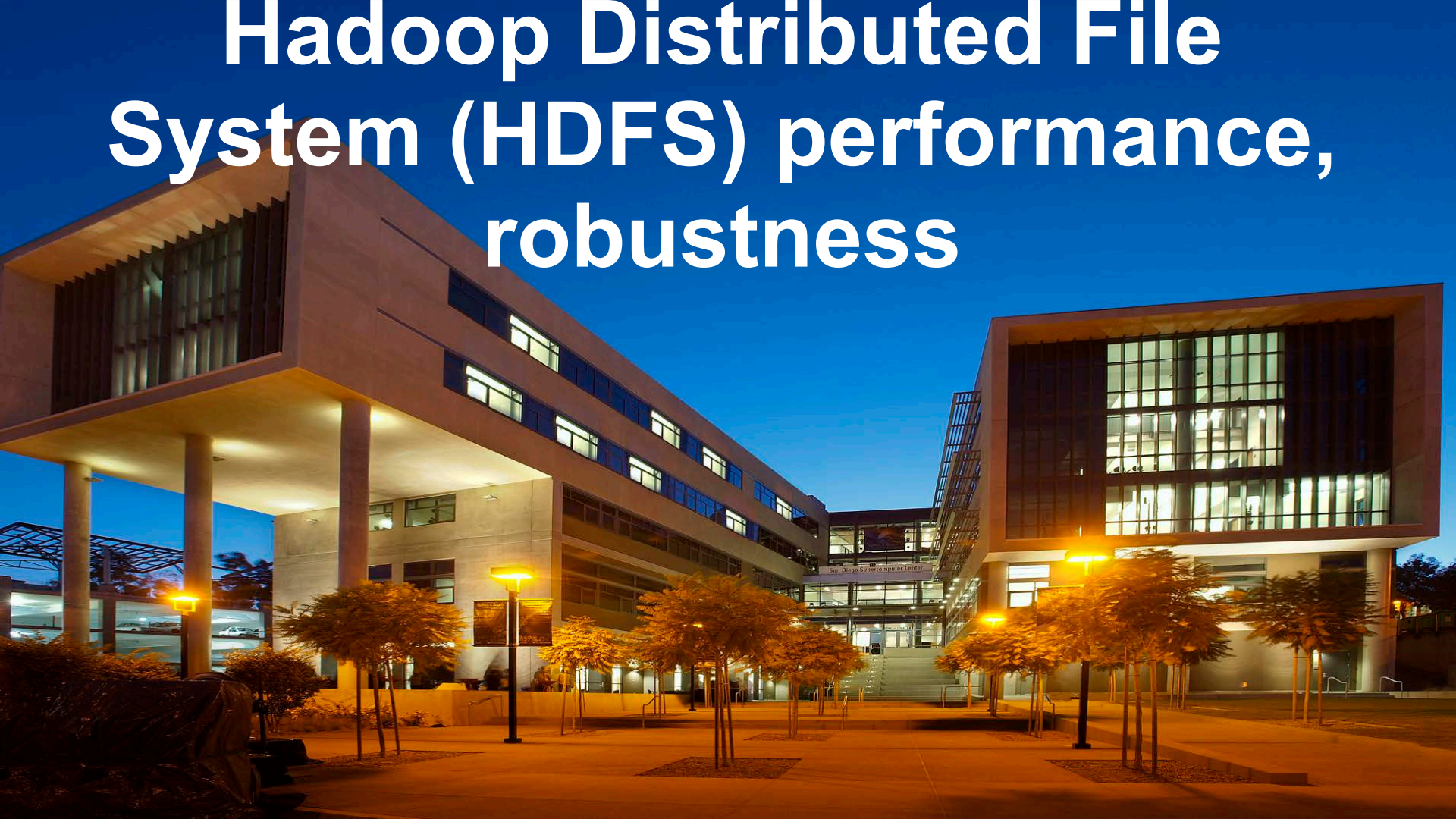
Hadoop Distributed File System (HDFS) performance, robustness

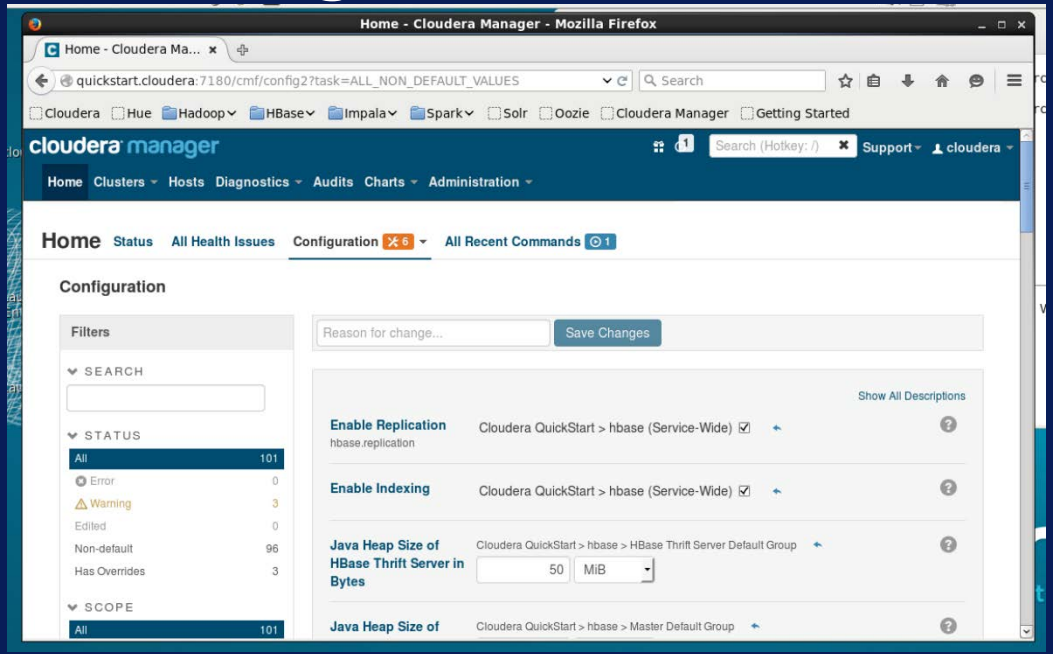


Overview of HDFS Tuning

- *Tuning parameters*
- *DFS Block Size*
- *NameNode, DataNode system/dfs parameters.*

HDFS Configuration file

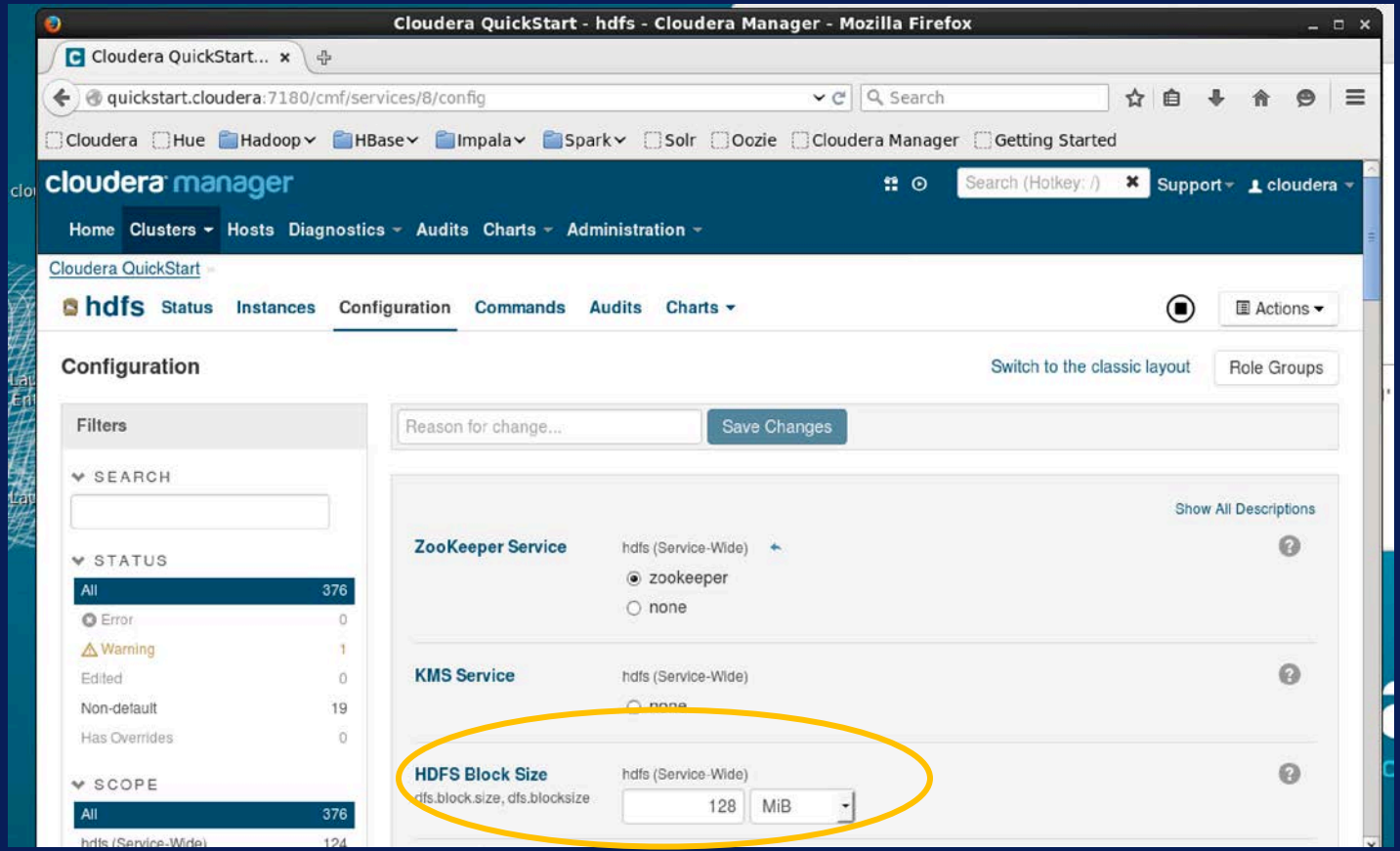
- *Parameters in hdfs-site.xml*
- Commercial vendors have GUI based management consoles to make changes.



HDFS Block Size

- Recall from previous video: *impacts NameNode memory, number of maps tasks, and hence performance.*
- *64MB is the default.* Can be changed based on workloads. Typically bumped up to 128MB.
- *dfs.blocksize, dfs.block.size*

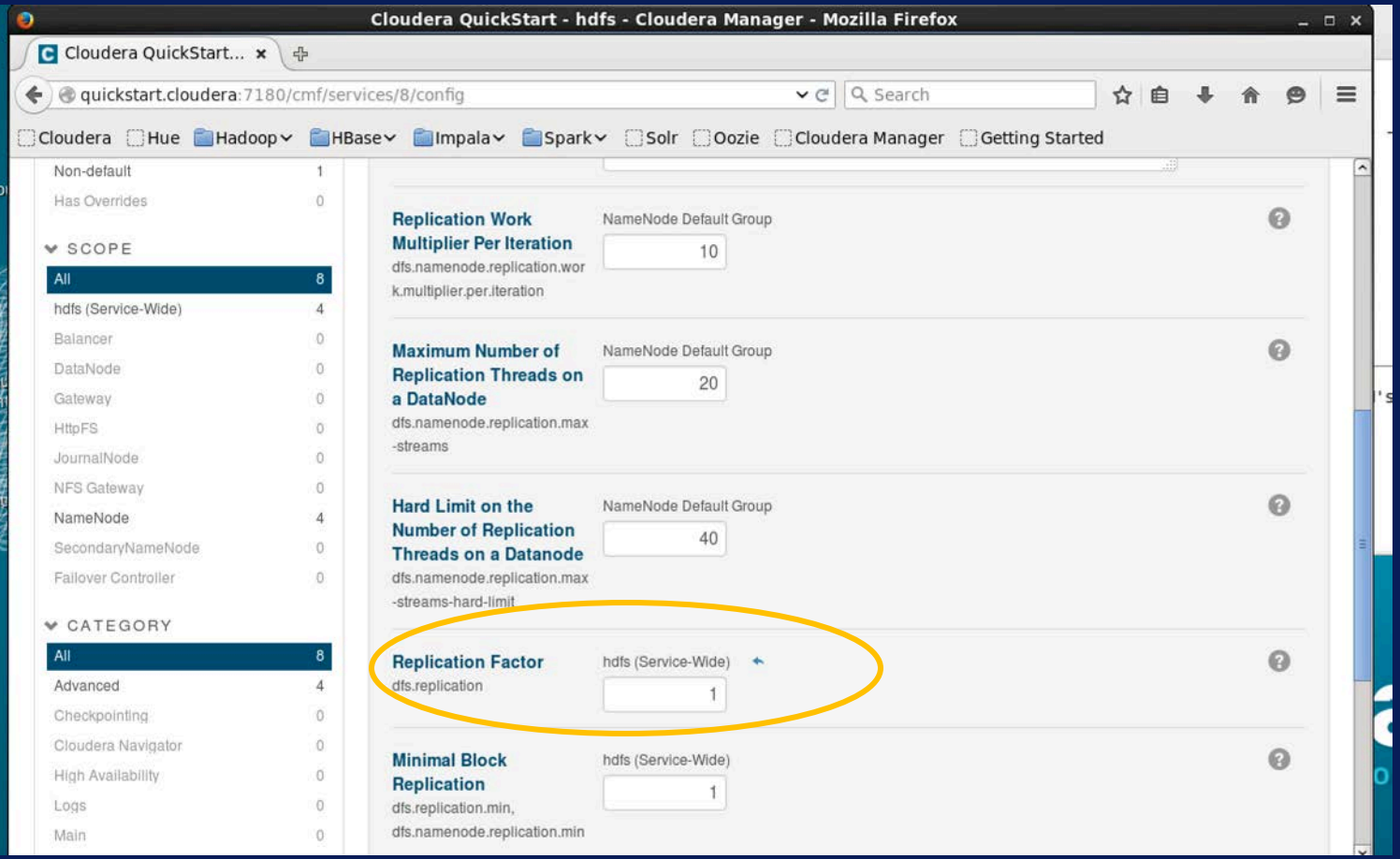
HDFS Block Size



HDFS Replication

- **Default replication is 3.**
- **Parameter: *dfs.replication***
- **Tradeoffs:**
  - *Lower it to reduce replication cost*
  - *Less robust*
  - *Higher replication can make data local to more workers*
  - *Lower replication => more space*

HDFS Replication



Lot of other parameters!

- *Various tunables for datanode, namenode.*

Examples:

- *dfs.datanode.handler.count* (10): Sets the number of server threads on each datanode.
- *dfs.namenode.fs-limits.max-blocks-per-file*: Maximum number of blocks per file.

Full list:

http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/hdfs-default.xml

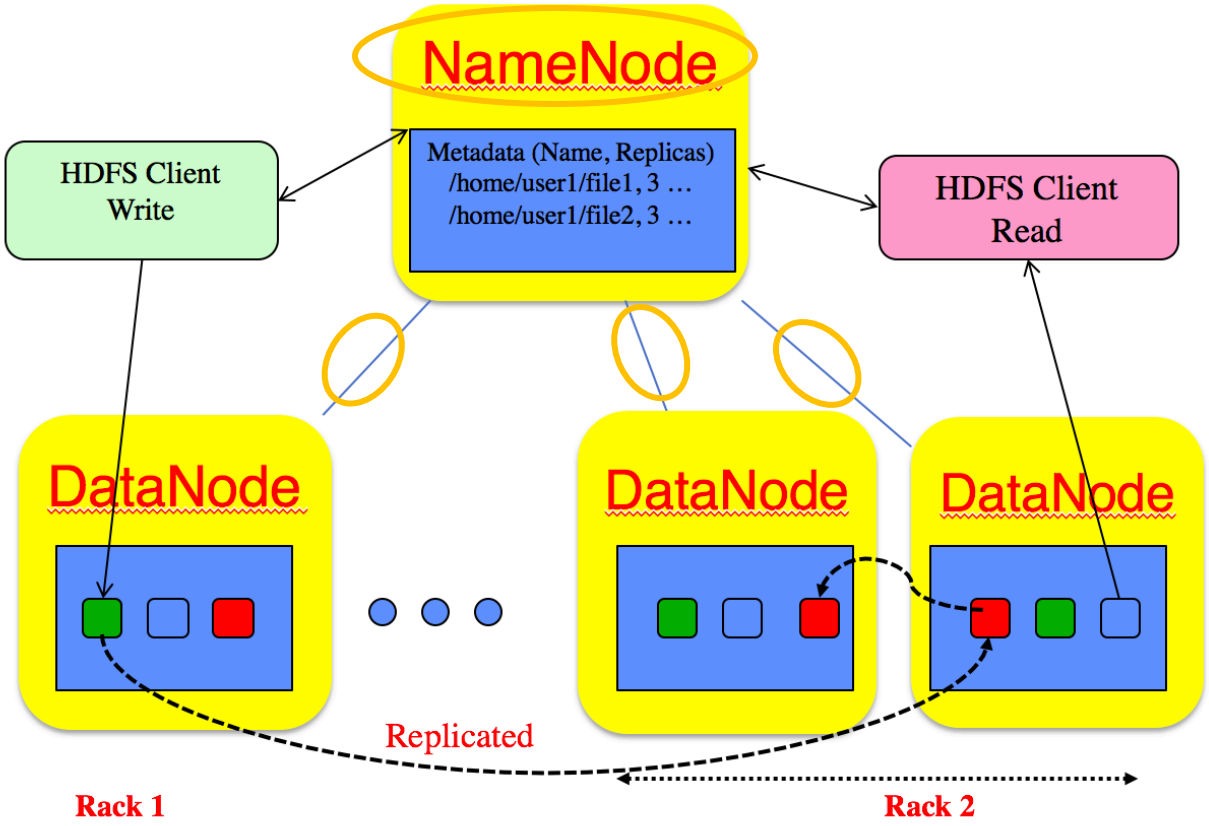
Performance & Robustness

- ***How robustness is achieved***
- ***Performance improvement and possible impact on robustness***

Common Failures

- ***DataNode Failures:*** **Server can fail, disk can crash, data corruption.**
- ***Network Failures***
- ***NameNode Failures:*** **Disk failure, node failure**

HDFS Robustness



- *NameNode receives heartbeat and block reports from DataNodes*

Mitigation of common failures

- ***Periodic heartbeat:*** **from DataNode to NameNode.**
- **DataNodes without recent heartbeat:**
  - *Marked dead, no new IO sent*
  - *Blocks below replication factor re-replicated on other nodes.*

Mitigation of common failures

- *Checksum computed* on file creation
- *Checksums stored in HDFS namespace.*
- Used to check retrieved data, *re-read from alternate replica if need.*

Mitigation of common failures

- Multiple copies of central meta data structures.
- Failover to standby NameNode – manual by default.

Performance

- **Recall***: changing blocksize and replication factor can improve performance*

Example: Distributed copy

- ***Hadoop distcp allows parallel transfer of files.***
- **This example is from one of SDSC's systems called Gordon.**
- **Copy 32 files and 512GB of data.**
- **Vary map and node counts, replication.**

Distributed Copy, Replication=3

Nodes	#Map Tasks	Copy Rate (MB/s)
8	2	183.64
16	2	220.57
16	4	276.09
16	8	387.51
32	8	618.99
32	16	**779.03**

Distributed Copy, Replication=1

Nodes	#Map Tasks	Copy Rate (MB/s)
8	4	530.66
8	8	1018.03
16	8	793.17
16	16	1814.15
32	32	**2995.93**

Replication trade off w.r.t robustness.

- **Reducing replication has a trade off w.r.t robustness:**
  - **Might lose a node or local disk** during the run – *cannot recover if there is no replication*.
  - If there is **data corruption of a block from one of the datanodes** – *again cannot recover without replication*.
- **Just one example*. Other parameter changes can have similar effects*. For example we saw with block size changes you can impact Namenode**