Light-weighted HDFS Disaster Recovery

Qiyuan Gong

Intel SSG/OTC

2018

Intel® Software contributions to big data and analytics. From open enterprise-ready software platforms to analytics building blocks, runtime optimizations, tools, benchmarks and use cases, Intel Software makes big data and analytics faster, easier, and more insightful.

https://software.intel.com/en-us/bigdata

Qiyuan Gong
Software Engineer, Intel OTC (Open Source Technology Center)
Working on: Deep Learning on Hadoop (Canceled)
SSM (Smart Storage Management for Big Data)
Ph.D on Data anonymization (related to GDPR)

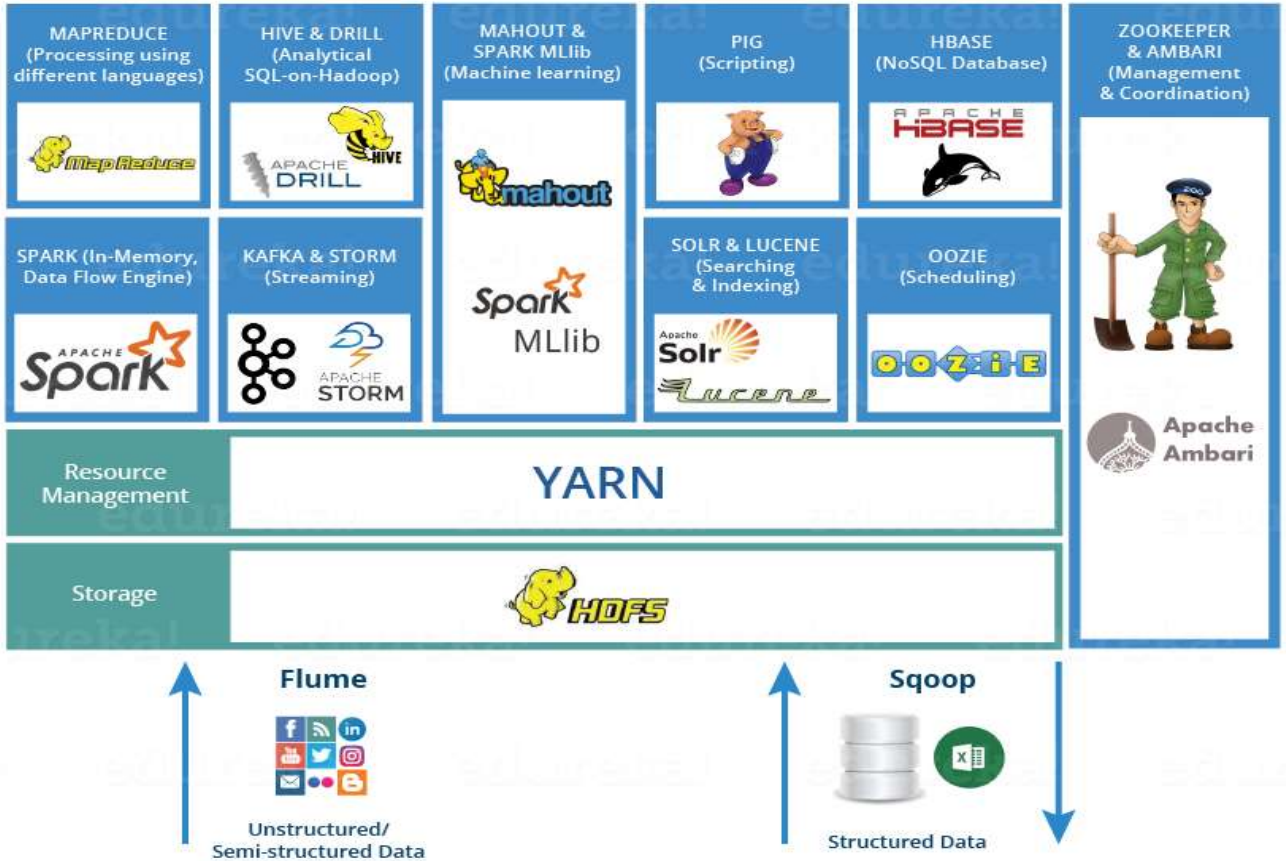
qiyuan.gong@intel.com
https://www.linkedin.com/in/qiyuangong/

Outlines

- Background & Motivation
- Basic Design & Experience
- Evaluation
- Example & Demo

Background & Motivation

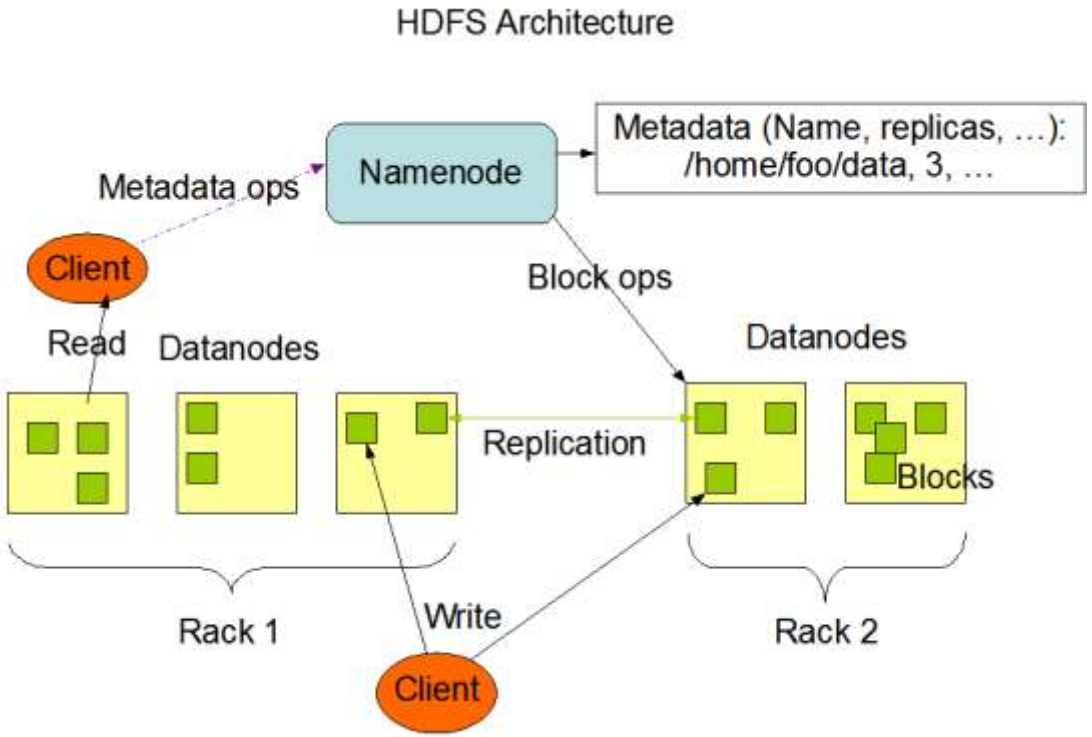
- HDFS (Hadoop Disturbed File System) is widely used in production



Source: https://www.edureka.co/blog/hadoop-ecosystem?utm_campaign=social-media-edureka-sep-ss&utm_medium=crosspost&utm_source=quora

Background & Motivation

- HDFS is architected to operate efficiently at scale for **normal hardware failures within a datacenter**
  - Disk, node or rack fails will not affect HDFS
- But HDFS is **not designed to handle datacenter failures**
  - If data center is down, we lost everything (meta and data)



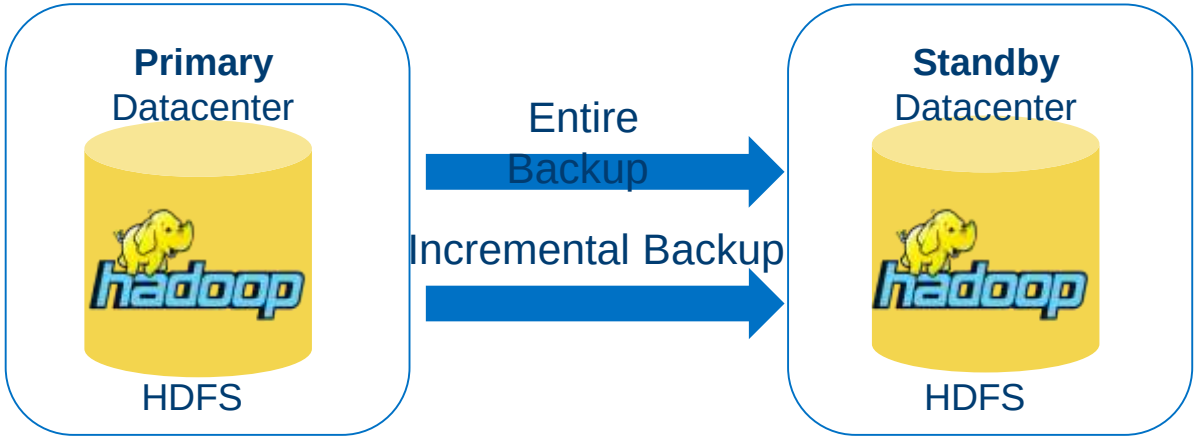
Source: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

Background & Motivation

- Disaster Recovery is not equal to backup
  - **Essential**
    - **Location Separated –** High latency & limited bandwidth
    - **No Data loss -** Backup & recovery
    - **High Availability –** Switch to standby data center
  - **Additional**
    - Performance - Higher is better
    - Resource Usage - Don’t use up all resource/ bandwidth
    - …
- **But let’s focus on backup**

Basic Design & Experience

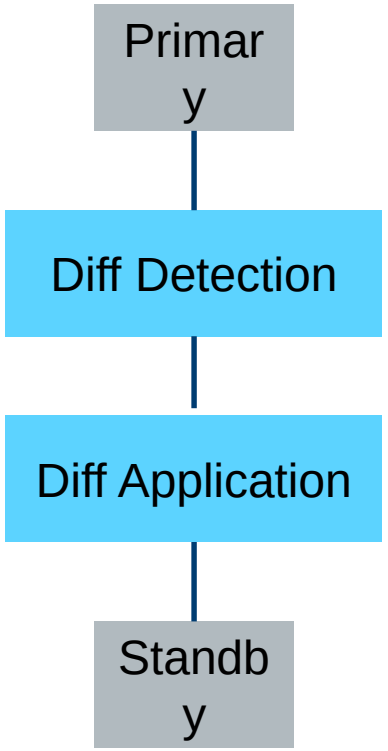
- Backup HDFS from Primary (local) to standby (remote)
  - Entire/Base backup – first backup, focus on performance, no tricks (compression and encryption)
  - **Incremental backup** – lots of **fun** & **pain**



Quit similar to version control system, such as Git

Basic Design & Experience

- **Incremental Backup**
  - **Diff detection - What is changed?**
    - Find out (**incremental)** difference
    - Building diff application tasks/list
  - **Diff application - How to apply changes to remote?**
    - Apply diff content to Standby Cluster
- **Key points**
  - Precise diff (algorithm)
  - Performance (concurrency) & Resource usage (CPU, memory and bandwidth)
  - Corner cases – merge failed with N conflicts



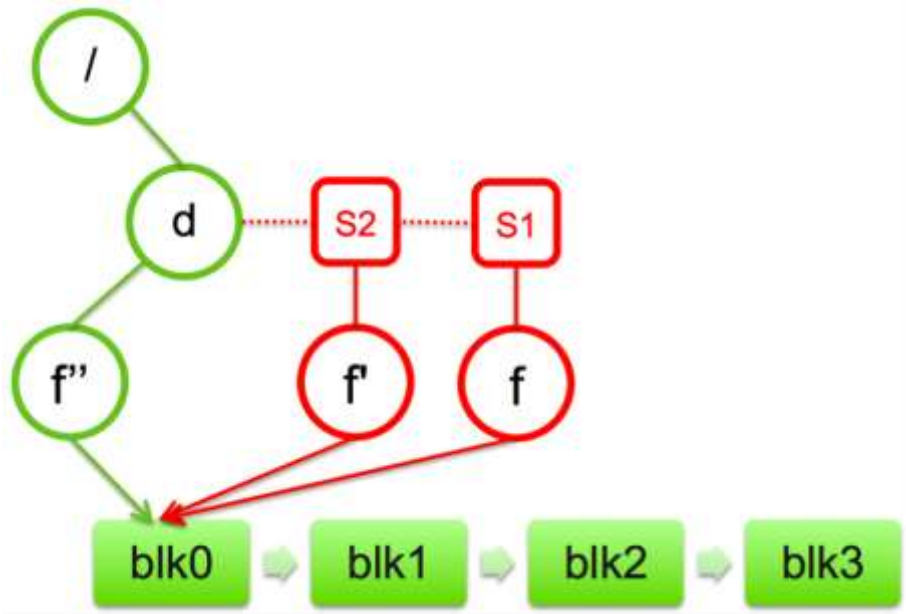
git fetch (patch) from remote

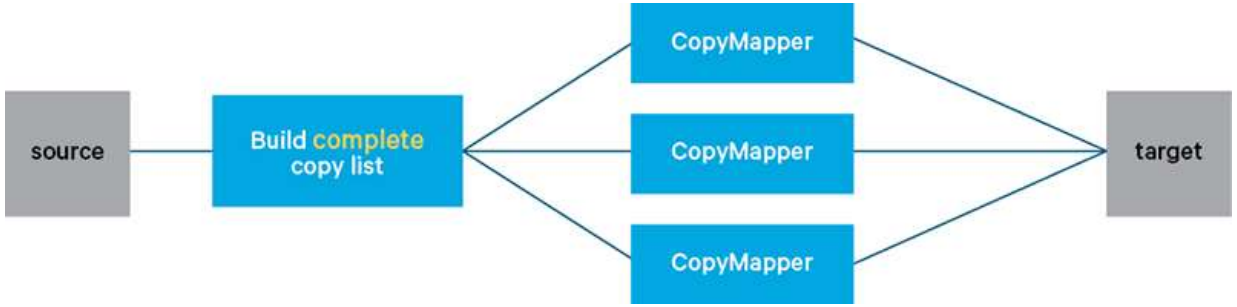
git rebase/merge

```
Auto-merging pom.xml
CONFLICT (content): Merge conflict in pom.xml
Automatic merge failed; fix conflicts and then commit the result.
```

Basic Design & Experience

- **DistCp's Advantages**
  - **Build-in** tool/command in Hadoop
  - Track changes with **Snapshot**
    - Snapshot report for diff detection (Create, Delete, Rename, Modify)
    - Copy from Snapshot (Snapshot is a reference which helps keeping deleted blocks)
  - Apply changes with **MapReduce**
    - Copy contents concurrently in block level
    - Old version in file level



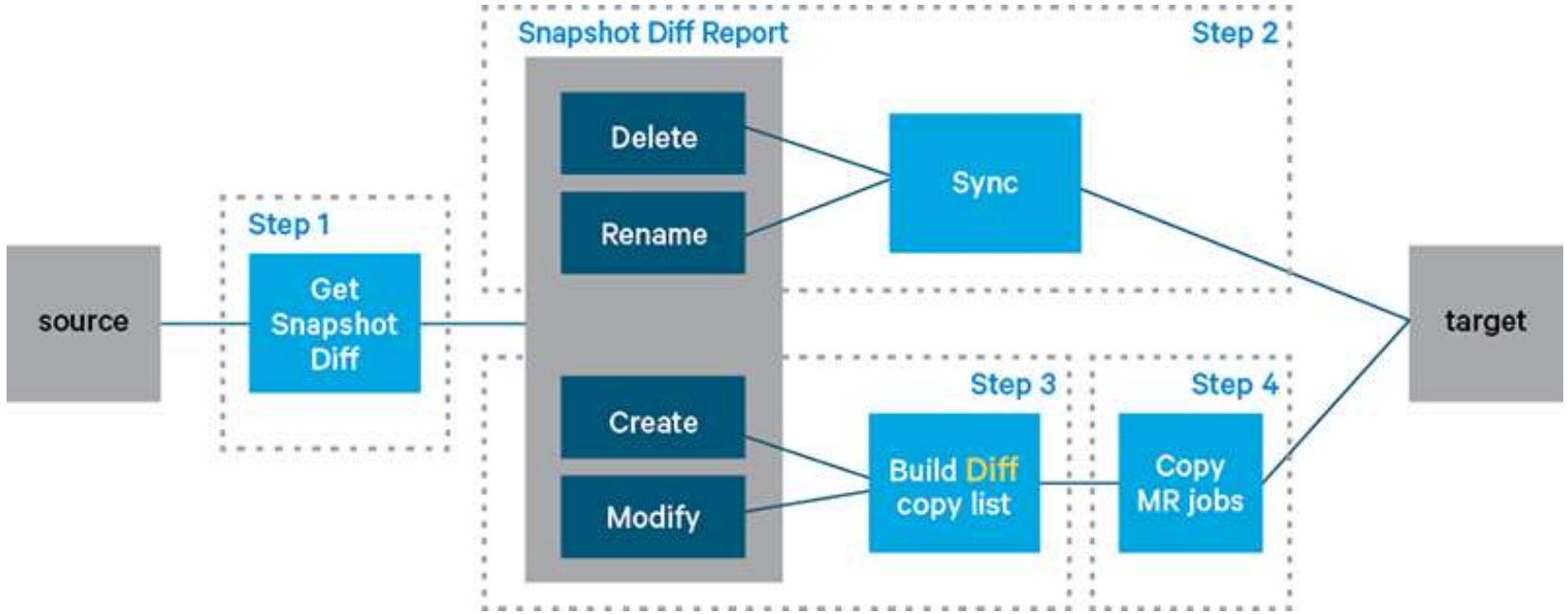


Source: https://community.hortonworks.com/articles/60546/hdfs-snapshots-1-overview.html
Source: http://blog.cloudera.com/blog/2015/12/distcp-performance-improvements-in-apache-hadoop/

Basic Design & Experience

- **DistCp does great jobs in most cases**

distcp -update -diff s0 s1 <sourceDir> <targetDir>



Source: http://blog.cloudera.com/blog/2015/12/distcp-performance-improvements-in-apache-hadoop/

Basic Design & Experience

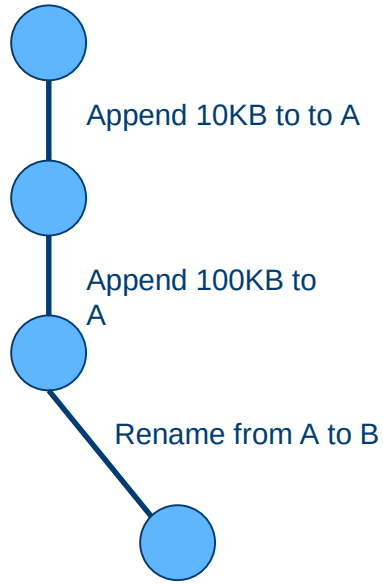
- **Pain points of DistCp:**
  - **Snapshot is not perfect**
    - Cannot take snapshot on root directory
    - **No details** about **Modify** (need to compare files)
    - **Corner cases**
  - **MapReduce is too heavy to incremental backup**
    - **1 container (2 GB memory & 1 core) for 10KB appe**
    - **Long launch time**
  - **Requires administrator** (failed tasks, snapshot manageme
  - No resource usage & bandwidth control

#	First op	Second op	Items in snapshot diff report	Extra Actions
1	create A	rename A -> B	create B	
2		rename A's parents C -> D	create C/A	Change path of create op from C/A -> D/A
3		rename A's child A/C -> A/D	create A	
4		rename a pre-existing C into A	create A	Exclude C while traversing A
5		rename A's child C out of A	create A, create C	
6		modify A	create A	
7		modify A's parents	create A, modify A's parents	
8		modify A's children	create A	
9		delete A	NULL	
10		delete A's parents	NULL	
11		delete A's children	create A	
12	modify A	rename A -> B	modify A, rename A -> B	Change path of modify op: A -> B
13		rename A's parents C -> D	modify A, rename A's parent C -> D	Change path of modify op: C/A -> D/A
14		rename A's children	modify A, rename A's children	
15		delete A	delete A	
16		delete A's parents	delete A's parent	
17		delete A's children	modify A	
18		create A	N/A	
19		create A's parents	N/A	
20		create A's children	modify A, create A's children	
21	delete A	create A	delete A, create A	
22		modify A	N/A	
23		rename A -> B	N/A	
24	rename A -> B	delete A	N/A	
25		delete A's parents-before-rename	delete A's parents-before-rename, rename A -> B[1]	
26		delete B's parents	delete B's parents, delete A	
27		delete A's children	N/A	
28		delete B's children	delete A's children, modify A, rename A -> B	Change path of modify op: A -> B
29		modify B	modify A, rename A -> B	Change path of modify op: A -> B
30		create A	create A, rename A -> B	No need to change path A of create op, since the newly created A is not the same as the old "A" which is renamed to B

Source: http://blog.cloudera.com/blog/2015/12/distcp-performance-improvements-in-apache-hadoop/

Basic Design & Experience

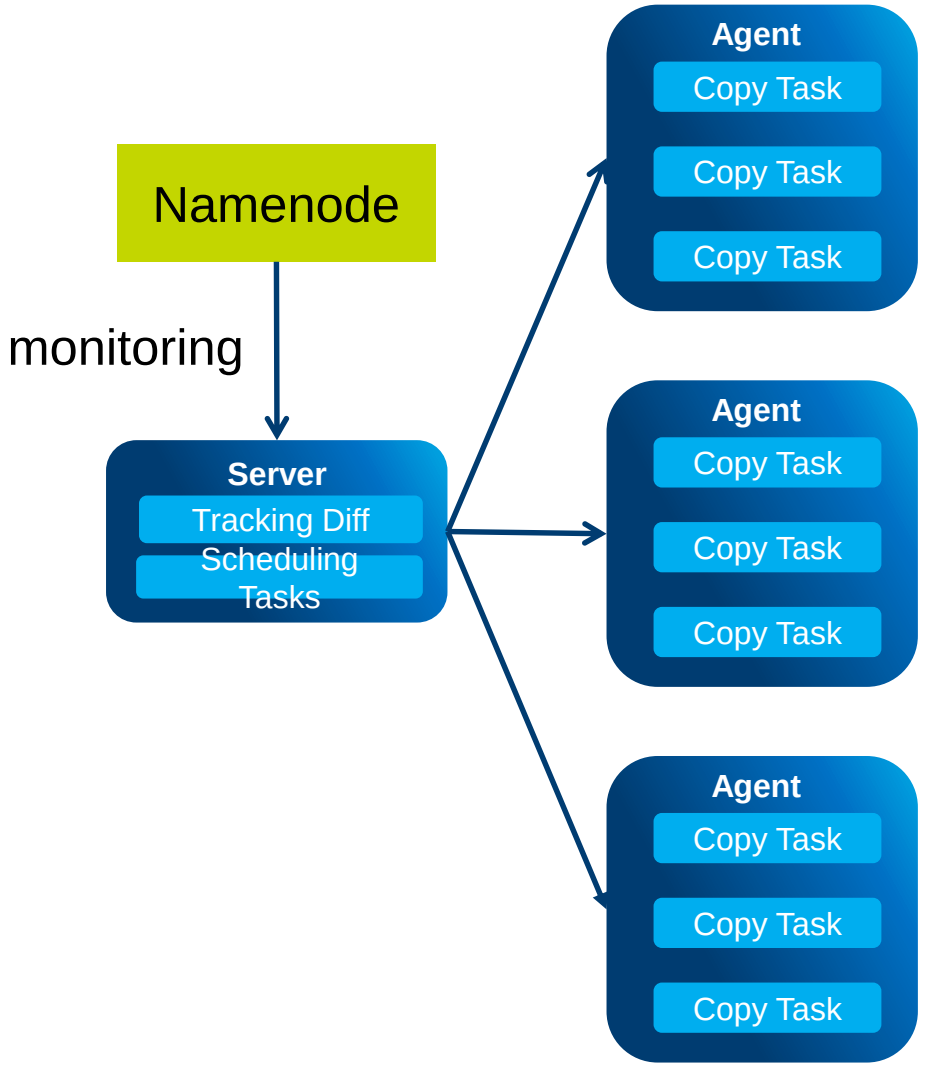
- **We want to address above problems in this way:**
  - **No Snapshot** – Track changes with **log (Inotify/editlog) post-processing**
    - Track changes in details (Create, Delete, Rename, **Meta**, **Close**, **Append**, **Truncate**) v.s. Snapshot (Create, Delete, Rename and Modify)
    - Build **lineage** from **Inotify** (subset of editlog)
    - Translate Events into tasks, e.g.,
      - Append means copy content from primary to remote
      - Delete means delete file in remote
      - …



Basic Design & Experience

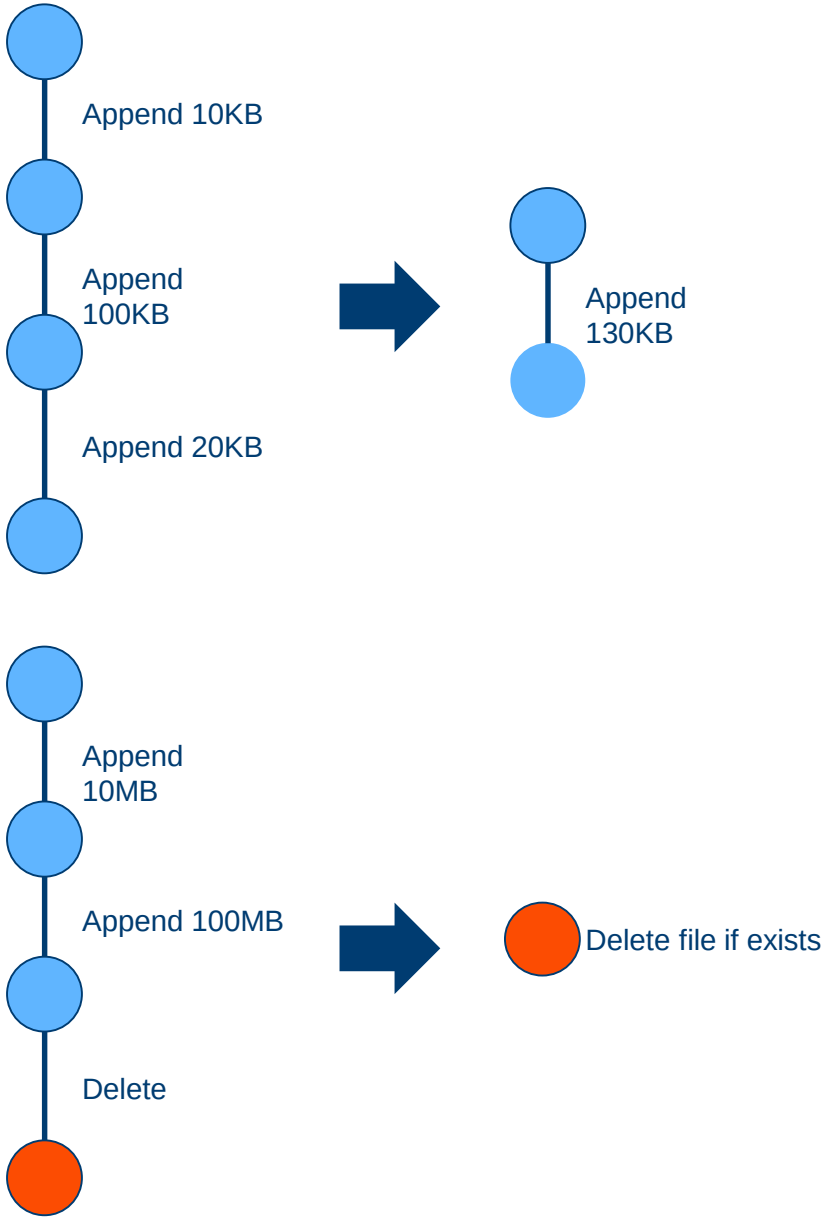
- **No MapReduce – Long running services**
  - Server
    - Tracking changes
    - Scheduling tasks
    - Resource and bandwidth management
  - Agents
    - Applying changes to remote

Sometimes, they don't even have MapReduce



Basic Design & Experience

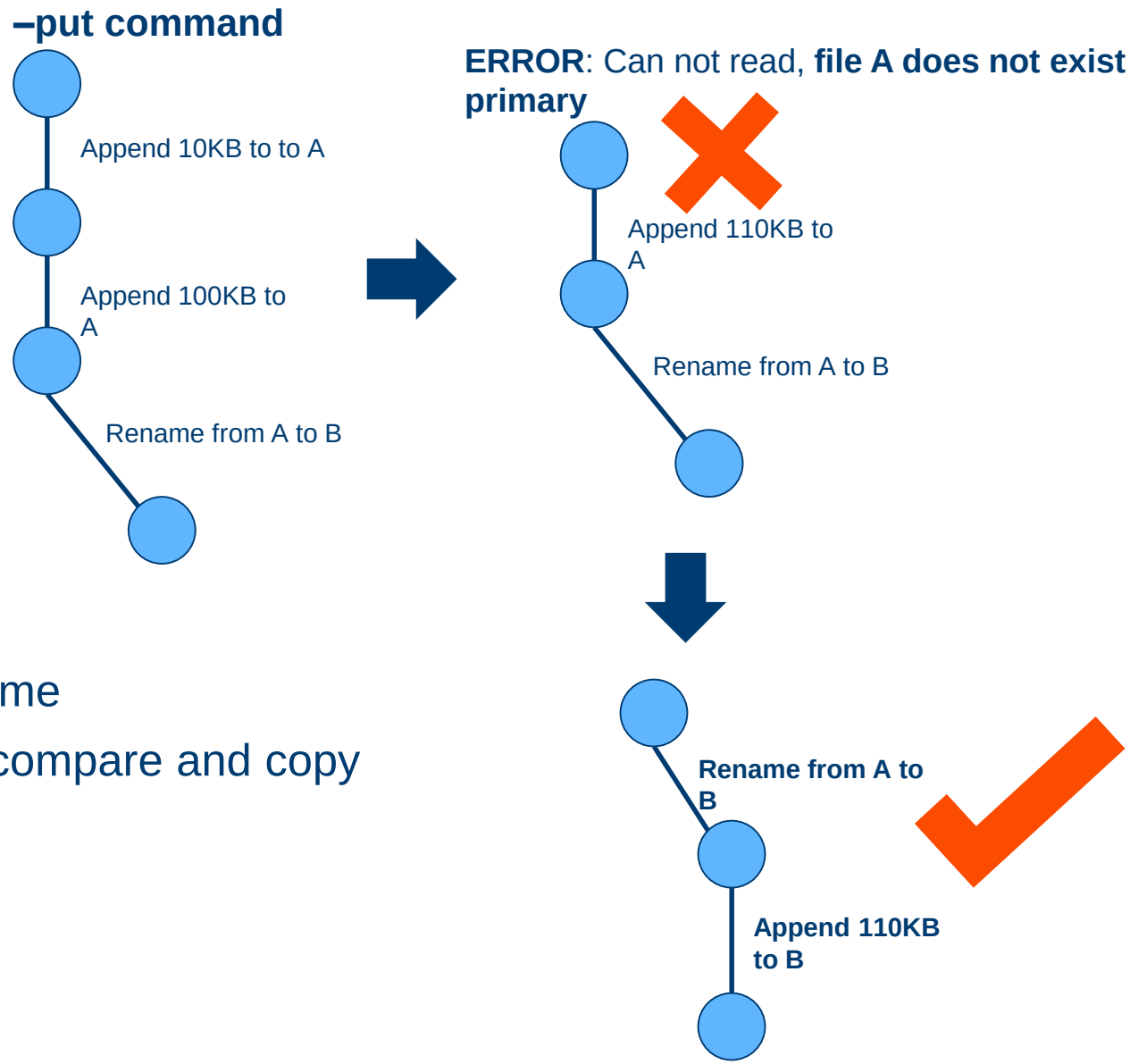
- **Advantages of SSM DR (Disaster Recovery)**
  - **Diff Detection: Log post-processing**
    - **Near real-time** diff detection
    - **More details than Snapshot, e.g., Modify**
    - Diff & Lineage is **mergeable**
    - Limited Impact to Namenode
      - Getting Inotify has **impact** to Namenode
      - Getting EditLog has **no impact**
  - **Diff Application: Long running Services**
    - Light-weighted for Cluster/Data center
    - Auto scale & bandwidth control
    - Can be Containerized



Background & Motivation

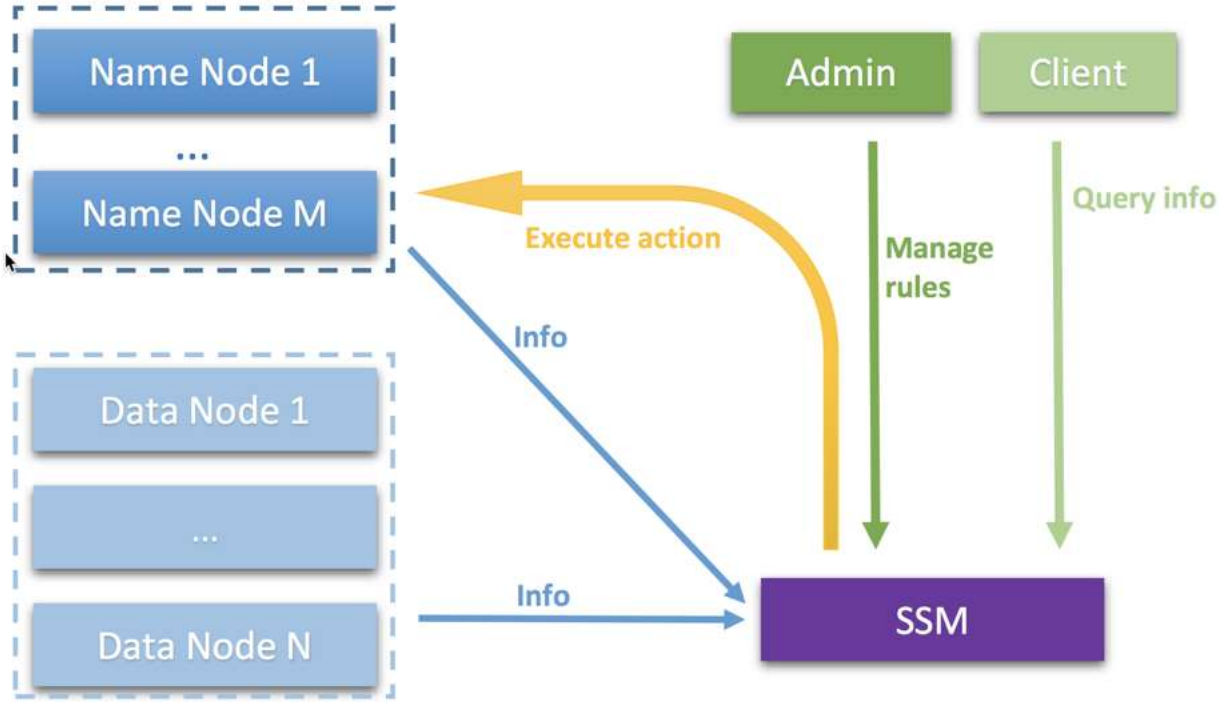
- **Pain points (painful experience)**
  - Corner cases
  - No more Locality

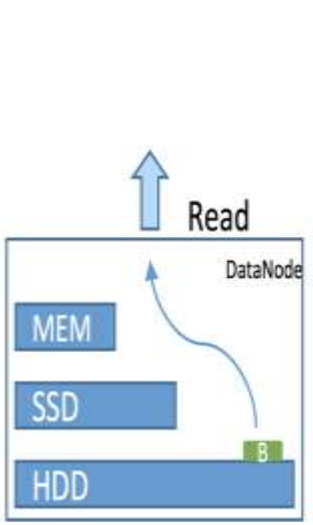
- How to handle it?
  - Corner cases
    - Handle general cases, such as rename
    - Let tasks fail, then trigger a directly compare and copy
  - No Locality
    - High speed network
    - Place agent near datanodes

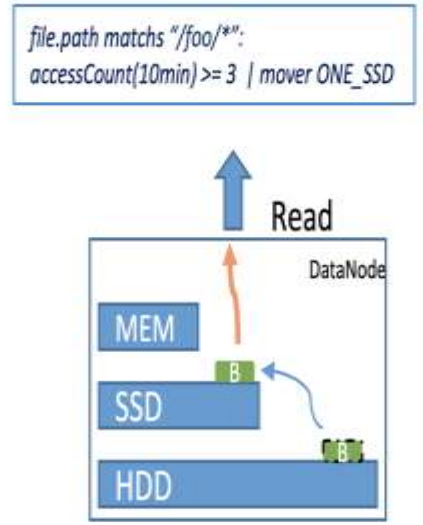


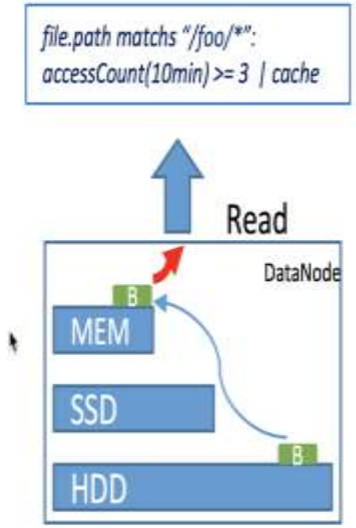
Basic Design & Experience

- Build Disaster Recovery Based on SSM (Smart Storage Management, Open source on Github)



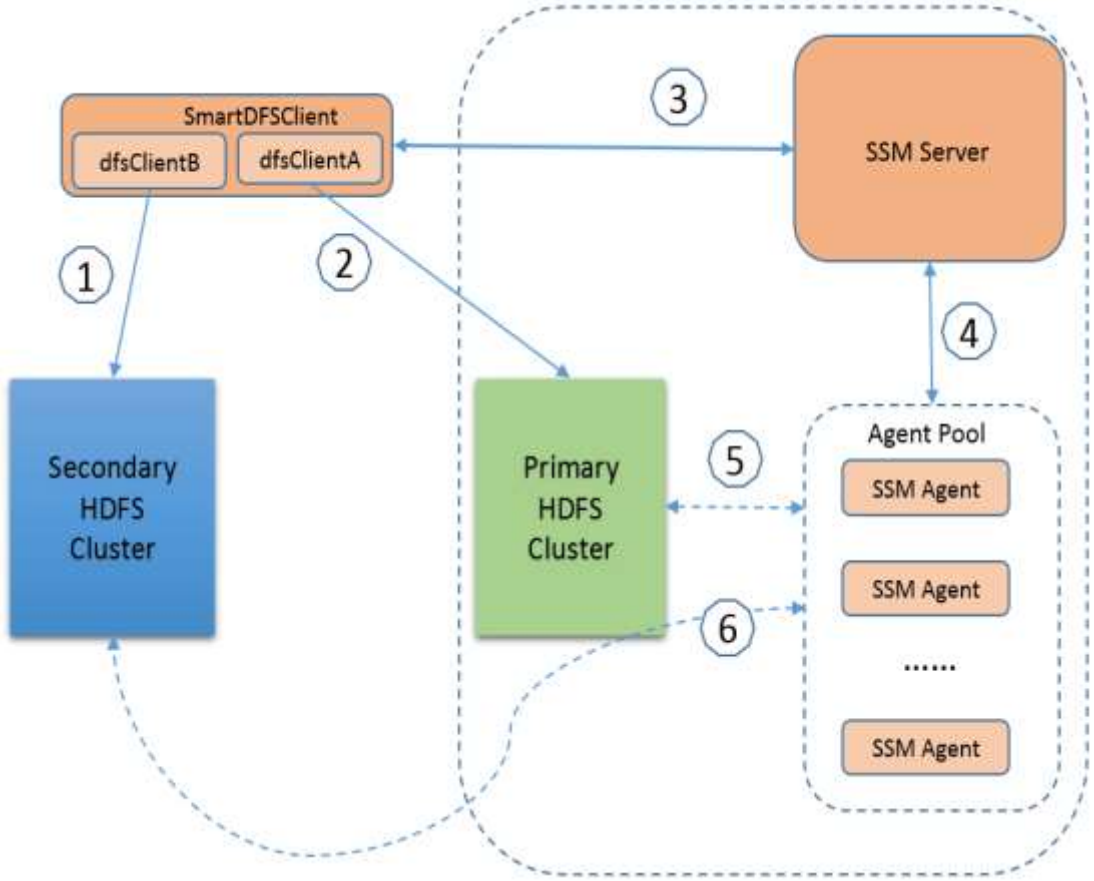






Basic Design & Experience

- Reuse SSM modules and metadata
  - SSM's Metadata for diff detection
  - SSM's modules (Agents and actions) for diff applicati
- Features
  - Smart management based on Rules
  - Near real time incremental backup
  - Less resource requirement
  - High Availability & Transparent read/backup (TODO)



Evaluation - Test Environment

HDFS Clusters

- Primary HDFS 4 Nodes
- Secondary HDFS 4 Nodes

SSM

- SSM Server
- 3 SSM Agents installed on Primary datanodes

Testing Methodology

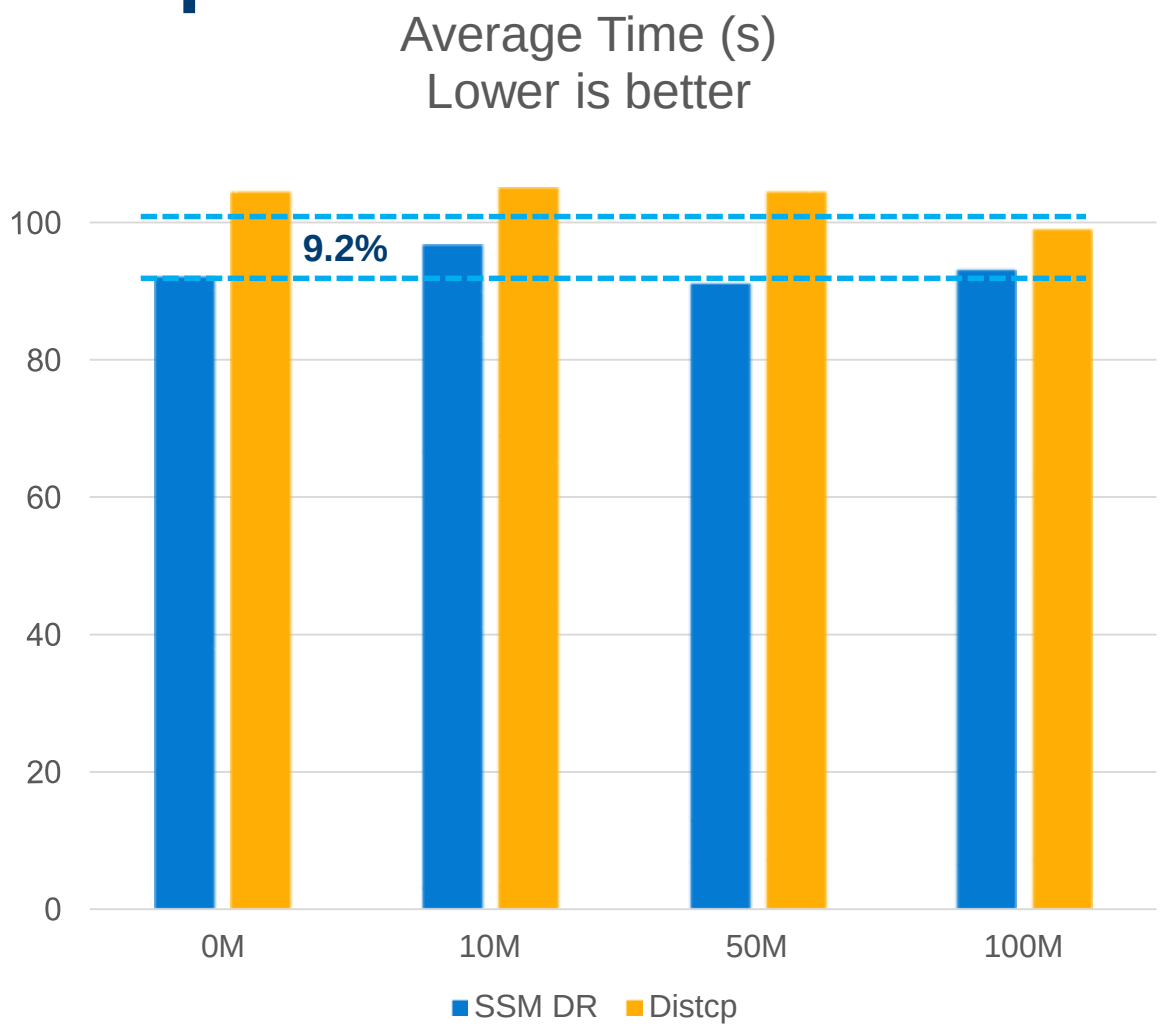
- Compare SSM DR with distcp
  - 10K small files – 1MB
  - 1K large files – 100MB

Hardware configuration	
Processor	Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz
DRAM	256GB DDR4 16x16GB @ 2133MHz
Network	10GbE
Disks	5 x SATA(2TB) ST2000NM0011

Software configuration	
OS	CentOS 7.3.1611 x86_64
Hadoop	2.7.3
Metastore	Mysql 5.7.19
Java	JDK1.7.0_141

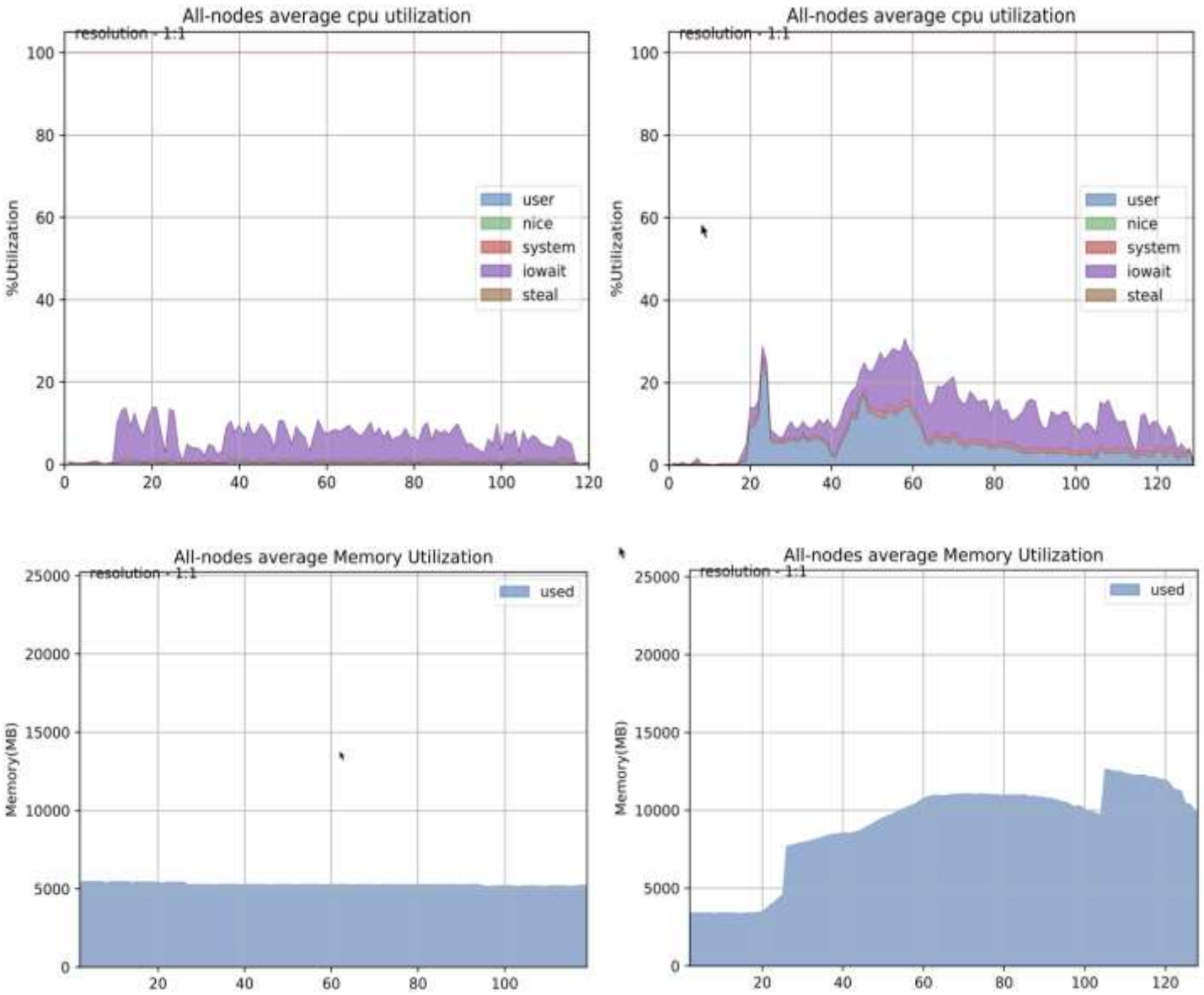
Workload-2 1MB Files Batch Backup

- Workload:
  - **10000 files * (1MB**) and 60-concurrency
- Result:
  - SSM DR will performer 9.2% **better** than DistCp



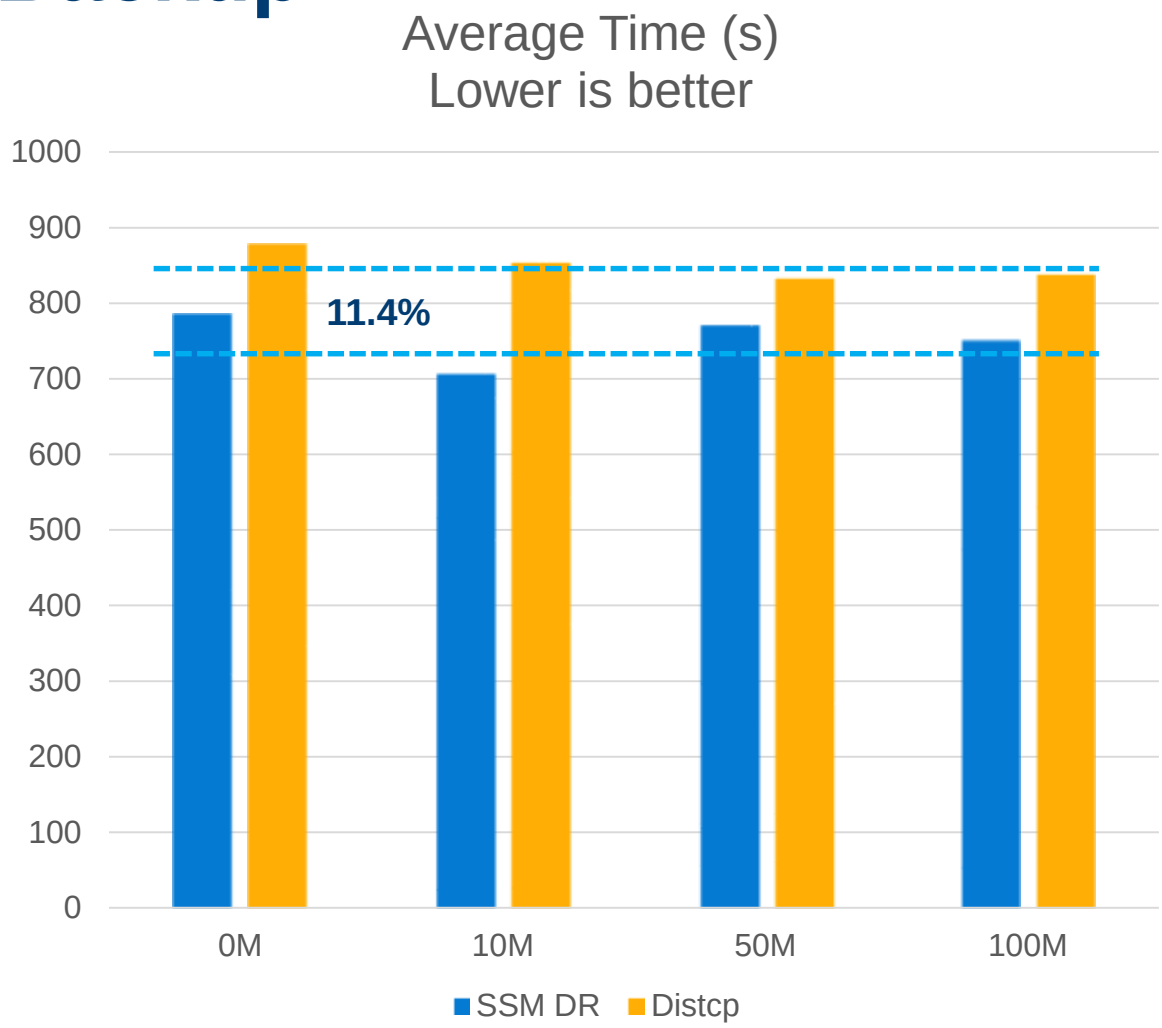
Resource Usage

- Workload:
  - Batch backup files (**10000 *1MB**)
- Result:
  - SSM DR requires **less** resource than DistCp on working nodes
    - **46%** less CPU
    - **41%** less memory



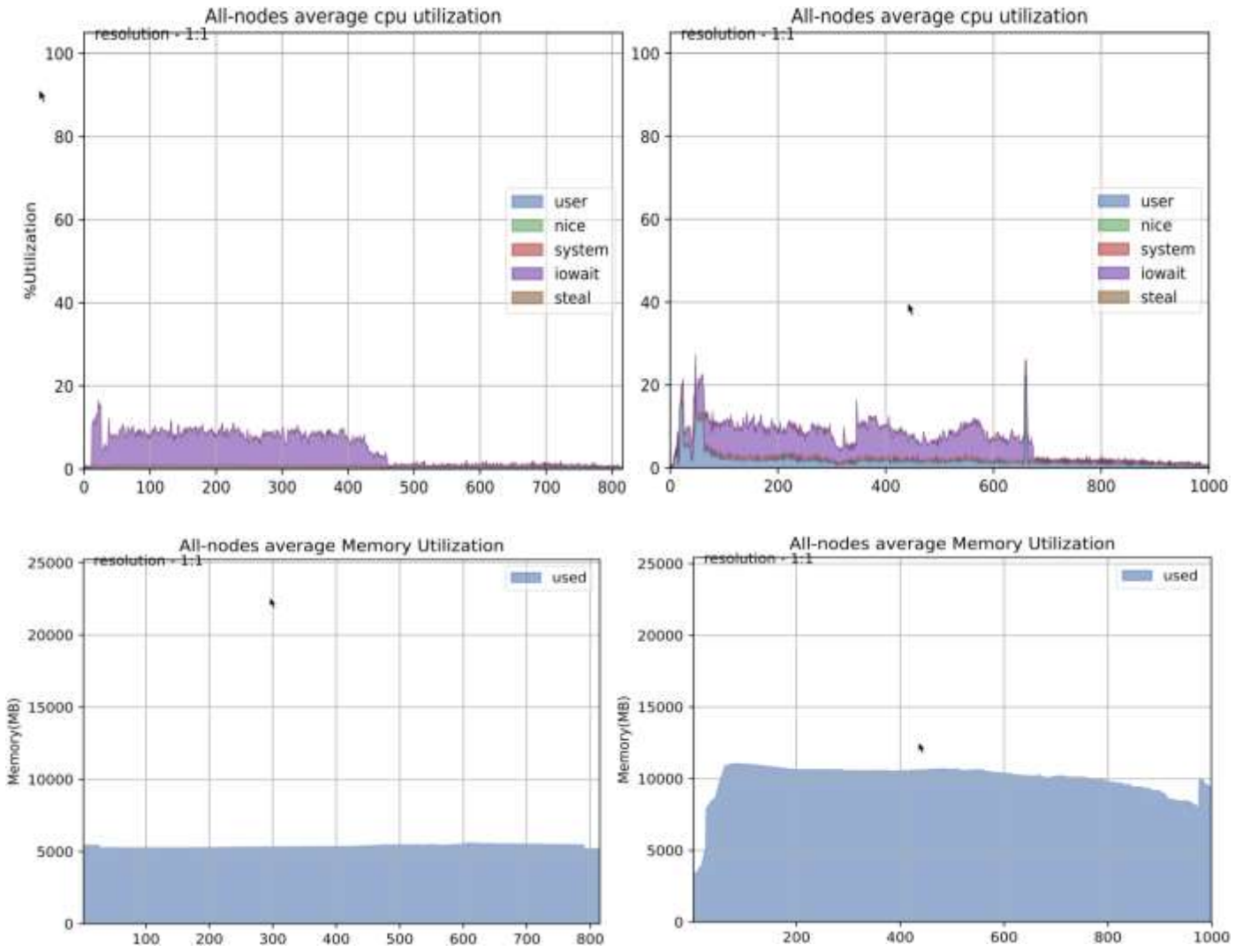
Workload-2 100MB Files Batch Backup

- Workload:
  - 1000 files * (100MB) and 60 concurrency
- Result:
  - SSM DR performs 11.4% **better** than DistCp



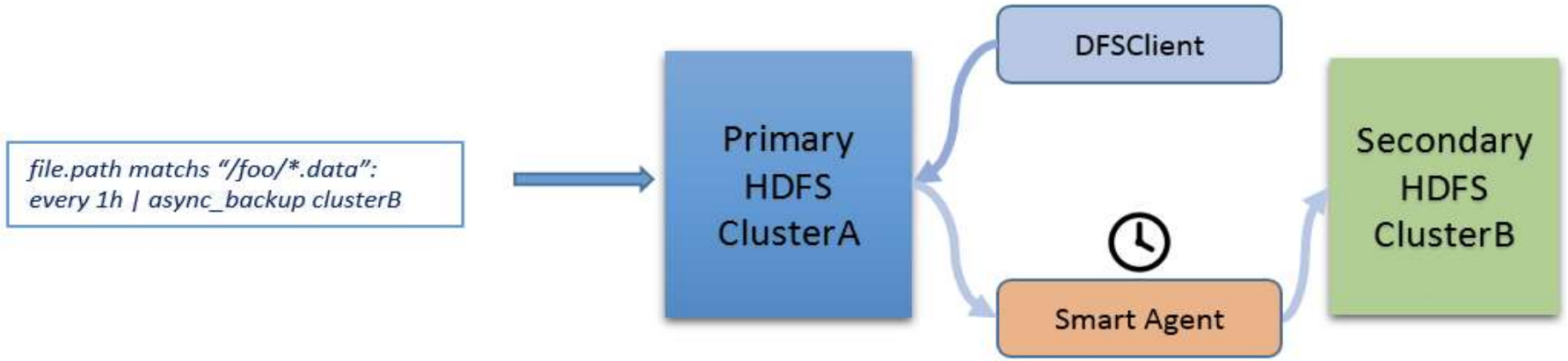
Resource Usage

- Workload:
  - Batch backup files (**1000 *100MB**)
- Result:
  - SSM DR requires **less** resource than DistCp on working nodes
    - **26%** less CPU
    - **46%** less memory



SSM Disaster Recovery: Example and Demo

- Rule
  - file: every 500ms | path matches "/src1/*" | sync –dest hdfs://namenode:9000/dest1/
  - file: every 2s | path matches "/src/*.txt" | sync –dest hdfs://namenode3:9000/dest/



Thanks

- Q&A
- More details
  - https://issues.apache.org/jira/browse/HDFS-7343
  - https://events.static.linuxfound.org/sites/events/files/slides/ApacheBigDataEurope2016-SSM.pdf
  - https://github.com/Intel-bigdata/SSM

Legal Disclaimer & Optimization Notice

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Optimization Notice

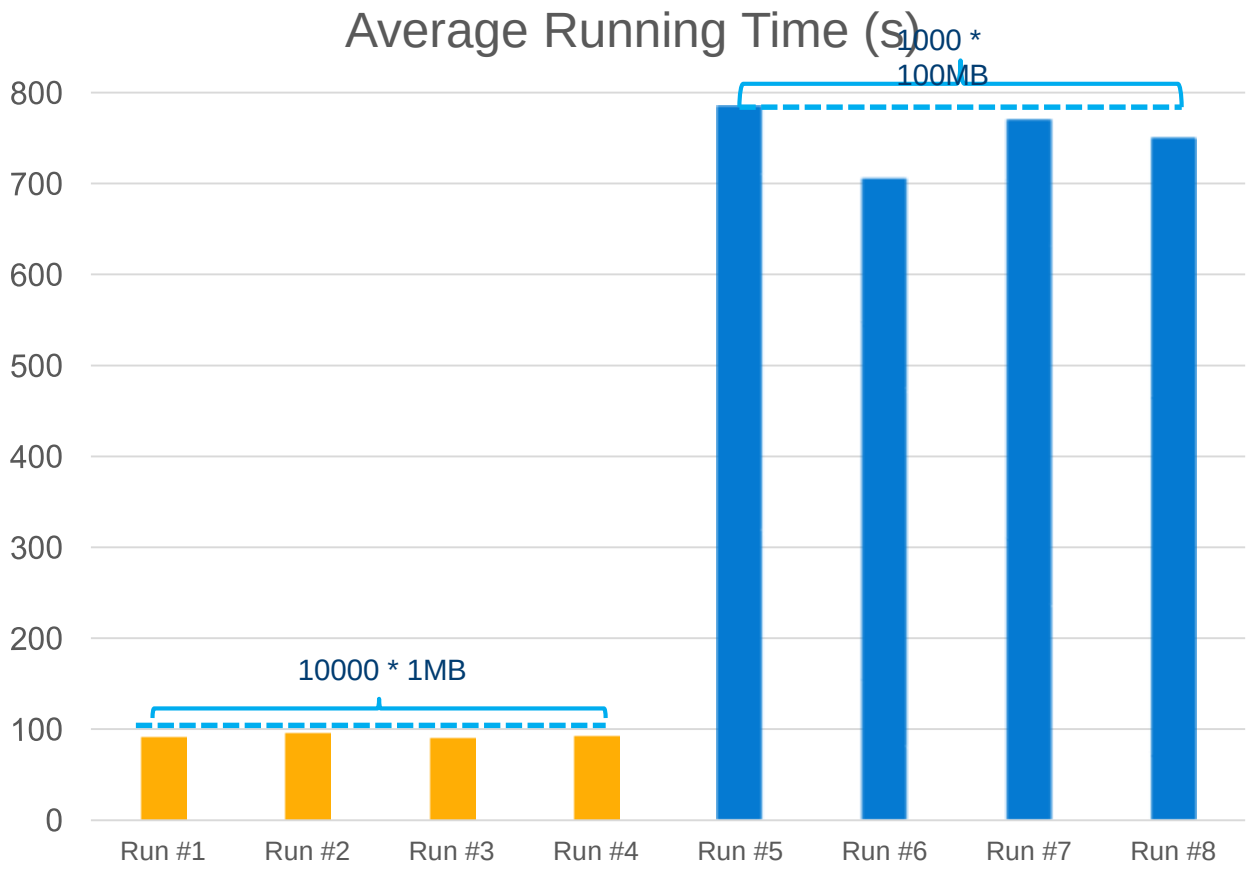
Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804


intel

Namespace Impact on SSM's Performance

- Workload:
  - Batch backup files
  - (0M, 10M, 50M, 100M) files in namespace
- Result:
  - Namespace has **very limited** impact on SSM Disaster Recovery's performance



Basic Design & Experience

- **Diff Detection**

Methods	View	Advantage	Dis-advantage
Directly Scan	Result view	Easy to develop	1. Long laterncy 2. Network overhead 3. Impact to Namenode 4. Not real-time
HDFS Snapshot	Result view	Easy to develop	1. A few constrains caused by snapshot 2. Impact to namenode 3. Not real-time
Log based	Continually View	**1. Real time async is possible** **2. Less impact to namenode**	Hard to develop and maintain

- V
  - Nearly real-time Async based on incremental diffs
  - Limited or no impact to namenode

Basic Design & Experience

- **Diff Detection**

Methods	View	Advantage	Dis-advantage
Directly Scan	Result view	Easy to develop	1. Long laterncy 2. Network overhead 3. Impact to Namenode 4. Not real-time
HDFS Snapshot	Result view	Easy to develop	1. A few constrains caused by snapshot 2. Impact to namenode 3. Not real-time
Log based	Continually View	**1. Real time async is possible** **2. Less impact to namenode**	Hard to develop and maintain

- V
  - Nearly real-time Async based on incremental diffs
  - Limited or no impact to namenode

Basic Design & Experience

- Diff Application

Methods	Advantage	Dis-advantage
Container based (e.g., map-reduce)	Easy to develop	1. More resource 2. Less parallelism in the same cluster
Task based	**1. Less resource usage** **2. More parallelism (multiple tasks in single container)**	1. In some cases, agents needed 2. Hard to develop

- We found: **Task based** is works better
  - **Light-weighted: multiple tasks in one container**
  - Hard to develop is not a question any more: lots of distributed frameworks, e.g., Akka

Basic Design & Experience

- Our **light-weighted** DR basic design
  - Diff Detection without Snapshot
    - Long running service monitoring **Editlog or Inotify**
    - **Directly Scan** to handle corner cases and namespace mis-match
  - Diff Application without Map-reduce
    - **Agents based on Akka** (can be replaced by other frameworks)
      - **Master** dispatch diff application tasks
      - **Slaves** apply diff to remote (multiple tasks in one container)

Light-weighted for Cluster/Data center:

1. Limited Impact to HDFS (HDFS event don't know DR
2. Less resource requirement
3. Can be Containerized (in future)