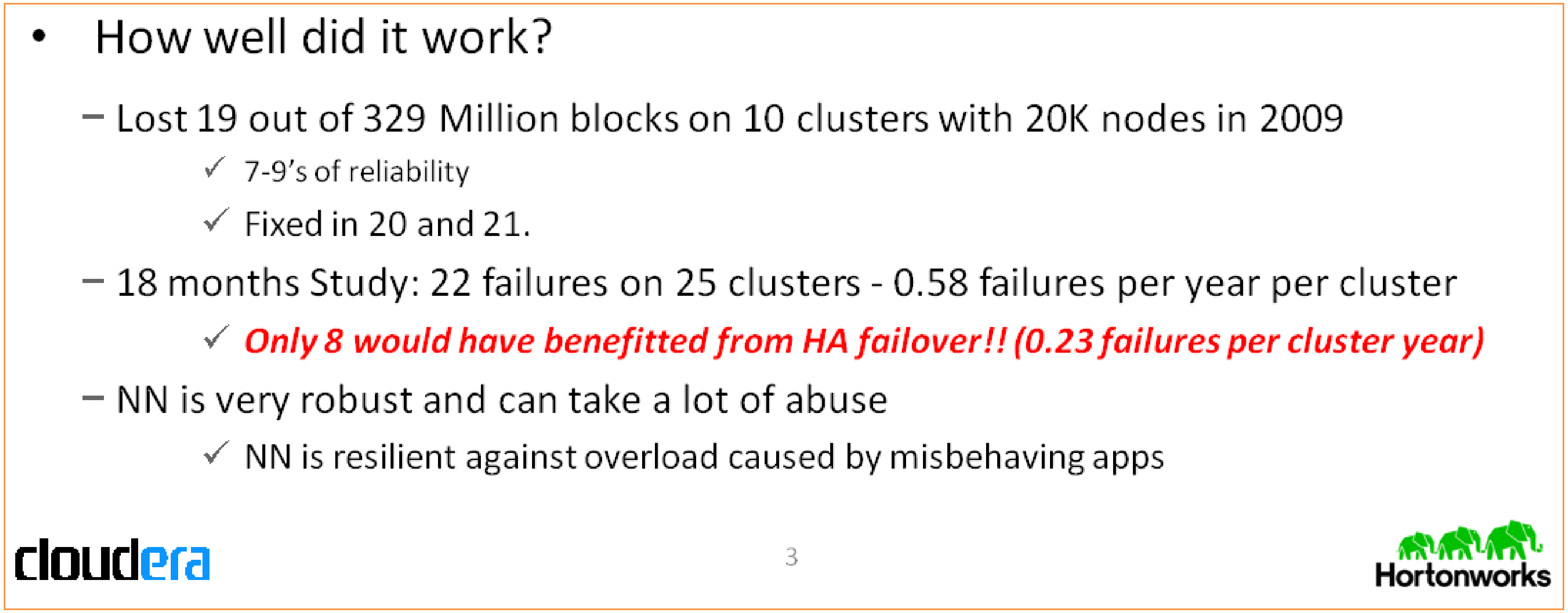
- Nothing is perfect:
  - Applications and servers crash
  - Avoid downtime
- Conventional for traditional DB and enterprise storage systems
- Industry standard requirement

And Why it is Not?

- Scheduled downtime dominates Unscheduled
  - OS maintenance
  - Configuration changes
- Other reasons for Unscheduled Downtime
  - Full GC
  - System bugs: HDFS and the stack above
- Pretty reliable

How Reliable is HDFS Today?

From Hadoop World Presentation By Sanjay, Suresh, Aaron



- How well did it work?
  - Lost 19 out of 329 Million blocks on 10 clusters with 20K nodes in 2009
    - ✓ 7-9's of reliability
    - ✓ Fixed in 20 and 21.
  - 18 months Study: 22 failures on 25 clusters - 0.58 failures per year per cluster
    - ✓ *Only 8 would have benefitted from HA failover!! (0.23 failures per cluster year)*
  - NN is very robust and can take a lot of abuse
    - ✓ NN is resilient against overload caused by misbehaving apps

3

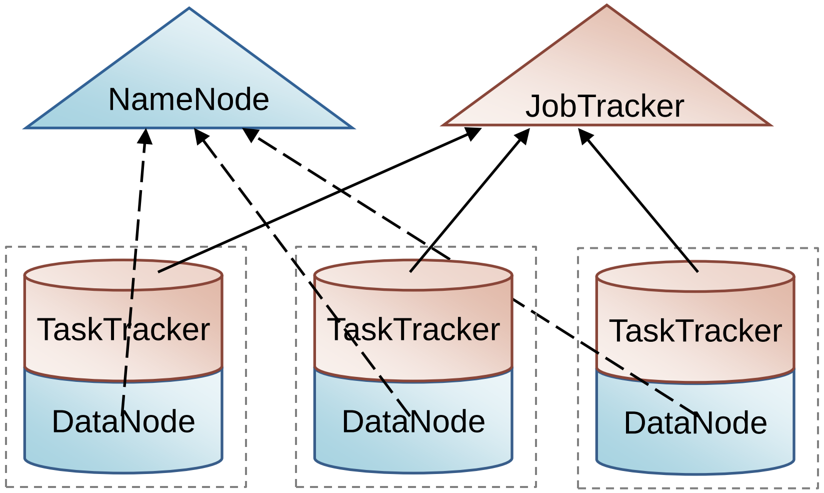
- Internet industry HA standard 99.94% (5.26 h/y)

Fable

- Need to eliminate SPOF
  - Enterprise storage standard
- Minimize the effort
  - No immediate business impact

Hadoop Cluster Components

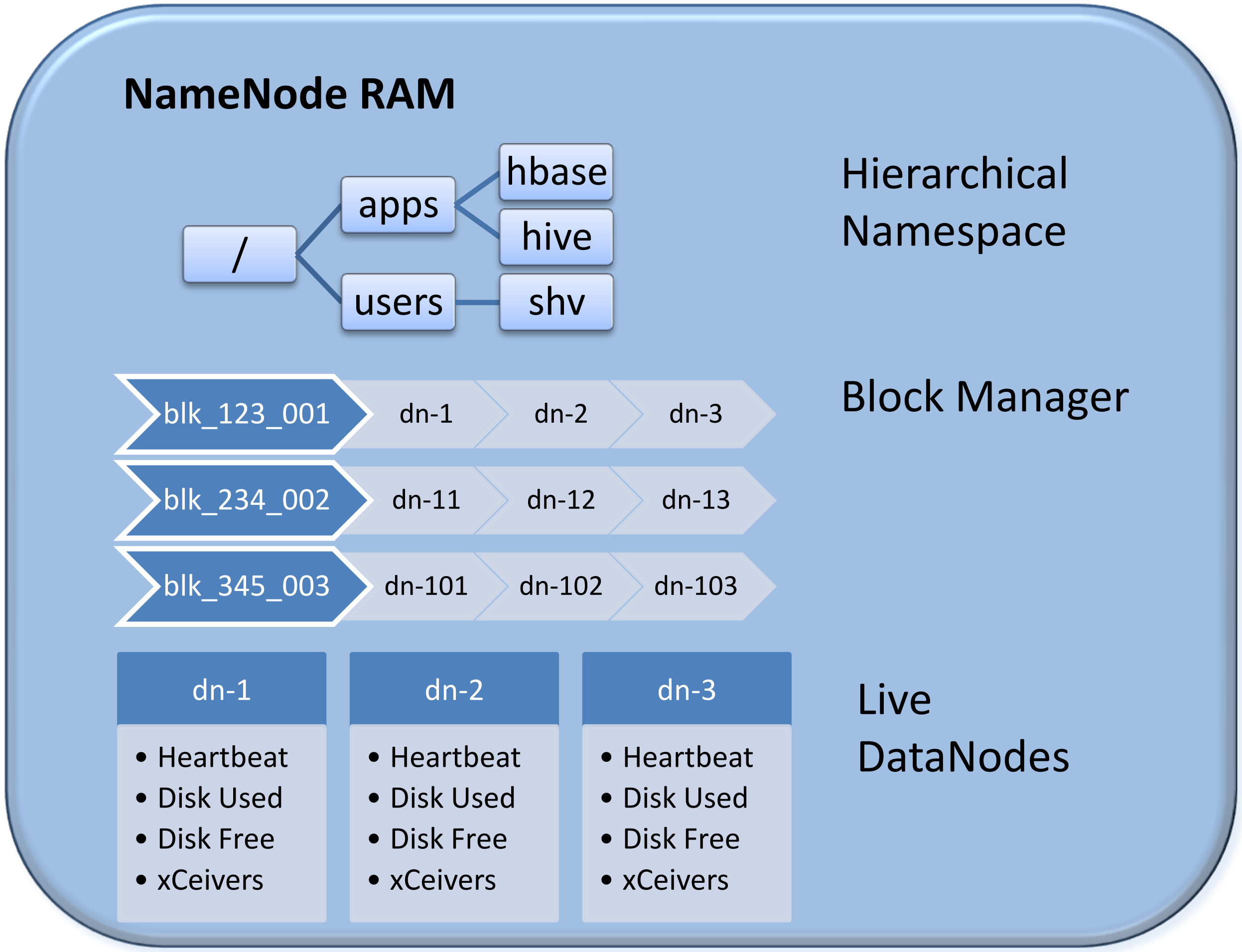
- HDFS – a distributed file system
  - NameNode – namespace and block management
  - DataNodes – block replica container
- MapReduce – a framework for distributed computations
  - JobTracker – job scheduling, resource management, lifecycle coordination
  - TaskTracker – task execution module
-



HDFS Overview

- The namespace is a hierarchy of files and directories
- Files are divided into data blocks (typically 128 MB)
- Namespace (metadata) is decoupled from data
  - Fast namespace operations
  - not slowed down by Data streaming
- Single NameNode keeps the entire name space in RAM
- DataNodes store block replicas as files on local drives
- Blocks are replicated on three DataNodes
  - for redundancy and availability

NameNode Transient State



NameNode Persistent State

- The durability of the name space is maintained by a write-ahead ***journal*** and ***checkpoints***
  - Journal transactions are persisted into ***edits*** file before replying to the client
  - Checkpoints are periodically written to ***fsimage*** file Handled by Checkpointer
  - Block locations discovered from DataNodes during startup via **block reports**. Not persisted on NameNode
- Types of persistent storage devices
  - Local hard drive
  - Remote drive or NFS filer
  - BackupNode

DataNodes

- DataNodes inform NameNode about itself
  - Heartbeats (3 sec)
  - Block reports (1 hour)
  - Replica received
- NameNode replies to DNs with control commands
  - Delete block replica
  - Replicate block to another DataNode
  - Re-register, send emergency block report, shutdown, etc.

NameNode HA Challenge

- Naïve Approach
  - Start new NameNode on the spare host, when the primary NameNode dies:
  - Use LinuxHA or VCS
- Not HOT
- NameNode startup may take up to 1 hour
  - Read the Namespace image and the Journal edits
  - Wait for block reports from DataNodes (SafeMode)

BackupNode

- BackupNode (> 0.21) is a read-only NameNode
  - Holds namespace in sync with NameNode
  - Works as a persistent storage device for NameNode
  - Performs periodic checkpoints
  - Can perform namespace read requests, as `ls`
  - Does not know replica locations
  - Rejects namespace modification requests, as `mkdir`
  - Cannot take over in case of NameNode failure

Failover Classification

- ***Manual-Cold*** (or no-HA) – an operator manually shuts down and restarts the cluster when the active NameNode fails.
- ***Automatic-Cold*** – save the namespace image and the journal into a shared storage device, and use standard HA software for failover. It can take up to an hour to restart the NameNode.
- ***Manual-Hot*** – the entire file system metadata is fully synchronized on both active and standby nodes, operator manually issues a command to failover to the standby node when active fails.
- ***Automatic-Hot*** – the real HA, provides fast and completely automated failover to the hot standby.
- ***Warm HA*** – BackupNode maintains up to date namespace fully synchronized with the active NameNode. BN rediscovers location from DataNode block reports during failover. May take 20-30 minutes.

HA: State of the Art

- Manual failover is a routine maintenance procedure: Hadoop Wiki
- Automatic-Cold HA first implemented at ContextWeb uses DRBD for mirroring local disk drives between two nodes and Linux-HA as the failover engine
- AvatarNode from Facebook - a manual-hot HA solution. Use for planned NameNode software upgrades without down time
- Five proprietary installations running hot HA
- Designs:
  - HDFS-1623. High Availability Framework for HDFS NN
  - HDFS-2064. Warm HA NameNode going Hot
  - HDFS-2124. NameNode HA using BackupNode as Hot Standby

Terminology

- **NameNode** – active NN, serving client requests
- **BackupNode** – keeps the up-to-date image of the namespace; available for read-only access
- **StandbyNode (SBN)** – has all the functionality of the BackupNode, plus it can change its role to active via the failover command

HA: Design Choices

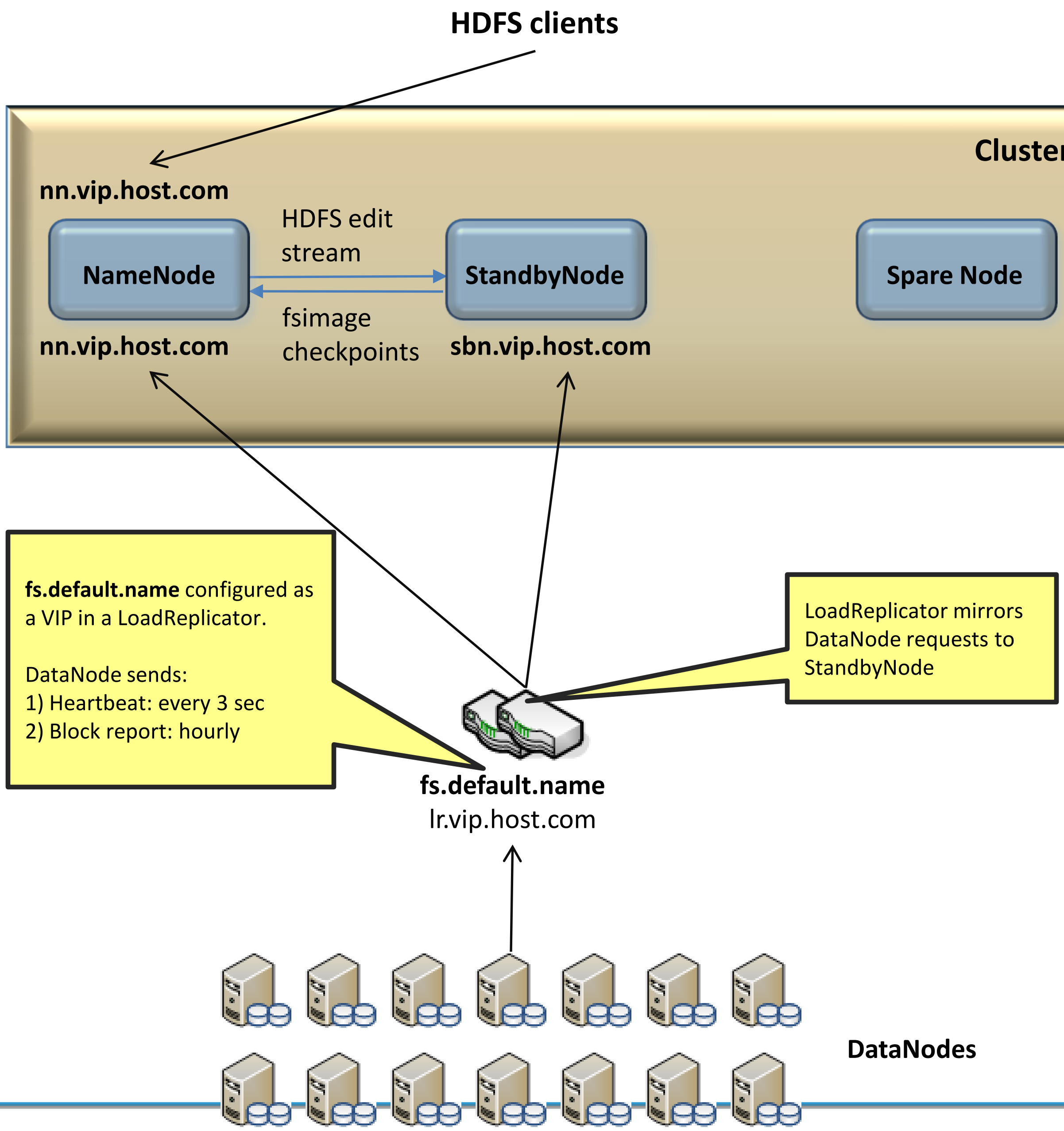
1. Keep StandbyNode *namespace* up to date
2. Up to date *block replica locations*
3. Coordination service / *Failover tool*

Simple Design for Automatic-Hot HA

- Minimalistic approach to eliminate SPOF for HDFS
- Leverage the power of trusted HA software
  - People dedicate their lives to building HA software
  - Hadoop should utilize the results and build on top
- Separate HA handling issues from HDFS-specific ones
  - Hadoop developers focus and optimize for the latter
- Minimize code changes
  - No code – No bugs, No code maintenance
- Minimize the number of distributed components
  - Less components - reduce coordination complexity

Automatic-Hot HA – Design Principles

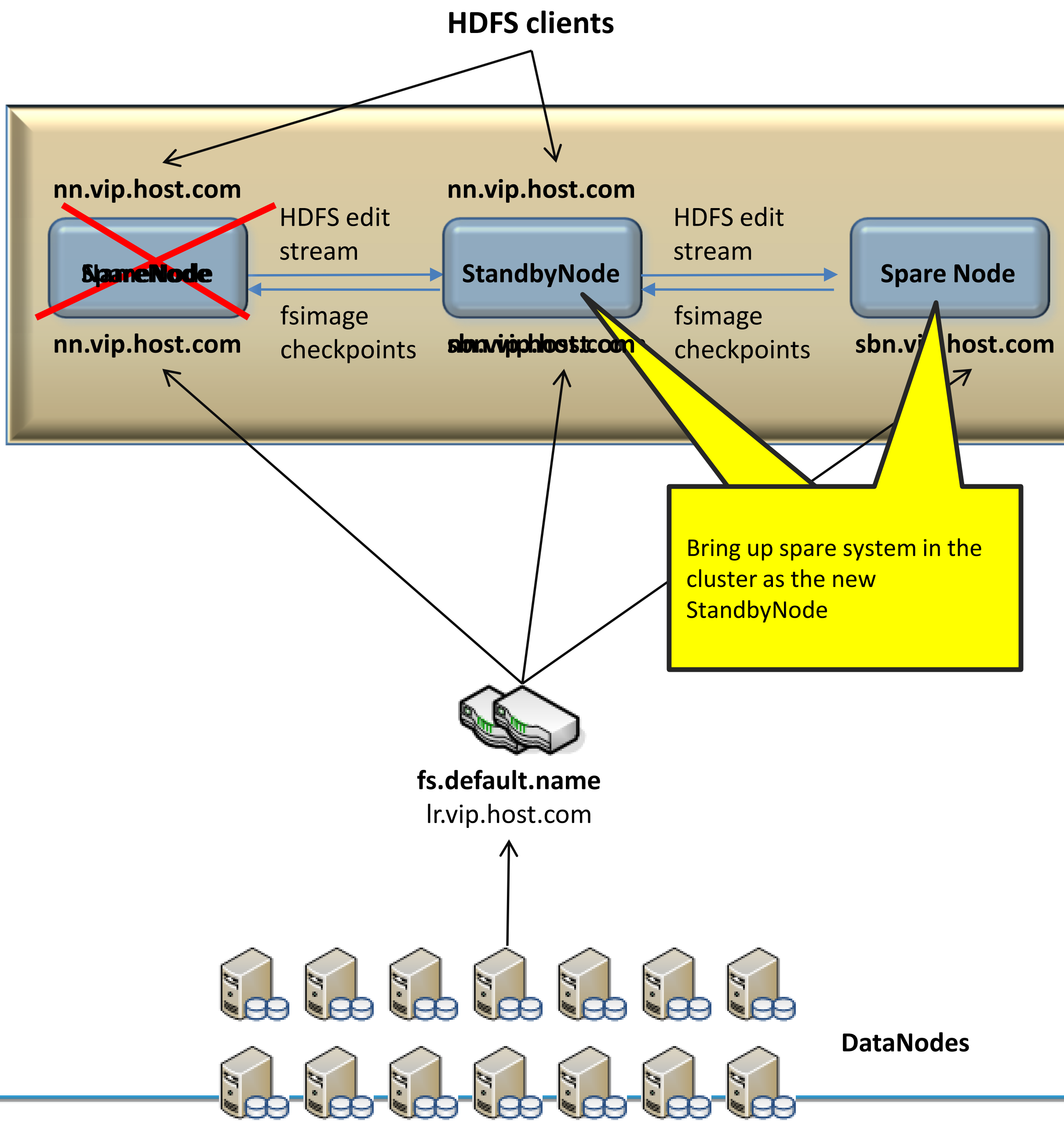
- Standard ***HA software***
  - LinuxHA, VCS, Keepalived
- ***StandbyNode*** – a BN that can become active NN
- ***LoadReplicator*** – a highly available cluster component
  - LoadReplicator is a proxy layer between DataNodes and Name- / Standby- Nodes
  - Zeus Load Balancer = Riverbed Traffic Manager
- ***VIPs*** are assigned to the cluster nodes by their role:
  - NameNode – nn.vip.host.com
  - LoadReplicator – lr.vip.host.com
  - StandbyNode – sbn.vip.host.com
- ***IP-failover***



fs.default.name configured as a VIP in a LoadReplicator.

DataNode sends:
1) Heartbeat: every 3 sec
2) Block report: hourly

LoadReplicator mirrors DataNode requests to StandbyNode


HDFS clients
nn.vip.host.com
SpareNode
NameNode
nn.vip.host.com
HDFS edit stream
fsimage checkpoints
nn.vip.host.com
StandbyNode
sbn.vip.host.com
nn.vip.host.com
HDFS edit stream
fsimage checkpoints
Spare Node
sbn.vip.host.com
Bring up spare system in the cluster as the new StandbyNode
fs.default.name
lr.vip.host.com
DataNodes

Load Replicator

- LoadReplicator is a new highly available component of the cluster, which acts as a proxy layer between DataNodes and NN – SBN pair
- LR is the new target for DataNode messages
- For each received message LR forwards it to NN and SBN
- LR obtains a response from the active NN and forwards it to the DN
- LR ignores any responses from SBN
- LoadReplicator runs on two or more physical nodes
  - Uses commodity hardware
- LR is highly available. When one of LR nodes fails the incoming load is seamlessly redistributed between the remaining nodes
- Introduction of LoadReplicator is justified by
  - Hadoop code changes are isolated to the StandbyNode only
  - Changes to NameNode, DataNode and HDFS client are negligible
- Can be replaced with Local LoadReplicators running on each DN

Failover Command

- `hadoop dfsadmin -failover [nnVIP]`
- Used for manual and auto failover

```
1. Verify that SBN owns passed nnVIP address
2. If nnVIP == NULL, then the verification is not performed.
3. Try to shutdown the NameNode gracefully by sending resign() request via
    •  regular RPC port
    •  HTTP servlet
    •  service RPC port
4. If checkpoint is in progress, then stop the process, and prevent from
   starting a new checkpoint.
5. Stop the RPC client talking to the NameNode
6. Digest journal spool if checkpoint was in progress. Increment
   checkpointTime to make it the latest image
7. Stop the Checkpoint Manager
8. Change role to active
9. Reset SafeMode extension to the configured value
10.Reset lease hard limit to the default
```

Comparison

	This approach HDFS-2064	Shared storage HDFS-1623
Namespace Synchronization	Real-time Journal Streaming to StandbyNode	Shared storage NFS Filer 60 sec delayed
Block Locations	LoadReplicator	DataNodes distinguish between Active NN and SBN
Failover	IP failover LinuxHA, VCS, keepalived	Zookeeper coordinating Clients, DataNodes, etc.
Code Changes	• SBN extension of BN • Admin “failover” command • NN health monitor	• All from left … Plus • Clients talking to ZK, NN • DNs talking to NN, SBN, ZK • Failover

Disadvantages of Shared Storage Synchronization

- Shared storage design requires investment in NFS Filer
  - $20K - $100K
  - Buy 10 - 20 more nodes instead
- Dependency on Enterprise Hardware for Hadoop commodity hardware clusters
- Hot HA, except 60 sec delay

Project Status

- Detailed design attached to HDFS-2064
  - Reviewed
- Patch for 0.22 is available
  - Will keep up to date
- Proof of concept demo
  - Keepalived
  - Trial version of Zeus
  - Configuration verified
- Just get HA done. Scalability is the biggest problem
  - Dynamic namespace partitioning with distributed NN
  - Federation provides static partitioning only

Thank You!

Hadoop 0.22

- Branch created 11/17/2010
- Testing of new and existing features
  - Append by HBase community
  - Different groups in Hadoop community
- RM from August
  - Testing team at eBay and many other people
  - Good testing results. Reliable
  - Bugs found in 0.22 are also in trunk
  - BigTop builds 0.22
- Community release: Important
  - Works but not 0.20. Good new features. Missing parts
  - 0.22.1 add Security, Disk Fail in place, MR task limits, optimizations, metrics 2.   Add HA?