Introduction to Distributed File Systems and HDFS

A Few Quotes to Start Us Off

Data is a precious thing and will last longer than the systems themselves.

(Tim Berners-Lee, 2008)

Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom.

(Sir Arthur Conan Doyle)

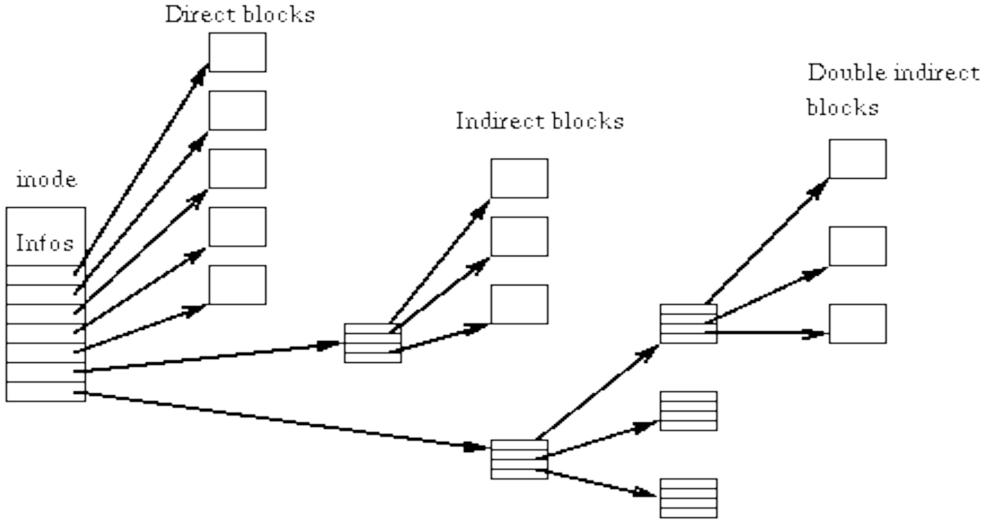
Learning Objectives

- Describe basic file system concepts
- Discuss an overview of HDFS
- Use the HDFS CLI
- Use the HDFS API
- Exporting MapR-FS with NFS

Describe Basic File System Concepts

Describe File System Concepts

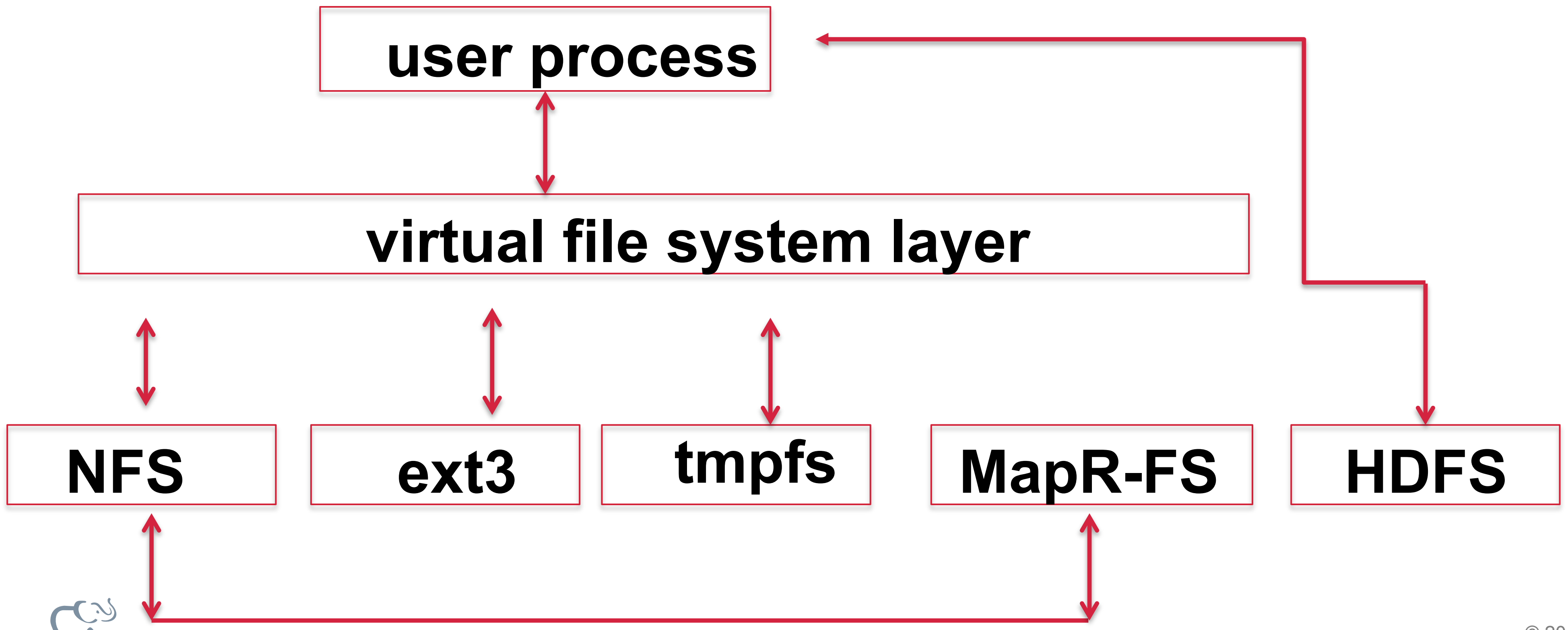
- Logical structure that organizes files on a storage medium
- Dictates how data is stored and retrieved
- Contains data and metadata



ext2 file system

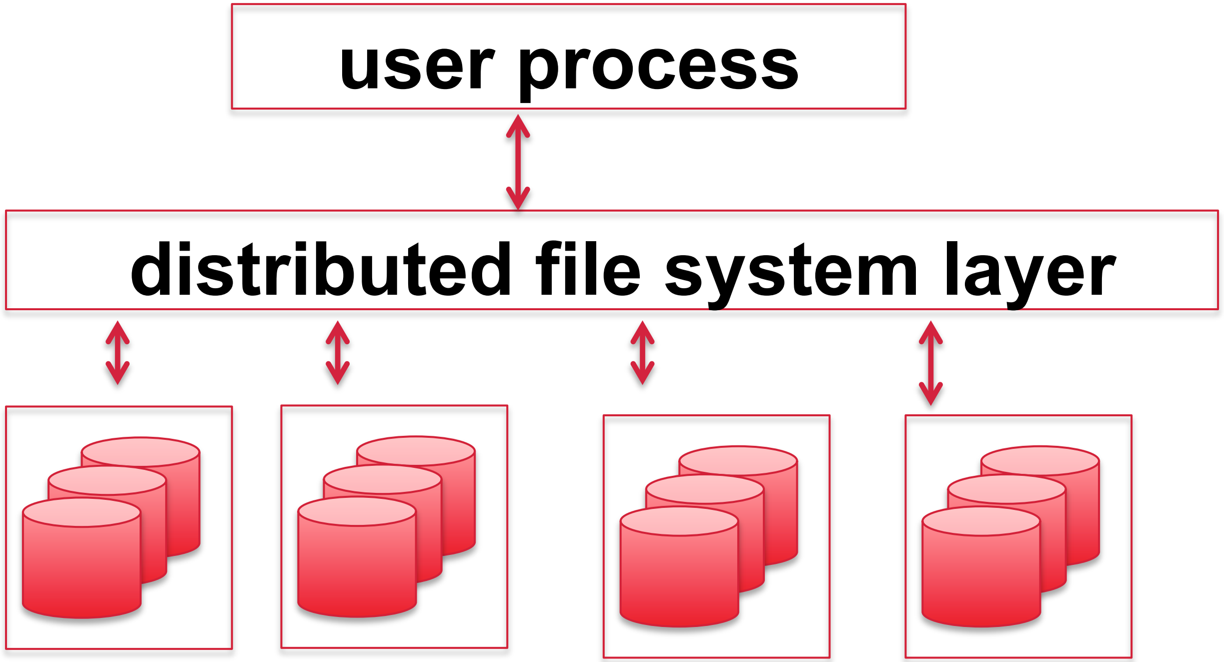
Describe the Purpose of a Virtual File System (VFS)

- Translation layer from generic file system to real file system
- Enables standard POSIX file access



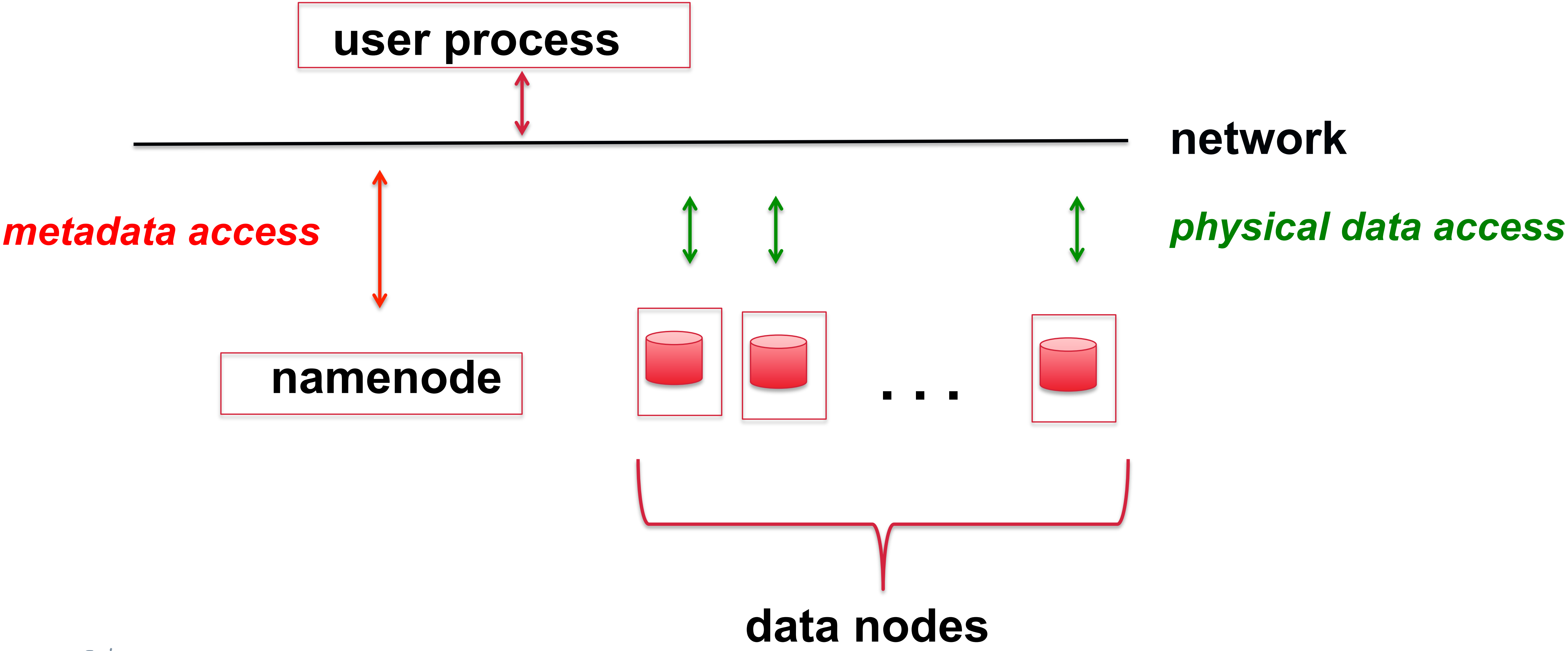
Describe Distributed File System Concepts

- Centrally stores metadata and distributes actual data
- Overcomes space, performance, and availability limitations of a single machine
- Abstracts data locality from client access

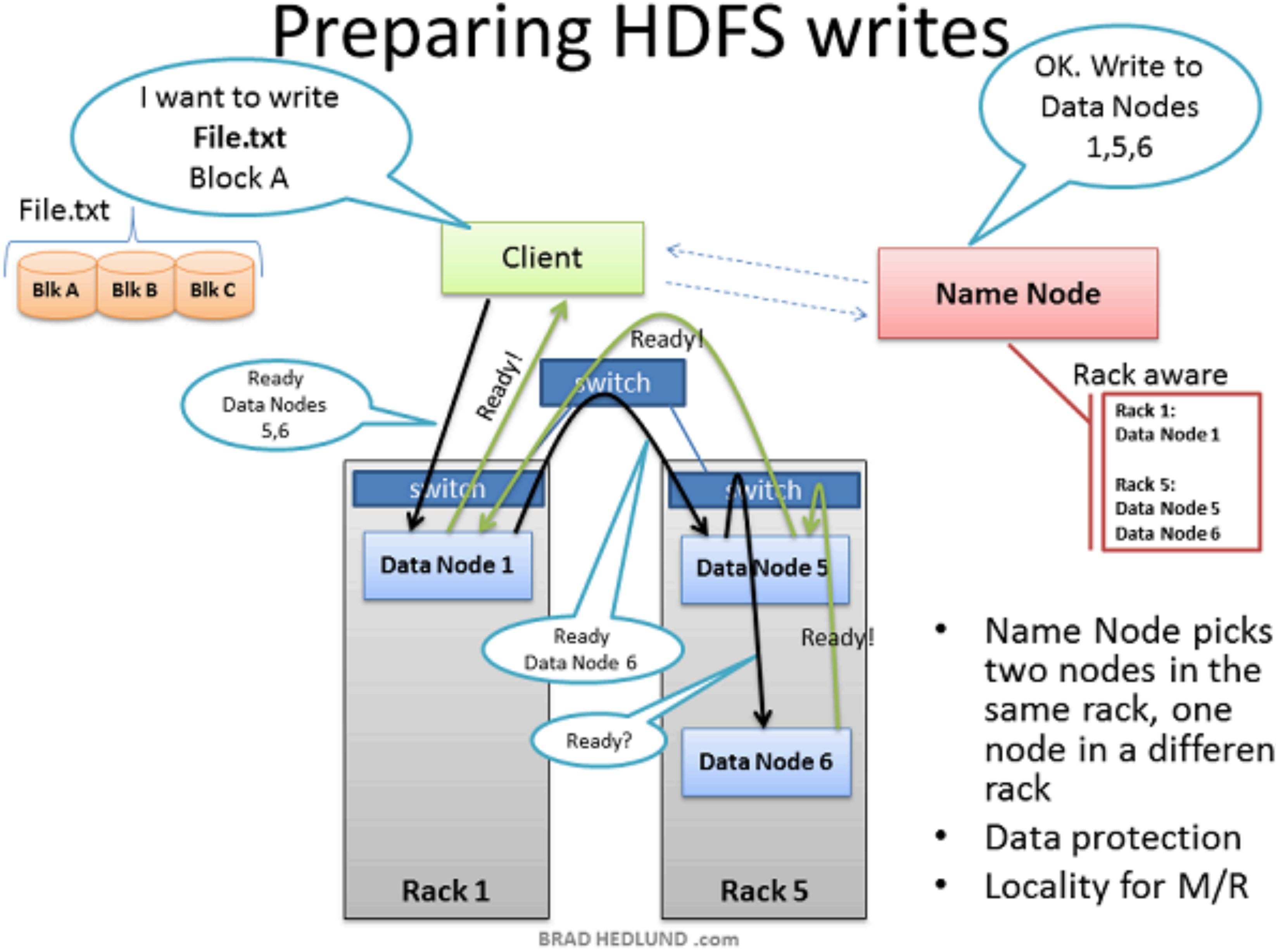


Discuss an Overview of HDFS

Describe the High-level HDFS Architecture

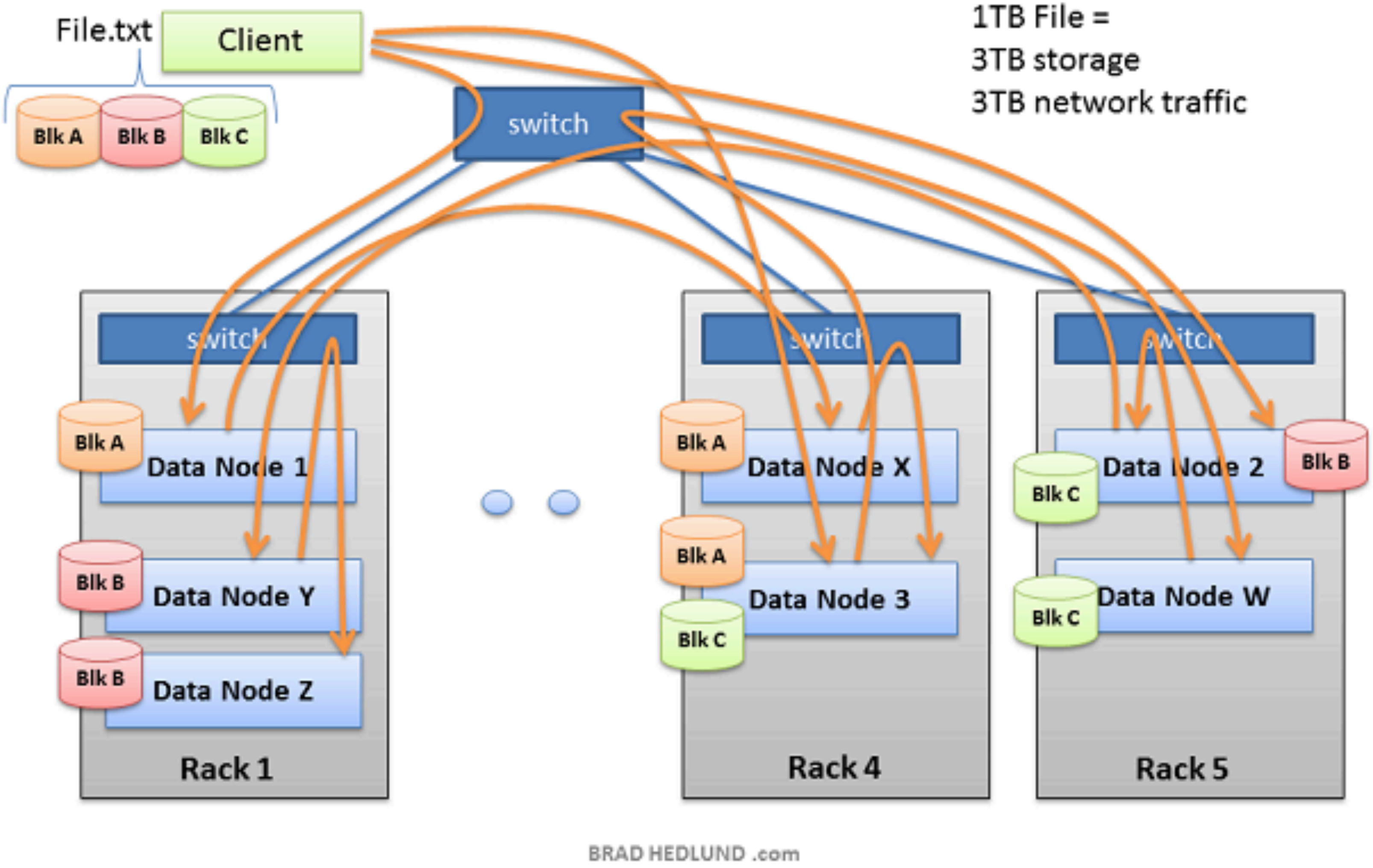


Preparing HDFS writes



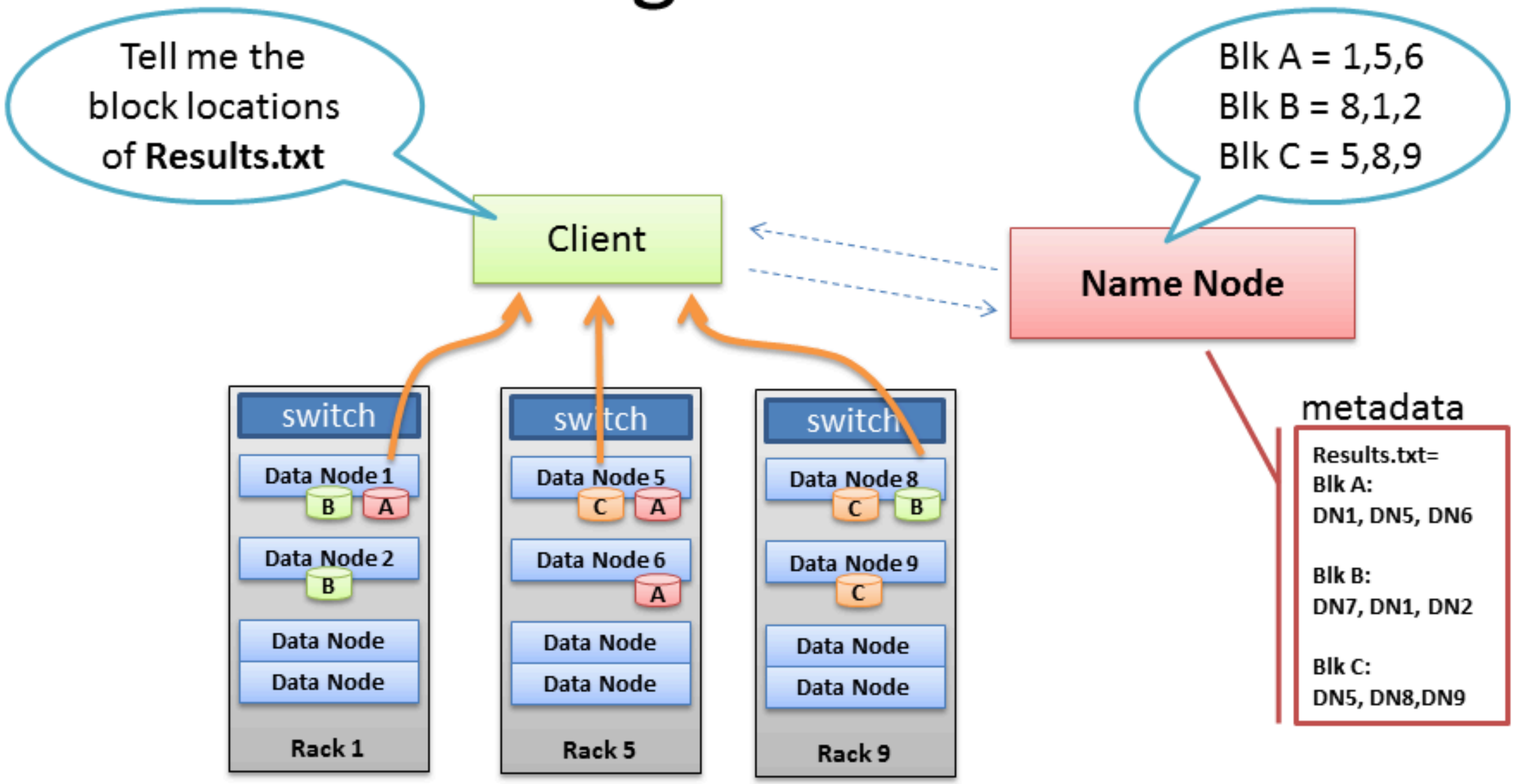
- Name Node picks two nodes in the same rack, one node in a different rack
- Data protection
- Locality for M/R

Multi-block Replication Pipeline



1TB File =
3TB storage
3TB network traffic

Client reading files from HDFS



- Client receives Data Node list for each block
- Client picks first Data Node for each block
- Client reads blocks sequentially

BRAD HEDLUND .com

Identify Limitations of HDFS

aspect	limitation
Block size	Same size used for I/O, replication, and sharding
Mutability	Write-once, read-many (WORM)
POSIX semantics	Must use HDFS API/CLI to access data
Availability	No snapshot or built-in mirroring capability
Scalability	Namenode only scales to 100M files
Performance	Written in Java and runs on block device

Cite Differences b/w MapR-FS and HDFS

aspect	feature
Block size	Different sizes used for sharding, replicating, and performing I/O
Mutability	Full read-write capability
Access	Can NFS-mount MapR-FS volumes
POSIX semantics	Can use native OS to access data
Availability	Snapshots and local/remote mirroring support
Scalability	No limit to the number of files
Performance	Written in C and runs on raw device

Using the HDFS CLI

Use the `hadoop fs` CLI

Usage: hadoop fs [command] [args]

```
hadoop fs -mkdir mydir

hadoop fs -copyFromLocal /etc/hosts mydir

hadoop fs -lsr mydir

hadoop fs -cat mydir/hosts

hadoop fs -rm mydir/hosts
```

Differentiate Absolute and Relative Paths

```
$ hadoop fs -ls /
data1 data2 tmp user var
```

```
$ hadoop conf -dump | grep fs.default.name
fs.default.name=maprfs:///
```

```
$ hadoop fs -ls
/user/jcasaletto/IN /user/jcasaletto/OUT
```

```
$ hadoop conf -dump | grep fs.mapr.working.dir
fs.mapr.working.dir=/user/$USERNAME/
```

Use the `hadoop mfs` CLI

Usage: hadoop mfs [command] [args]

```
hadoop mfs -ln mydir yourdir

hadoop mfs -setcompression off mydir

hadoop mfs -setchunksize 65536 mydir

hadoop mfs -lnh original-file new-file
```

Use the Operating System CLI

```
mkdir /user/james/mydir

cp /etc/hosts /user/james/mydir

ls -R /user/james/mydir

ln -s /user/james/mydir /user/james/yourdir

rm  /user/james/mydir/hosts

tail / grep / awk / sed
```

Using the HDFS Java API

Sample code

```
/* Copyright (c) 2009 & onwards. MapR Tech, Inc., All rights reserved */

//package com.mapr.fs;

import java.net.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;

/**
* Assumes mapr installed in /opt/mapr
*
* Compilation:
* javac -cp $(hadoop classpath) MapRTest.java
*
* Run:
* java -cp .:$(hadoop classpath) MapRTest /test
*/
```

Sample code (2)

```
public class MapRTest
{
 public static void main(String args[]) throws Exception {
        byte buf[] = new byte[ 65*1024];
        int ac = 0;
        if (args.length != 1) {
            System.out.println("usage: MapRTest pathname");
        return;
        }

        // maprfs:/// -> uses the first entry in /opt/mapr/conf/mapr-clusters.conf
        // maprfs:///mapr/my.cluster.com/
        // /mapr/my.cluster.com/

        // String uri = "maprfs:///";
        String dirname = args[ac++];

        Configuration conf = new Configuration();

        //FileSystem fs = FileSystem.get(URI.create(uri), conf); // if wanting to use a different cluster
        FileSystem fs = FileSystem.get(conf);
```

Sample Code (3)

```
 Path dirpath = new Path( dirname + "/dir");
        Path wfilepath = new Path( dirname + "/file.w");
        //Path rfilepath = new Path( dirname + "/file.r");
        Path rfilepath = wfilepath;


        // try mkdir
        boolean res = fs.mkdirs( dirpath);
        if (!res) {
                System.out.println("mkdir failed, path: " +
dirpath);
        return;
        }
```

Sample code (4)

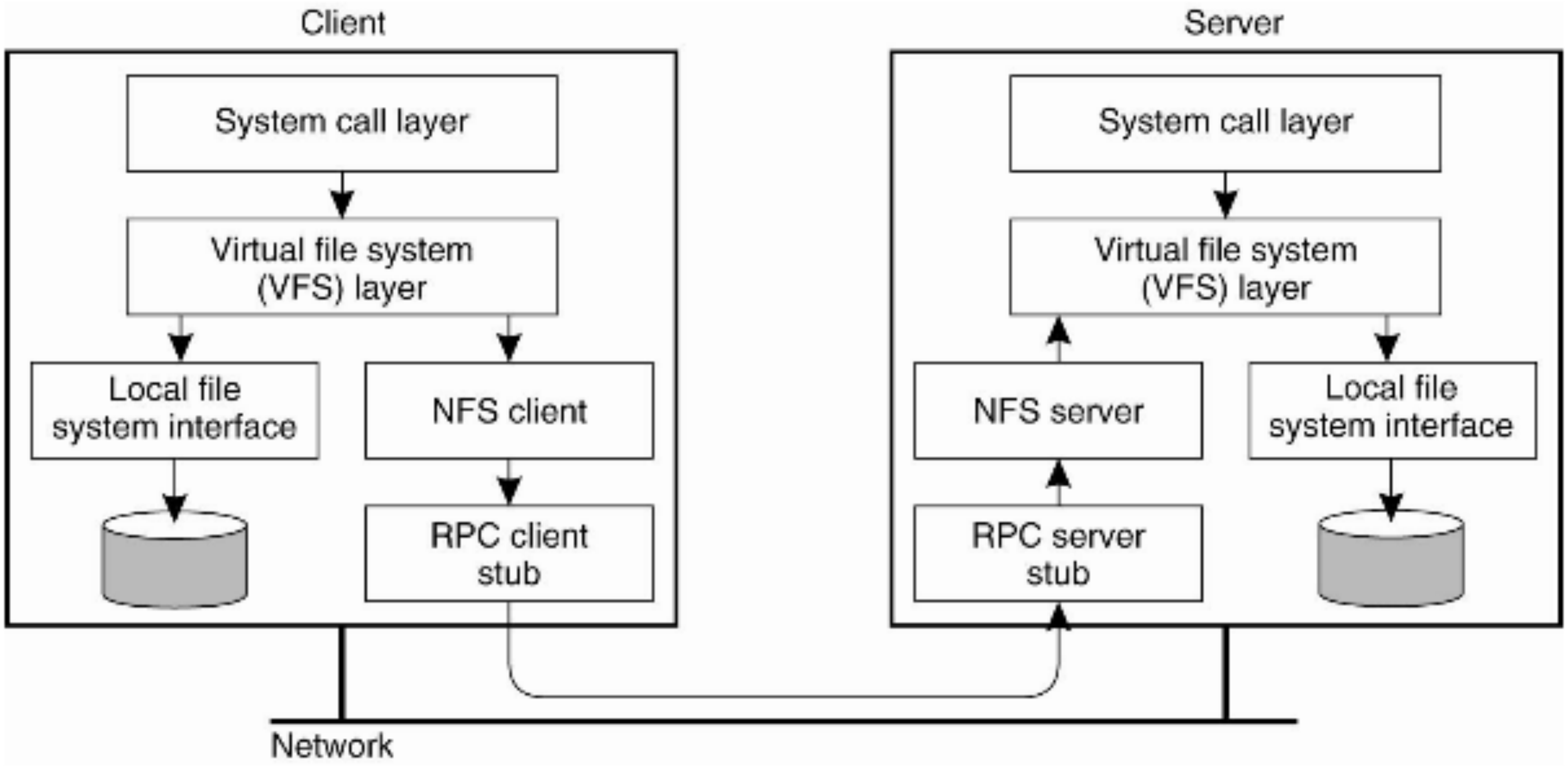
```
        System.out.println( "mkdir( " + dirpath + ") went ok, now writing file");

        // create wfile
        FSDataOutputStream ostr = fs.create( wfilepath,
                true, // overwrite
                512, // buffersize
                (short) 1, // replication
                (long)(64*1024*1024) // chunksize
                );
        ostr.write(buf);
        ostr.close();

      System.out.println( "write( " + wfilepath + ") went ok");

        // read rfile
        System.out.println( "reading file: " + rfilepath);
        FSDataInputStream istr = fs.open( rfilepath);
        int bb = istr.readInt();
        istr.close();
        System.out.println( "Read ok");
        }
}
```

Exporting MapR-FS with NFS

What is NFS?

- NFS = network file system
- Technology developed by Sun Microsystems in 1984
- Leverages VFS to present remote file system as if local
- NFS v1, v2, v3, v4 (MapR-FS supports v3)
- Runs as RPC service (MapR uses custom version of RPC)

NFS architecture


Client
System call layer
Virtual file system (VFS) layer
Local file system interface
NFS client
RPC client stub
Server
System call layer
Virtual file system (VFS) layer
NFS server
Local file system interface
RPC server stub
Network

How to configure an NFS service

- **In Linux:**
  - edit `/etc/exports`:

```
dir1 host1(option1, option2, ...) host2(option1, option2, ...)
dir2 host1(option1, option2, ...) host2(option1, option2, ...)
```

example:

```
/usr/local  192.168.0.1(ro) 192.168.0.2(ro)
/home   192.168.0.1(rw,nosuid) 192.168.0.2(rw,nosuid)
```

- **For MapR:**
  - edit `/opt/mapr/conf/exports`

How to configure an NFS client

- **In standard Linux environment:**
  - edit `/etc/fstab`

    example:

```
venus.mapr.com:/user /mnt/user nfs nolock,actimeo=0,soft 0 0
```

- **In MapR environment:**
  - edit `/opt/mapr/conf/mapr_fstab`

    example:

```
localhost:/mapr/user /user nolock,soft,intr
```