

Ghislain Fourny

Big Data for Engineers Spring 2020

4. Distributed file systems



Poll

Go *now* to:

<https://eduapp-app1.ethz.ch/>



or install EduApp 3.x



The EduApp 3.x interface is shown in five panels. The first panel shows a weekly schedule from Monday to Friday. The second panel shows a detailed view of a specific class. The third panel displays a poll question about photosynthesis. The fourth panel shows a satellite map with a red polygon and a poll question. The fifth panel shows a poll result for the photosynthesis question.

Schedule (Monday):

- 10:00 - 11:15: Chemie II
- 11:30 - 12:45: Geologie der Schweiz
- 13:15 - 14:30: Mathematik II: Analysis Biund Synt...
- 14:45 - 15:30: Dynamische Erde II
- 15:45 - 16:30: Dynamische Erde II
- 16:45 - 17:30: Dynamische Erde II

Class Detail:

Chemie II
13.15-14.30 ETZ E 9

Poll Question:

Was ist die korrekte Formel der Photosynthese?

Options:

- $\text{CO}_2 + 2\text{H}_2\text{O} \xrightarrow{\text{P}} \text{C}_6\text{H}_{12}\text{O}_6 + 2\text{H}_2\text{O}$
- $\text{CO}_2 + 2\text{H}_2\text{O} \xrightarrow{\text{P}} \text{C}_6\text{H}_{12}\text{O}_6 + 2\text{O}_2$
- Andere

Map Poll:

Was ist die korrekte Formel der Photosynthese?

$\text{CO}_2 + 2\text{H}_2\text{O} \xrightarrow{\text{P}} \text{C}_6\text{H}_{12}\text{O}_6 + 2\text{O}_2$

Poll Result:

Wählen Sie die korrekte Formel der Photosynthese.

Ergebnis: 11 von 11 (100%)

So far...

We've
rehearsed
relational
databases

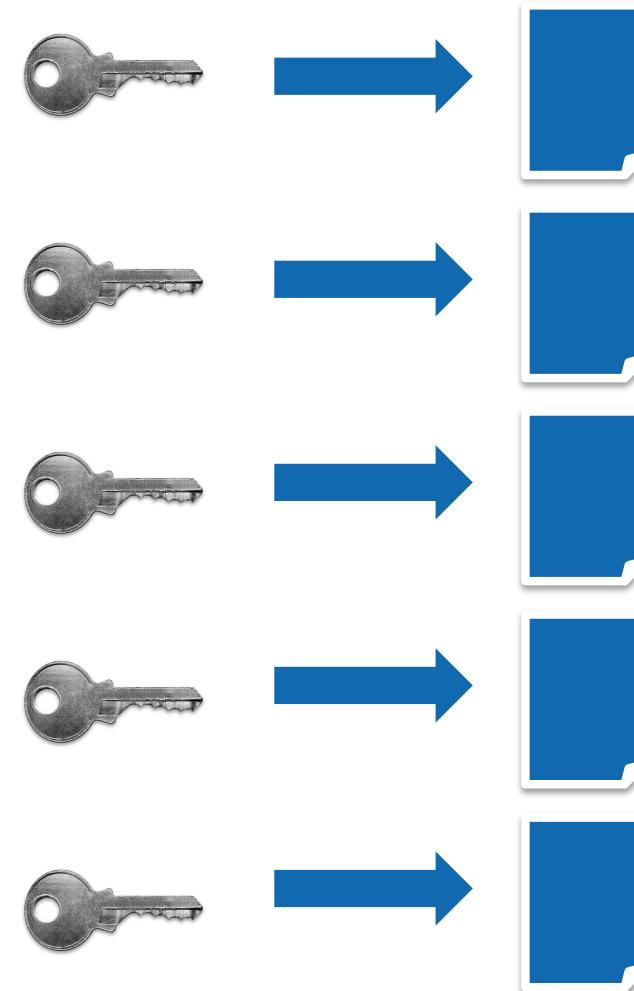
So far...

We've
looked into
scaling out



So far...

We've
seen
Object storage



Where does the data come from?

Sensors
Measurements
Events
Logs



Oleg Dudko / 123RF Stock Photo

Aggregated data
Intermediate data



Anton Starikov / 123RF Stock Photo

Raw Data

Derived Data

There is
Big Data
and
Big Data



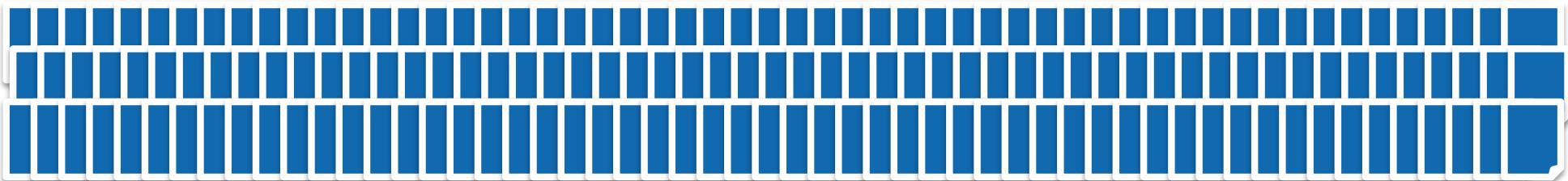
Vadym Kurgak / 123RF Stock Photo



Anna Liebiedieva / 123RF Stock Photo

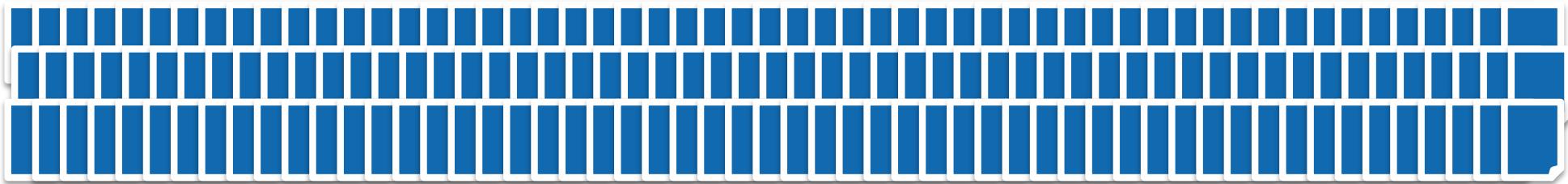
Use cases

A **huge** amount of **large** files?

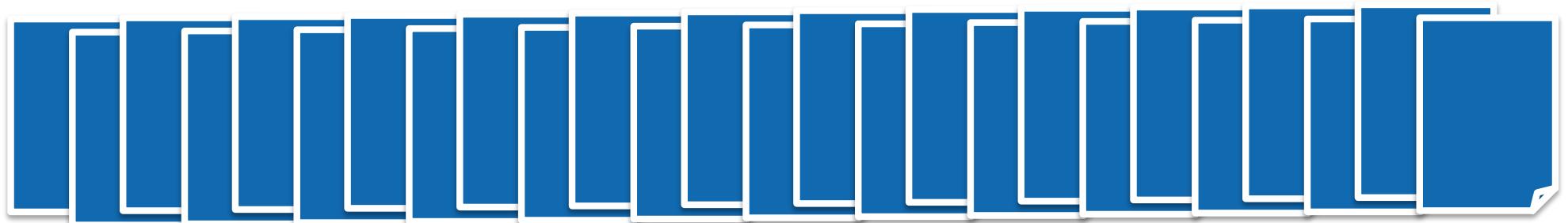


Use cases

A **huge** amount of **large** files?



vs.



A **large** amount of **huge** files?

Use cases

Billions of TB files



vs.



Millions of PB files

Technologies and models

Key-Value Model

Object Storage

Billions of
<TB files

VS.

Millions of
<PB files

Technologies and models

Key-Value Model

Object Storage

File System

Block Storage

Billions of
<TB files

VS.

Millions of
<PB files

Distributed file systems: inception

Google FS

Fault tolerance and robustness



Local disk

It **might** fail

Vitaly Korovin / 123RF Stock Photo

Fault tolerance and robustness



Local disk

Vitaly Korovin / 123RF Stock Photo



Cluster with 100s to 10,000s of machines

Kheng Ho Toh / 123RF Stock Photo

It **might** fail

nodes **will** fail

Fault tolerance and robustness

Monitoring



Fault tolerance and robustness

Error detection

Monitoring



Fault tolerance and robustness

Automatic Recovery

Error detection

Monitoring



Fault tolerance and robustness

Fault tolerance

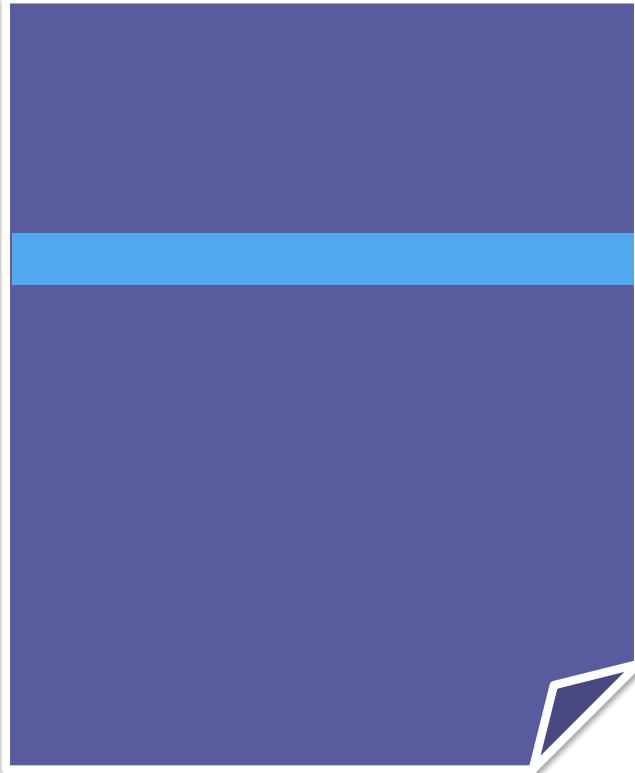
Automatic Recovery

Error detection

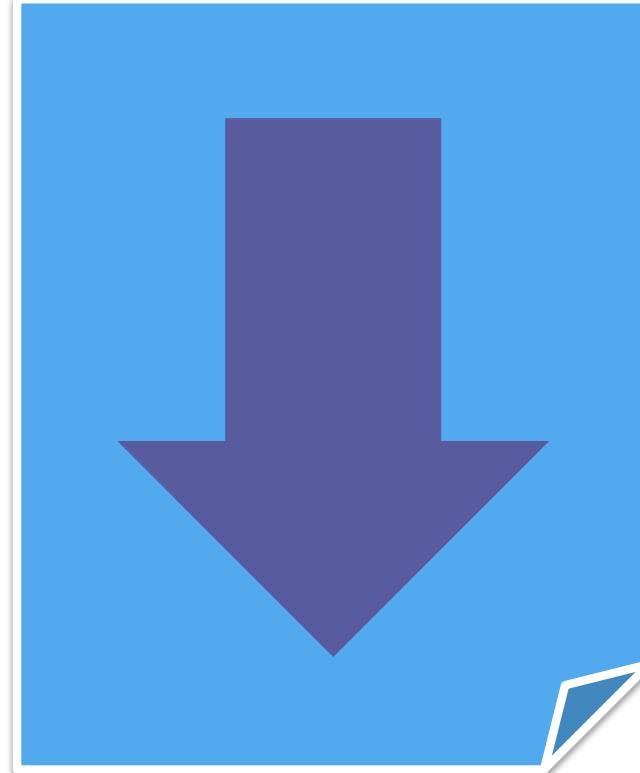
Monitoring



File read model



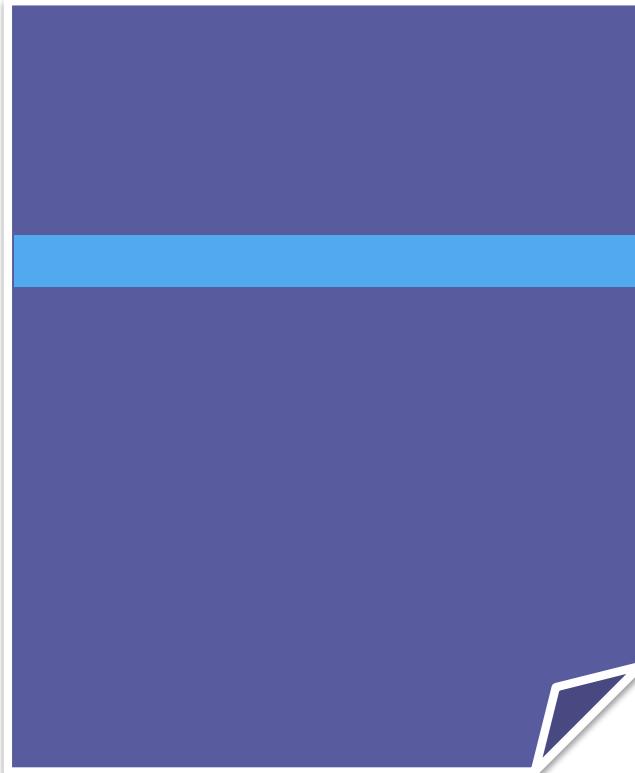
vs.



Random access

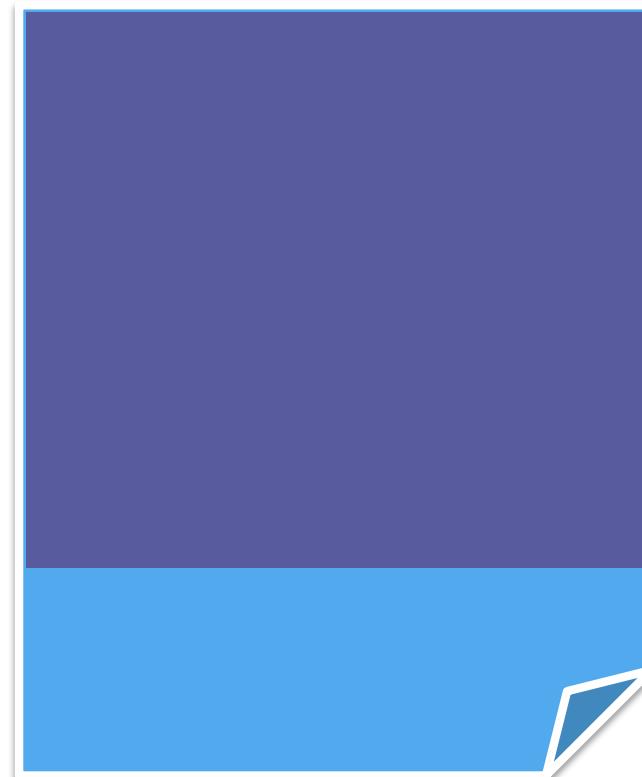
Scan the file

File update model



Random access

vs.



Append

File update model

suitable for

Sensors



Logs

Intermediate data



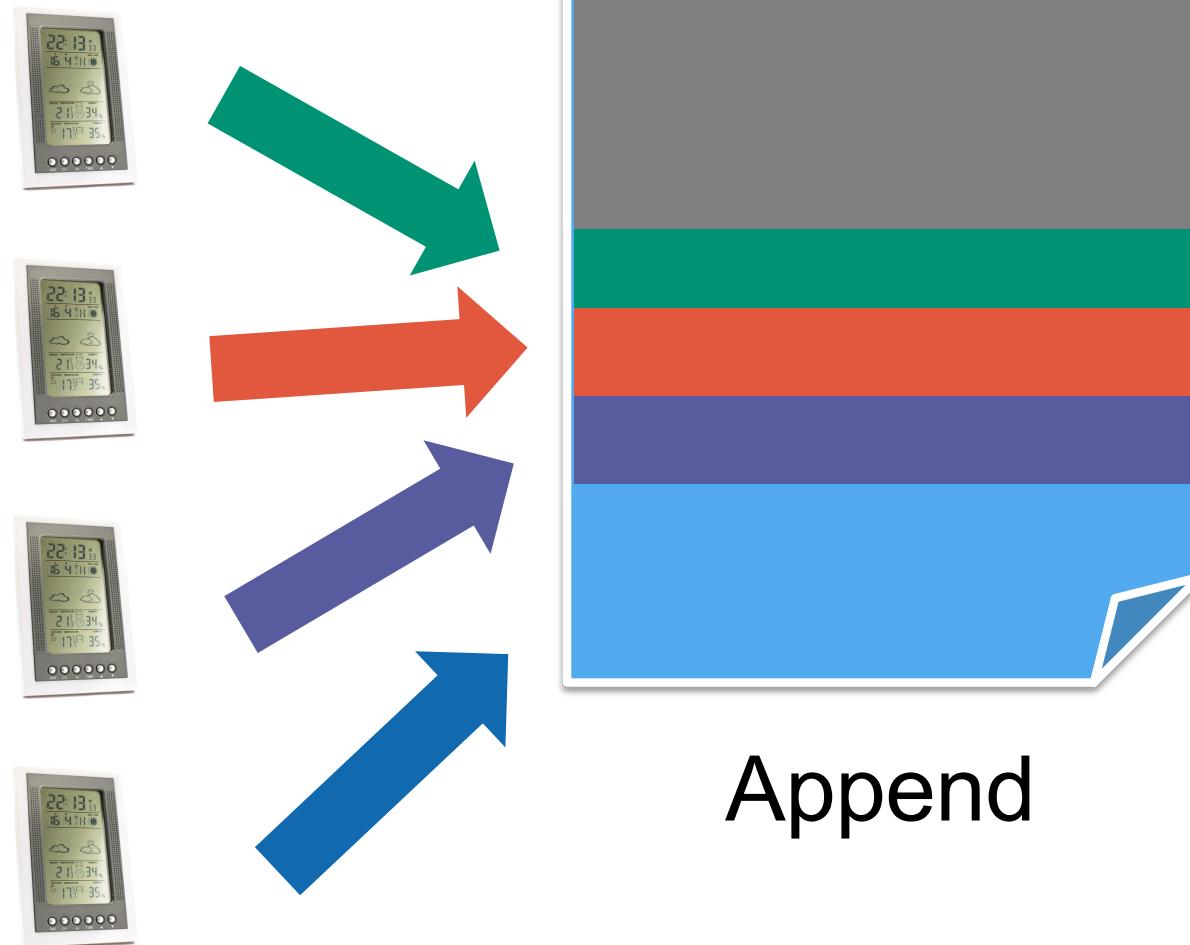
immutable

Append

Appends

100s of clients
in parallel

atomic



Performance requirements

Top priority:

Throughput



Performance requirements

Top priority:

Throughput



Secondary:

Latency



The progress made (1956-2020): Logarithmic

200,000,000,000x



Capacity
(per unit of volume)

10,000x



Throughput

8x



Latency

The progress made (1956-2020): Logarithmic

200,000,000,000x



Capacity
(per unit of volume)

10,000x



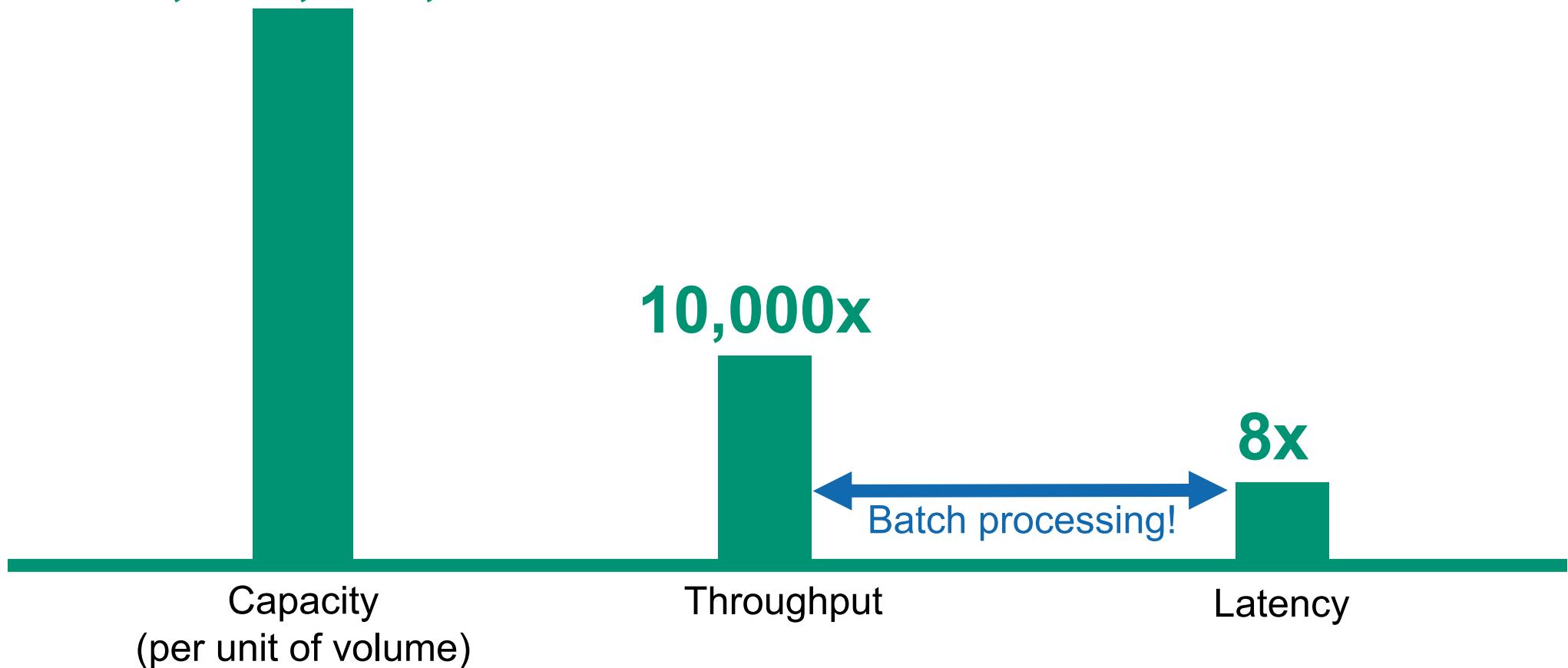
Throughput

8x

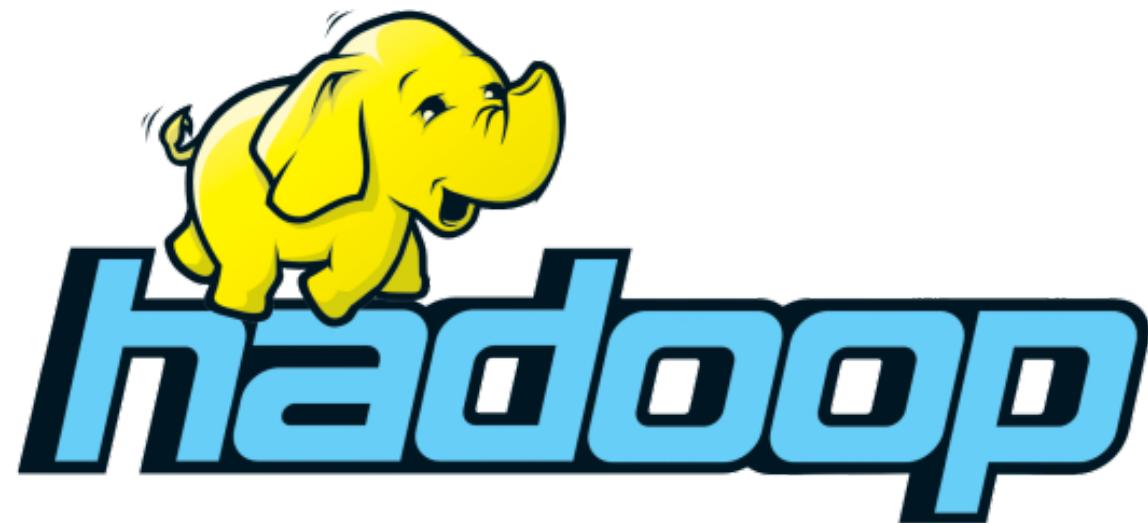
Latency

The progress made (1956-2020): Logarithmic

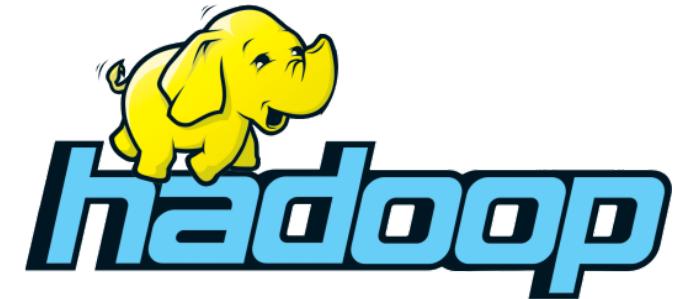
200,000,000,000x



Hadoop



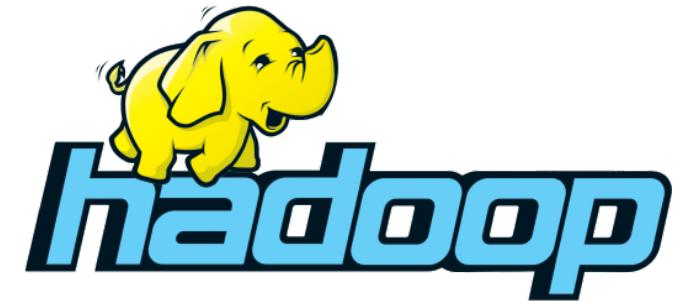
Hadoop



Initiated in
2006

Covered in this lecture

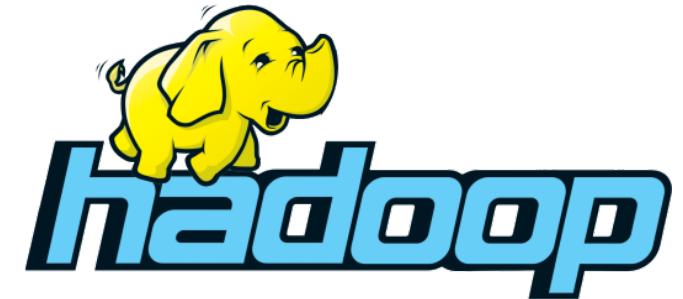
Hadoop



Primarily:

- Distributed File System (HDFS)
- MapReduce
- Wide column store (HBase)

Hadoop



Inspired by Google's

- GFS (2003)
- MapReduce (2004)
- BigTable (2006)

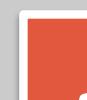
Size timeline

Date	Size reported by Yahoo
April 2006	188
May 2006	300
October 2006	600
April 2007	1,000
February 2008	10,000 (index generation)
March 2009	24,000 (17 clusters)
June 2011	42,000 (100+ PB)
November 2016	100,000? (600PB)

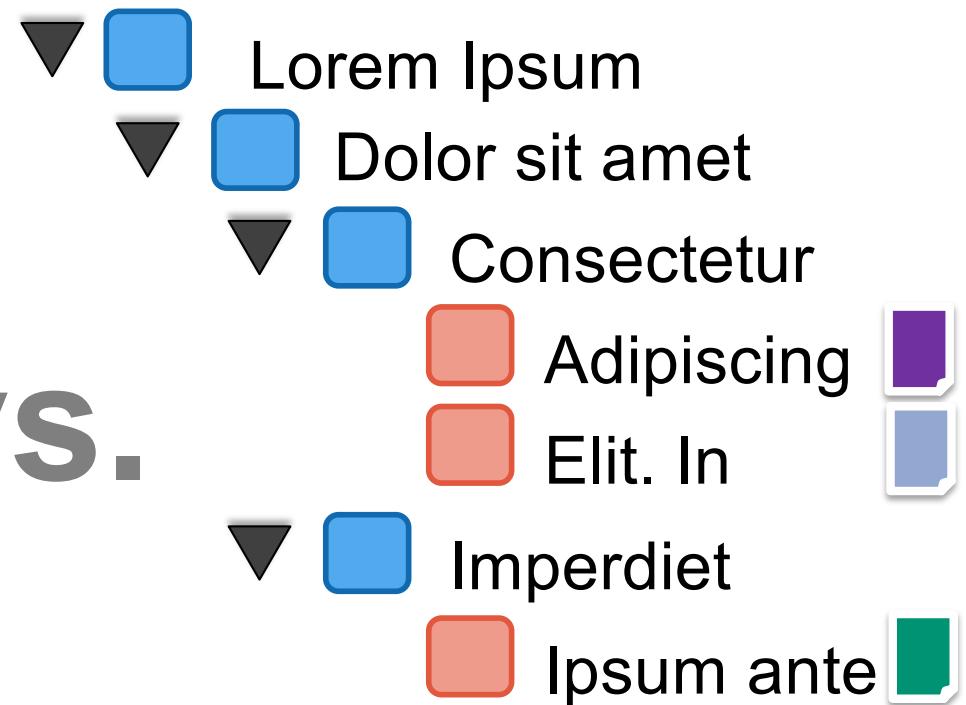


Distributed file systems: the model

File Systems (Logical Model)

Lorem Ipsum	
Dolor sit amet	
Consectetur	
Adipiscing	
Elit. In	
Imperdiet	
Ipsum ante	

vs.



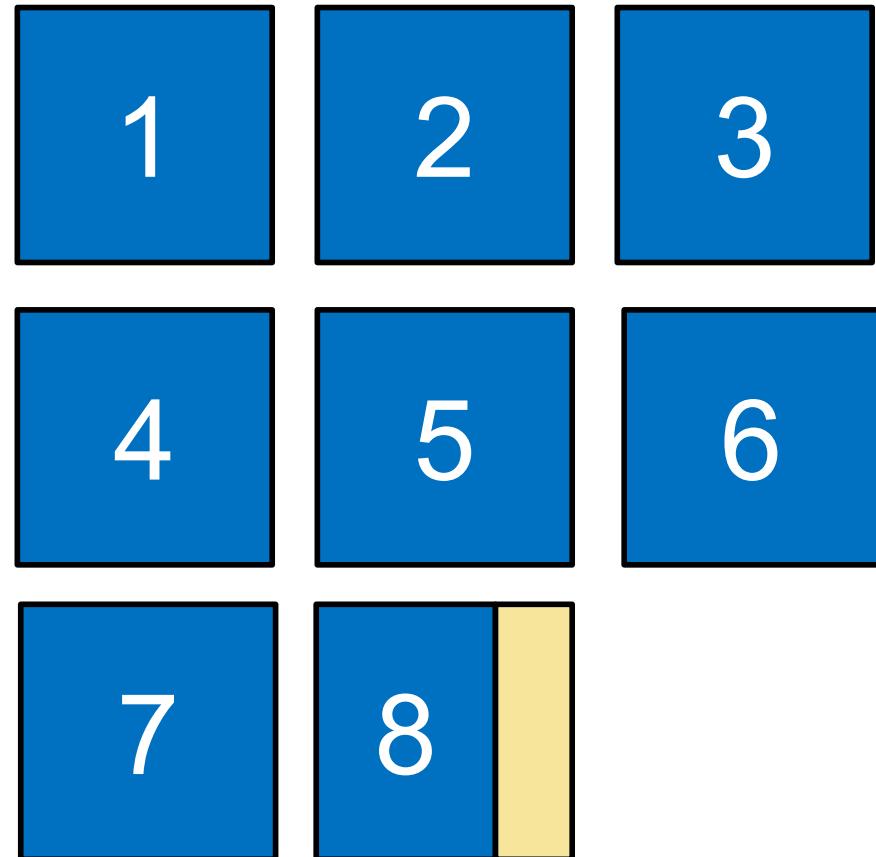
Key-Value Model

File Hierarchy

Block Storage (Physical Storage)



vs.



Object Storage

Block Storage

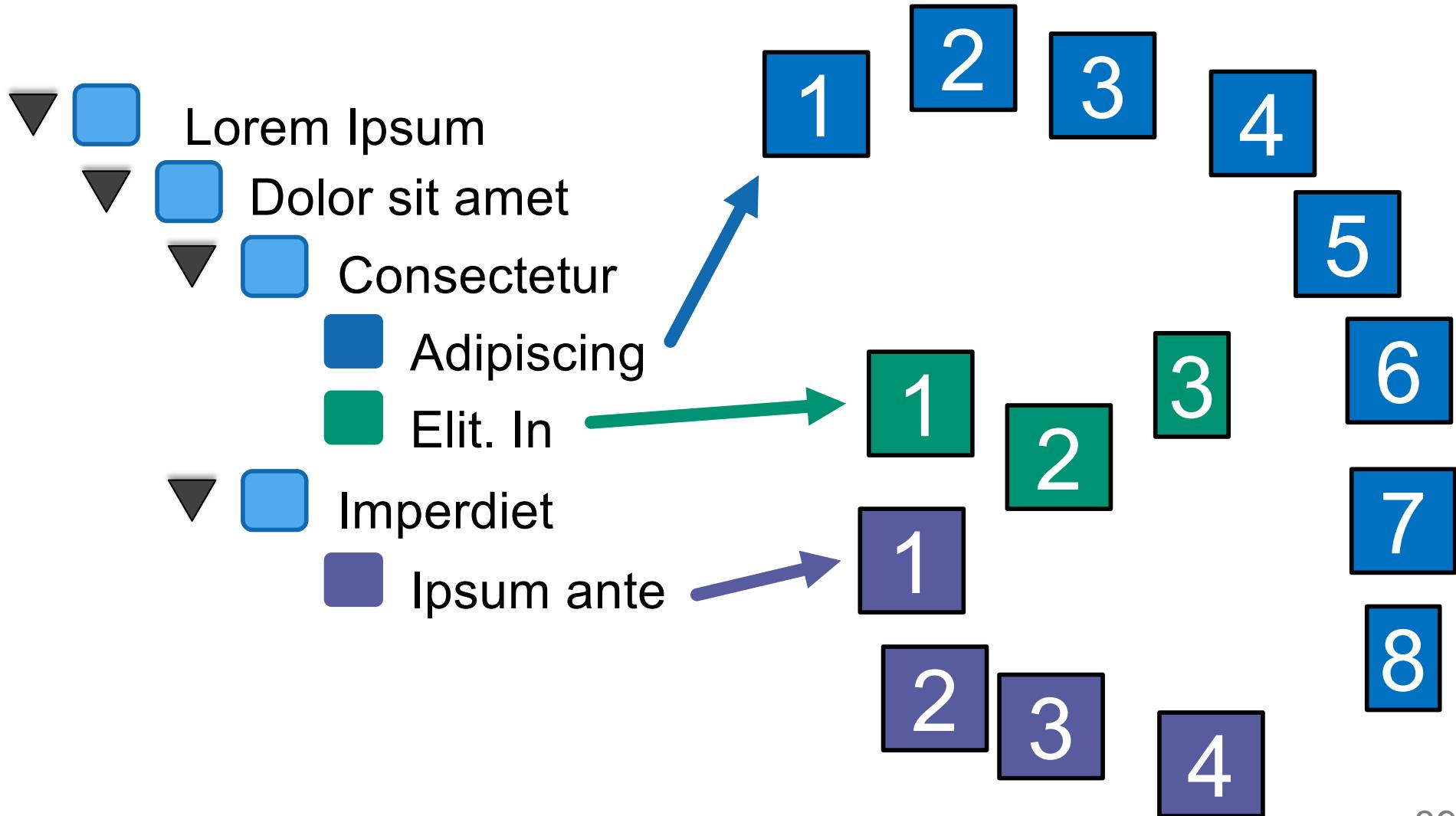
Terminology

HDFS: Block
GFS: Chunk

Files and blocks

- ▼  Lorem Ipsum
 - ▼  Dolor sit amet
 - ▼  Consectetur
 -  Adipiscing
 -  Elit. In
 - ▼  Imperdiet
 -  Ipsum ante

Files and blocks



Why blocks?

Why blocks?

1. Files bigger than a disk

PBs!

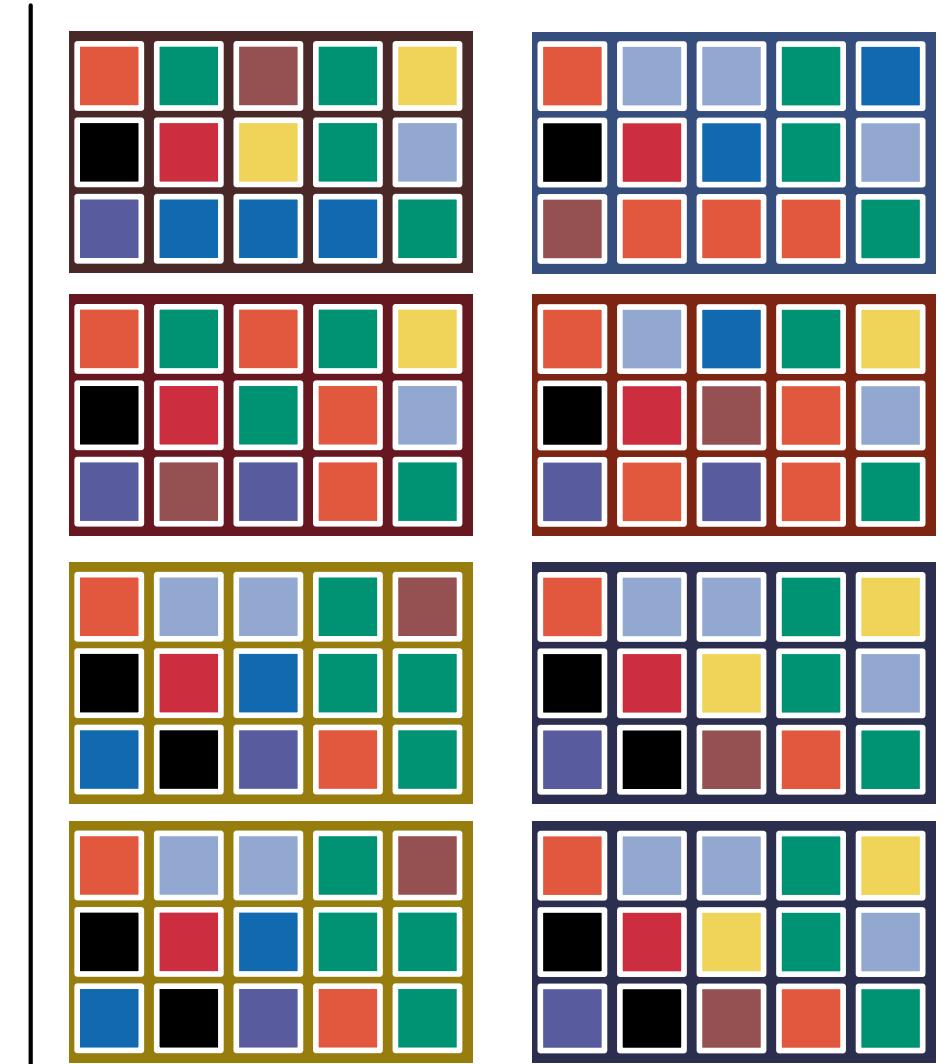
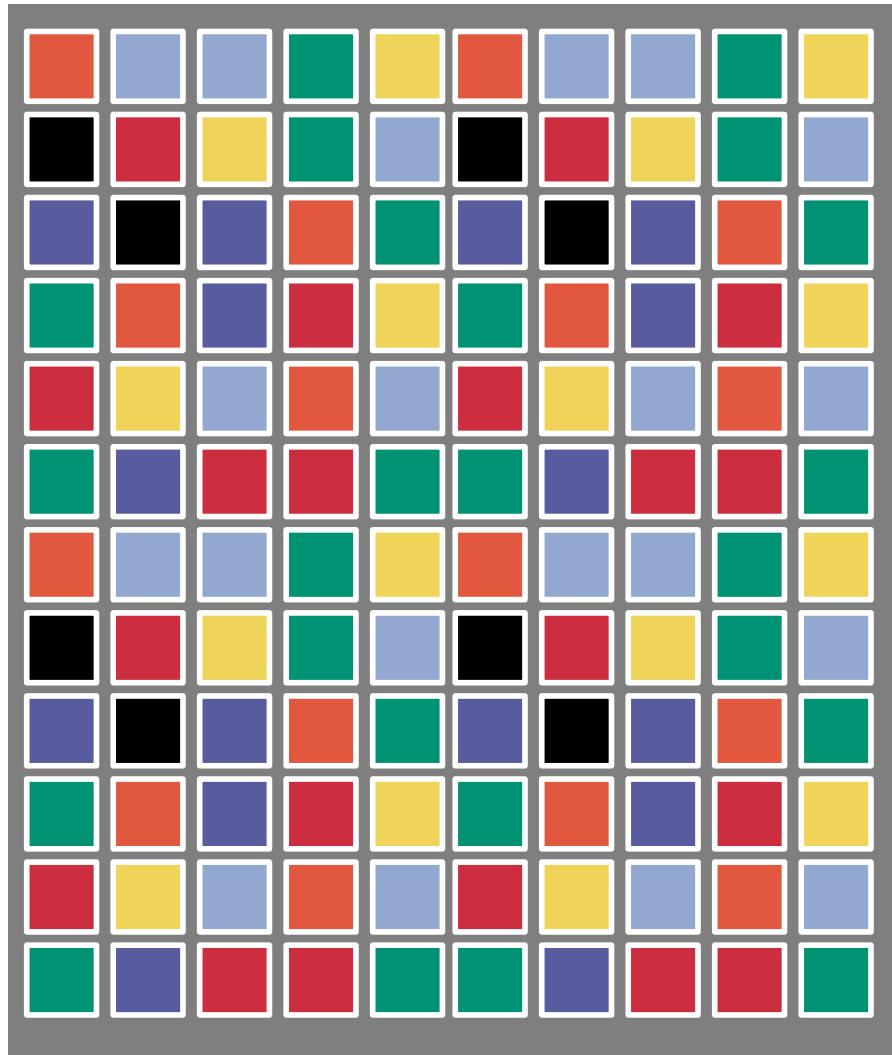
Why blocks?

1. Files bigger than a disk

PBs!

2. Simpler level of abstraction

Single machine vs. distributed



The right block size

■
4 kB

Simple file system

Poll

Go *now* to:

<https://eduapp-app1.ethz.ch/>



or install EduApp 3.x



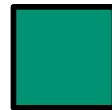
The EduApp 3.x interface consists of five panels. The leftmost panel shows a weekly schedule with various subjects like Chemistry II, Geologie der Schweiz, and Mathematik II. The second panel shows a detailed view of a Chemistry II class. The third panel displays a question about photosynthesis with a map overlay. The fourth panel shows another question about photosynthesis. The rightmost panel is a summary or results screen.

The right block size



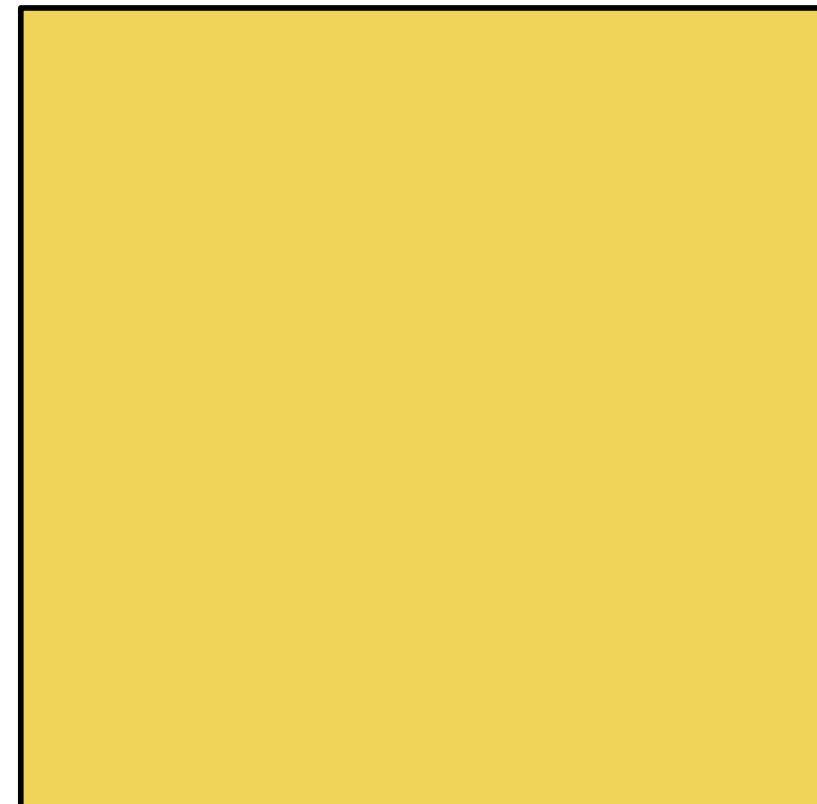
4 kB

Simple file system



4 kB – 32 kB

Relational Database



64 MB – 128 MB

Distributed file system

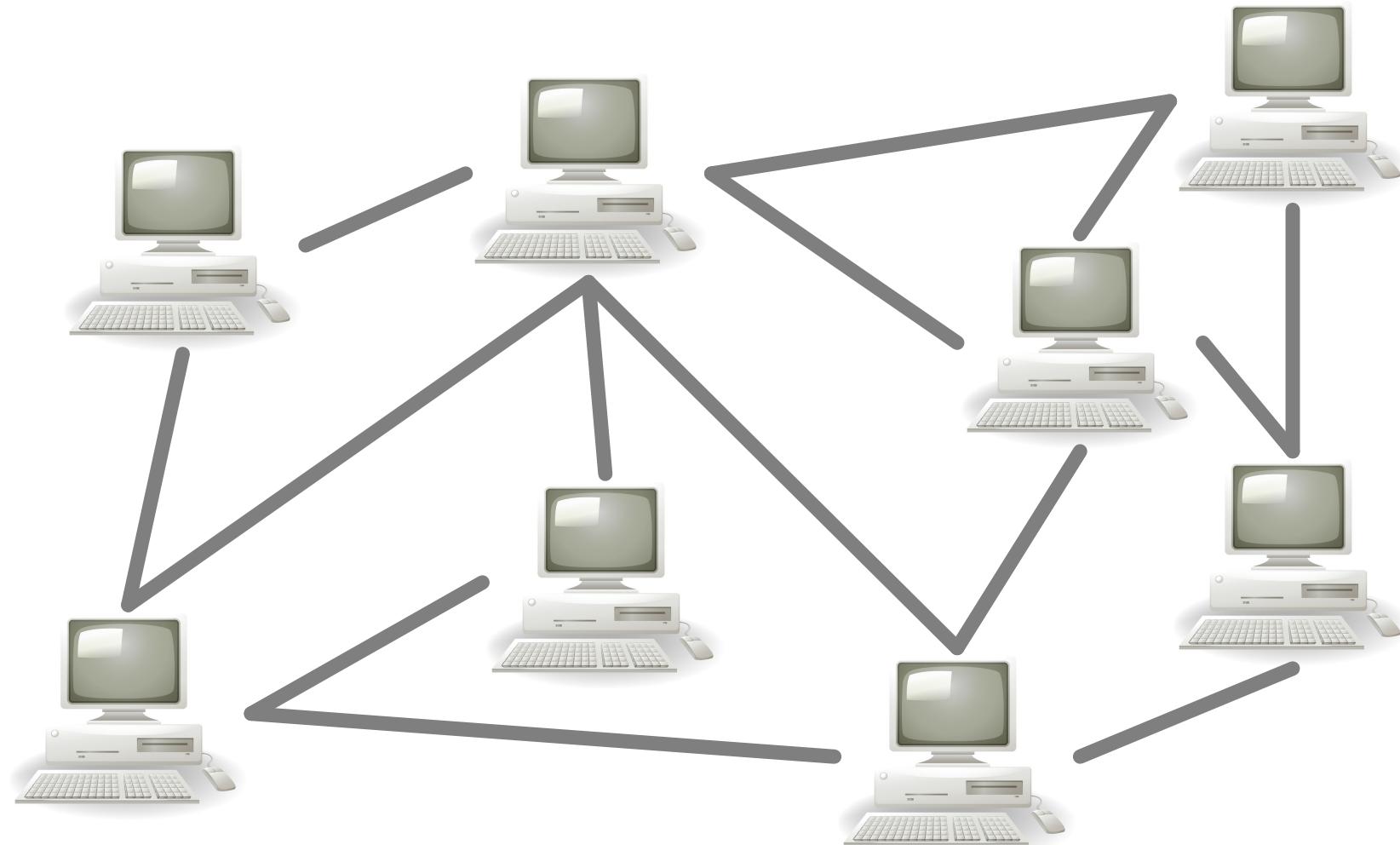


HDFS Architecture

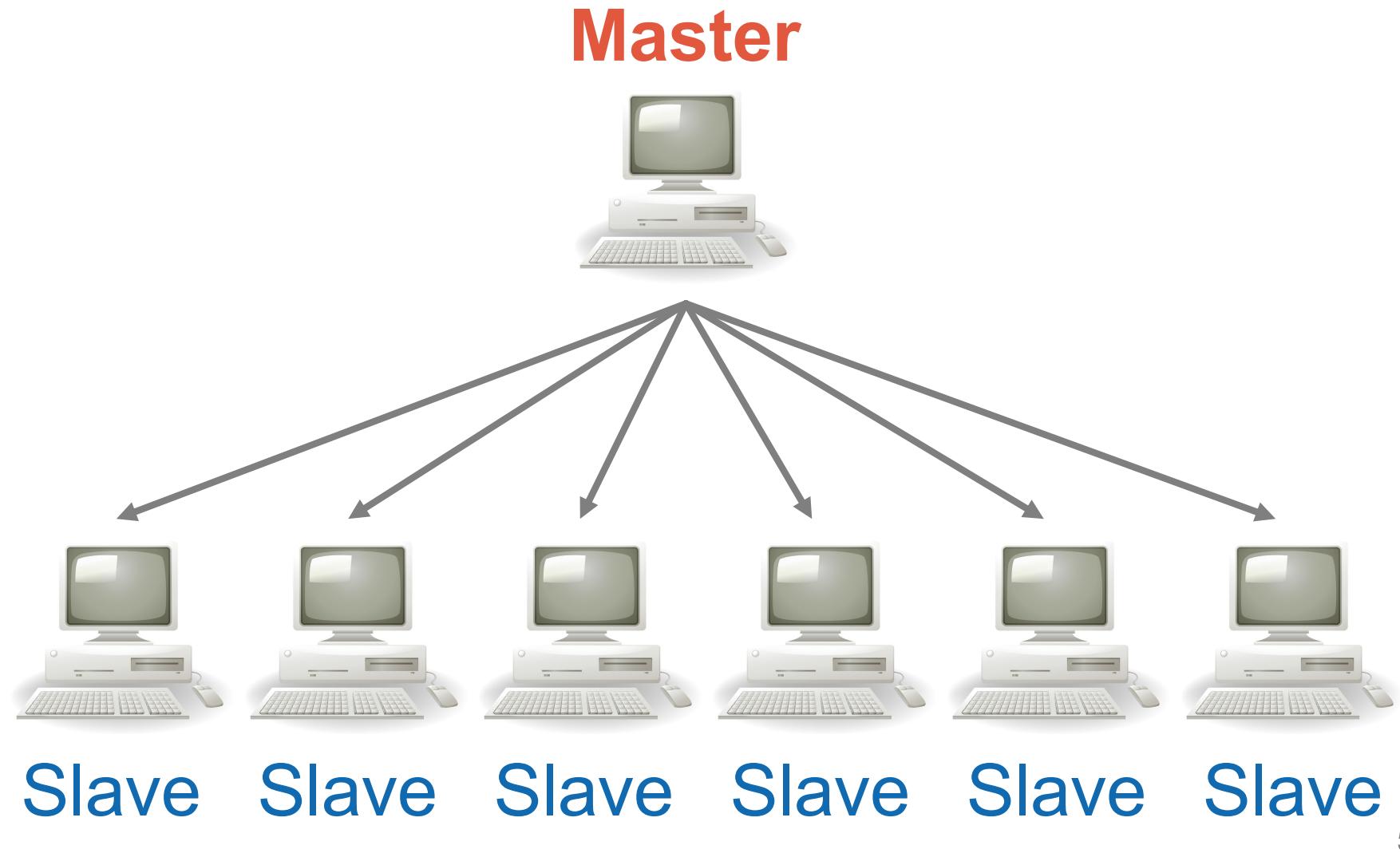
How do we connect the many machines?



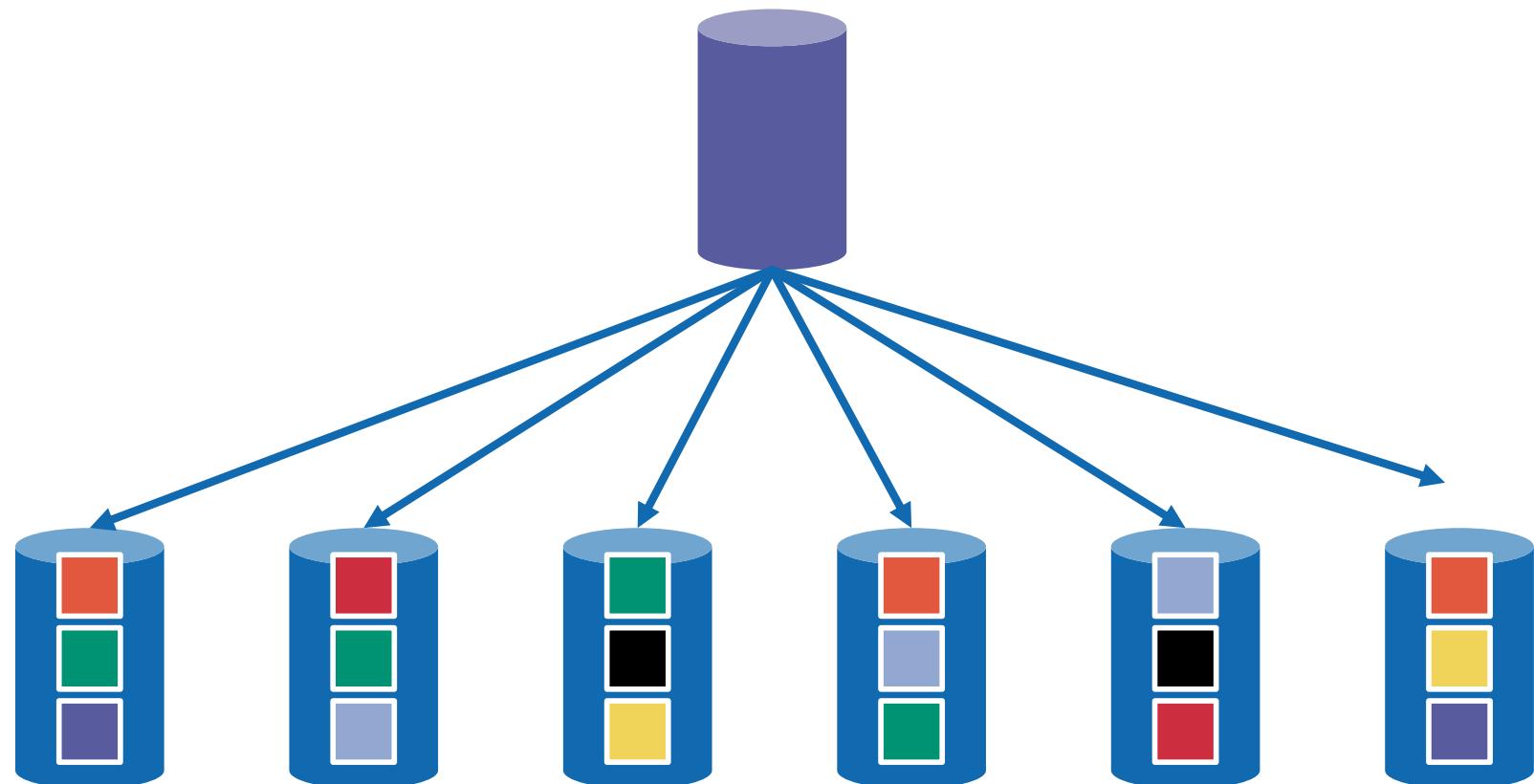
Peer-to-peer architecture



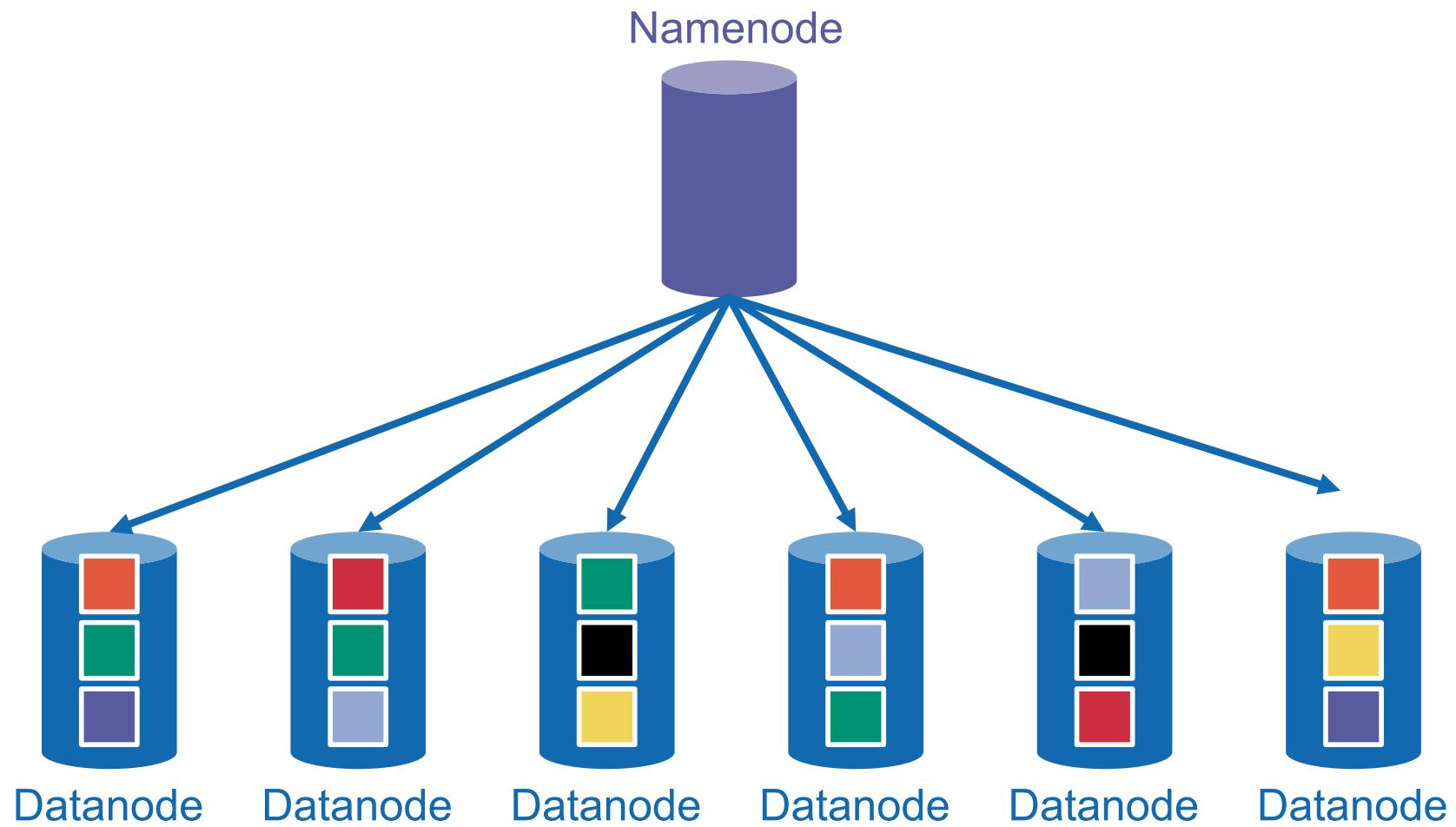
Master-slave architecture



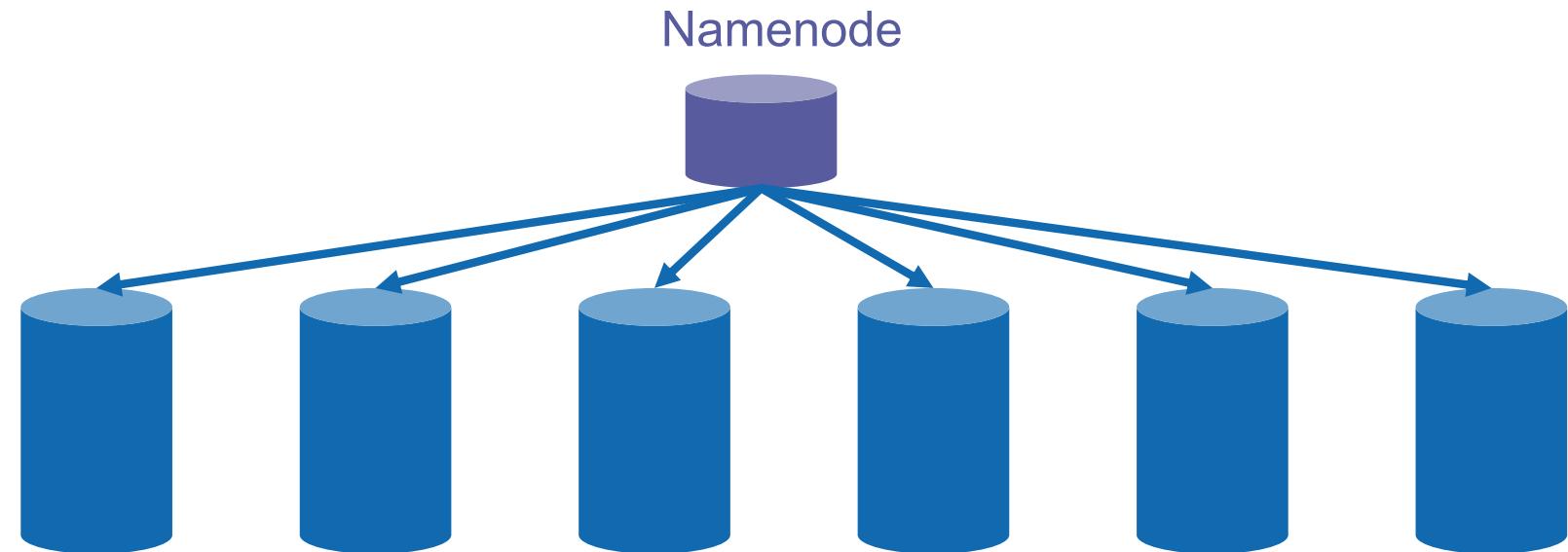
HDFS server architecture



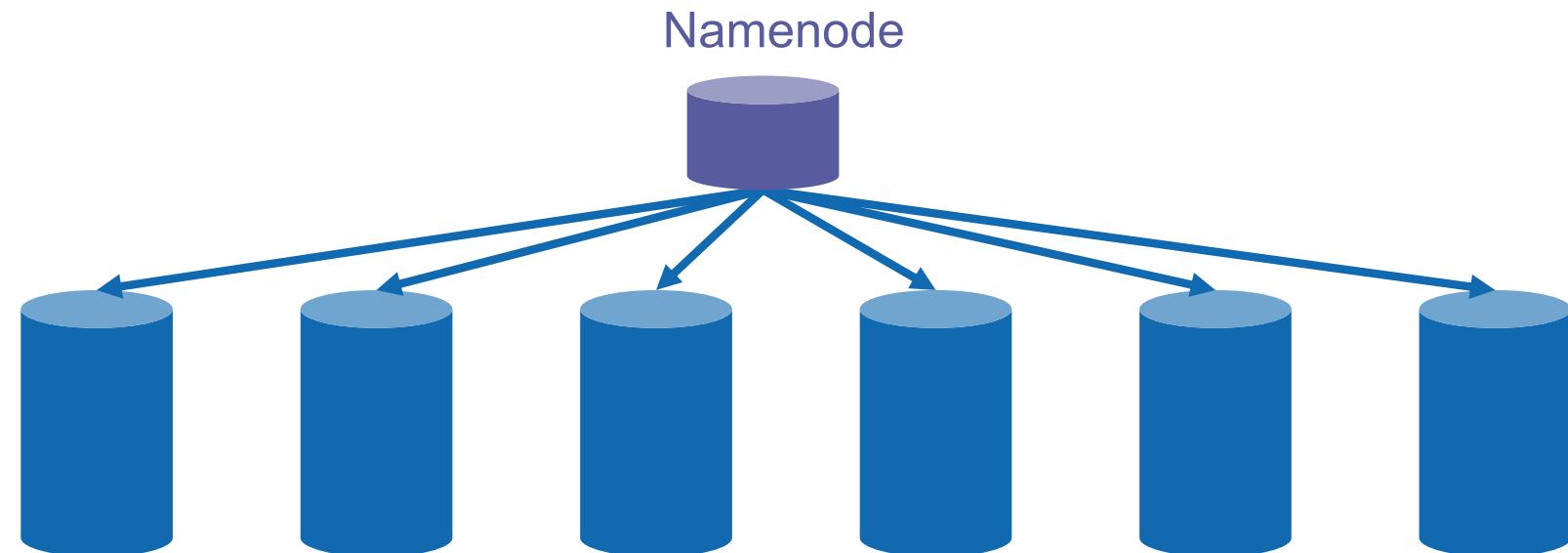
HDFS server architecture



From the file perspective



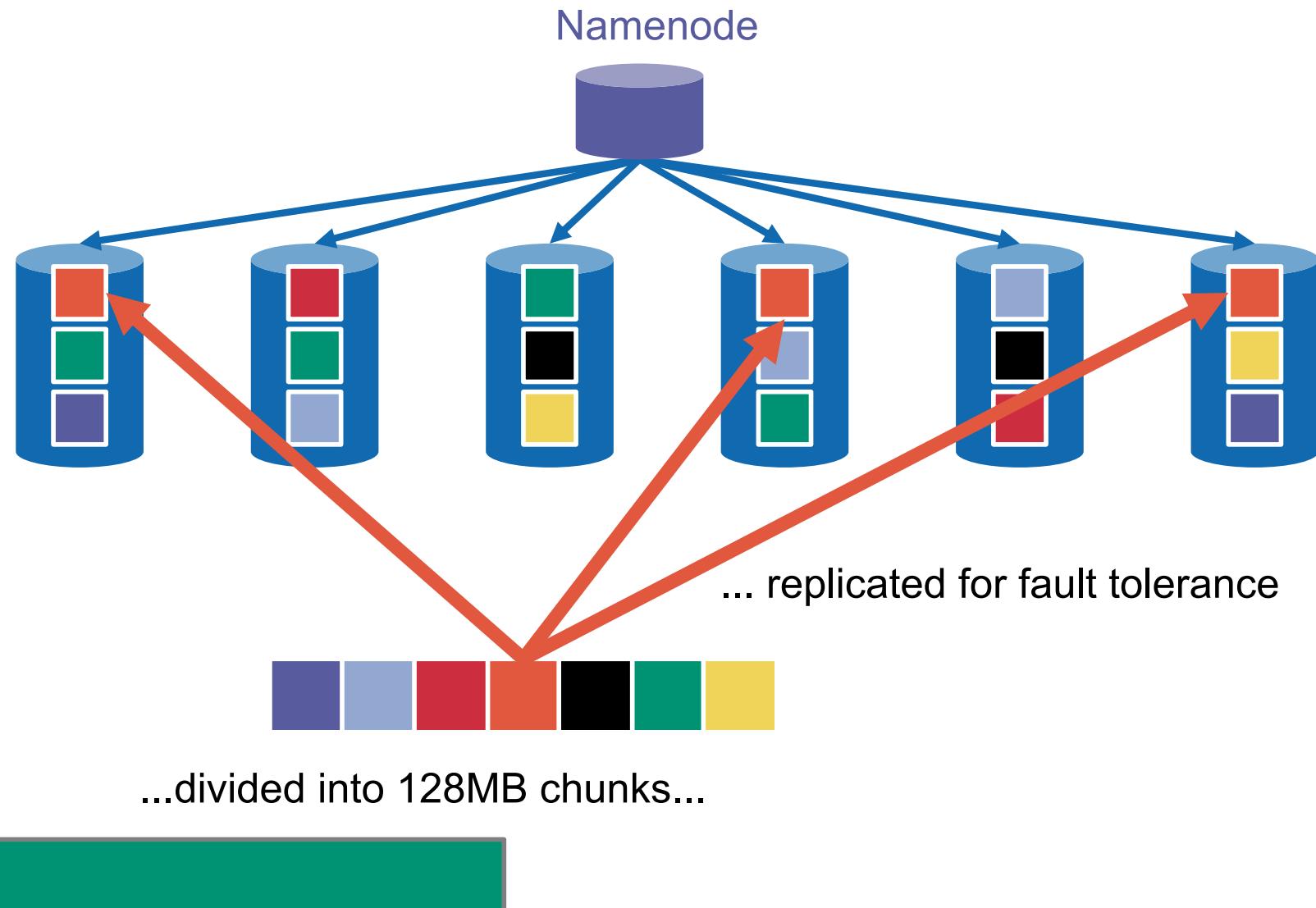
From the file perspective



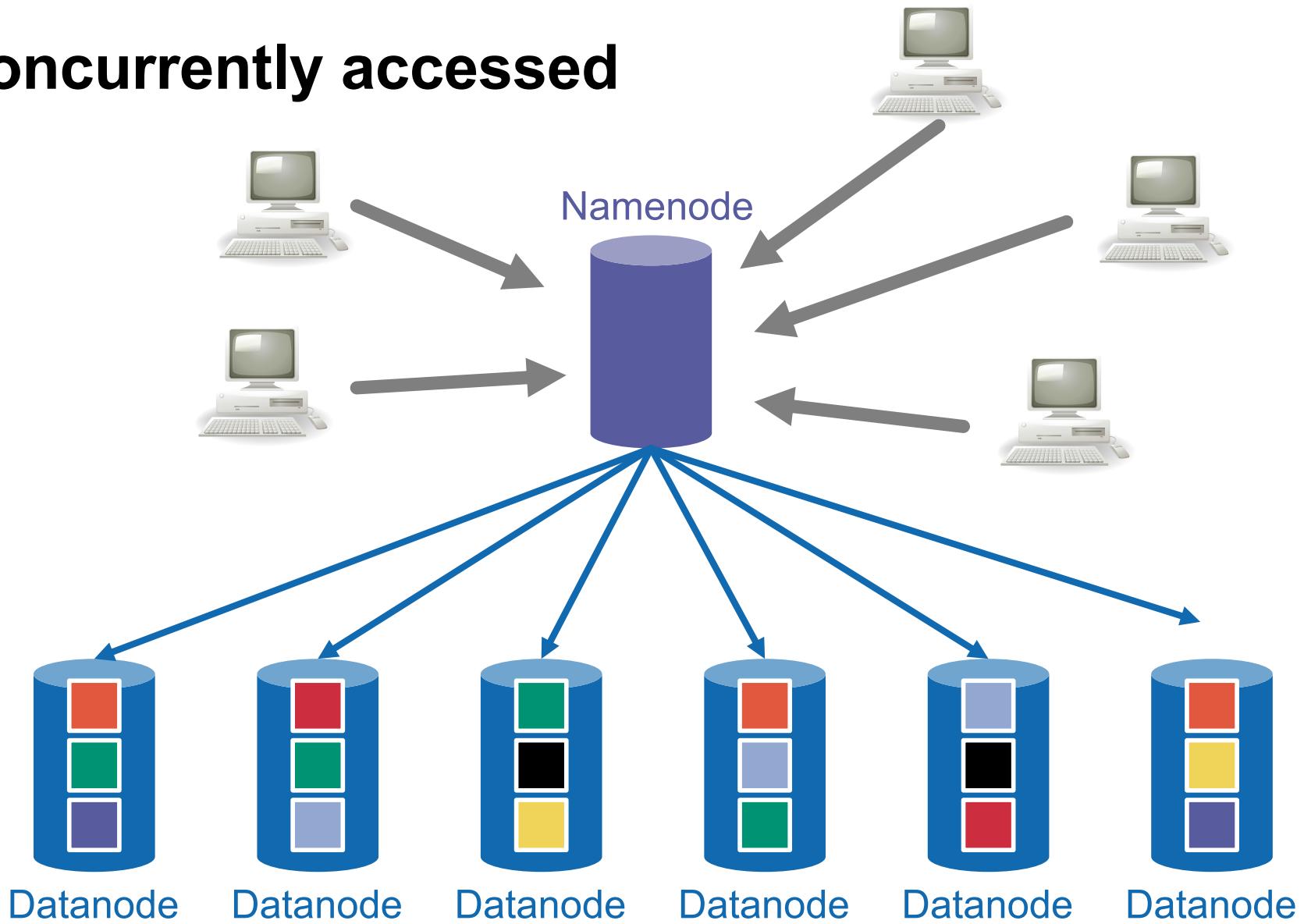
...divided into 128MB chunks...

File...

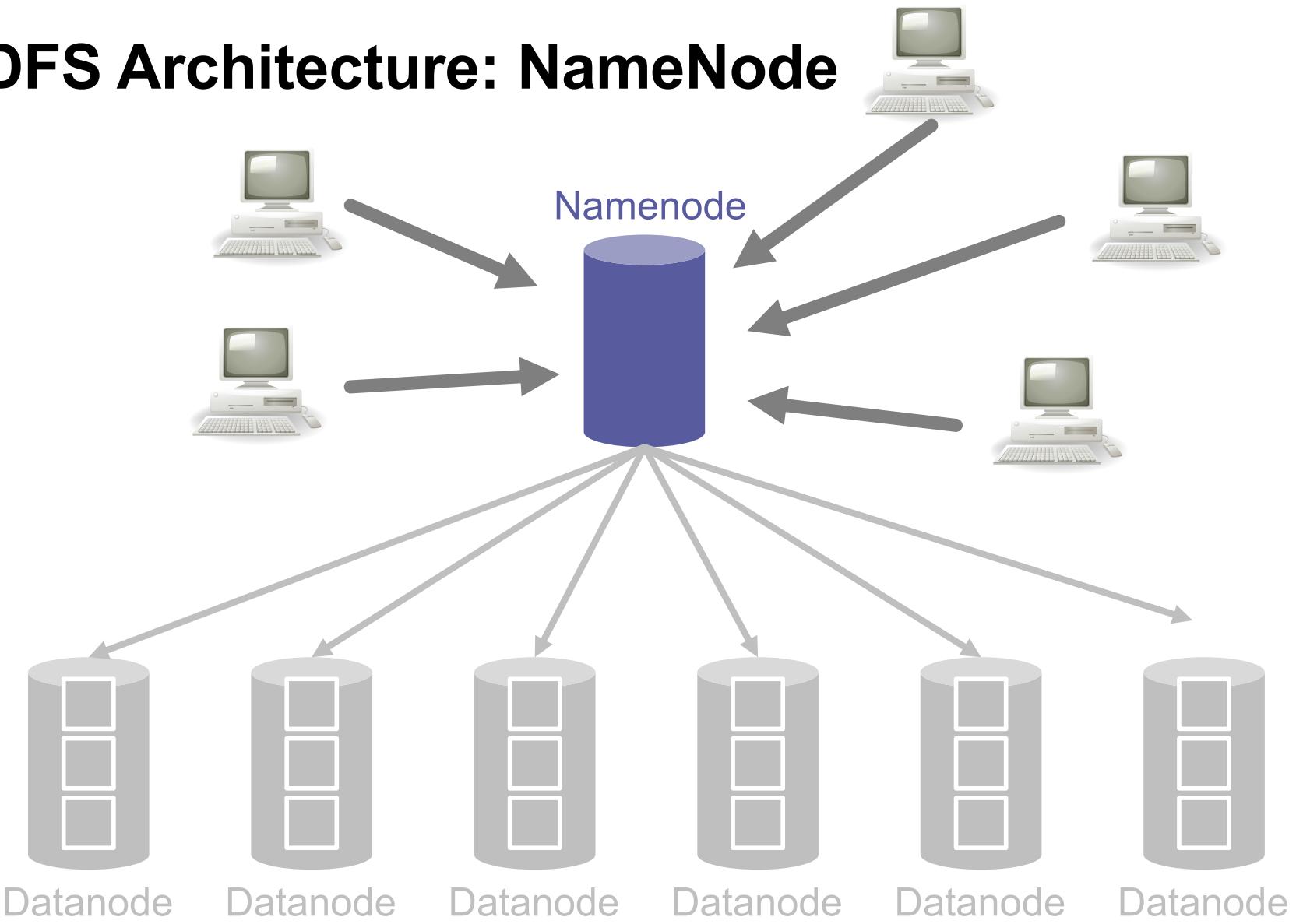
From the file perspective



Concurrently accessed



HDFS Architecture: NameNode

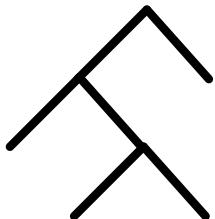


NameNode: all system-wide activity

NameNode: all system-wide activity

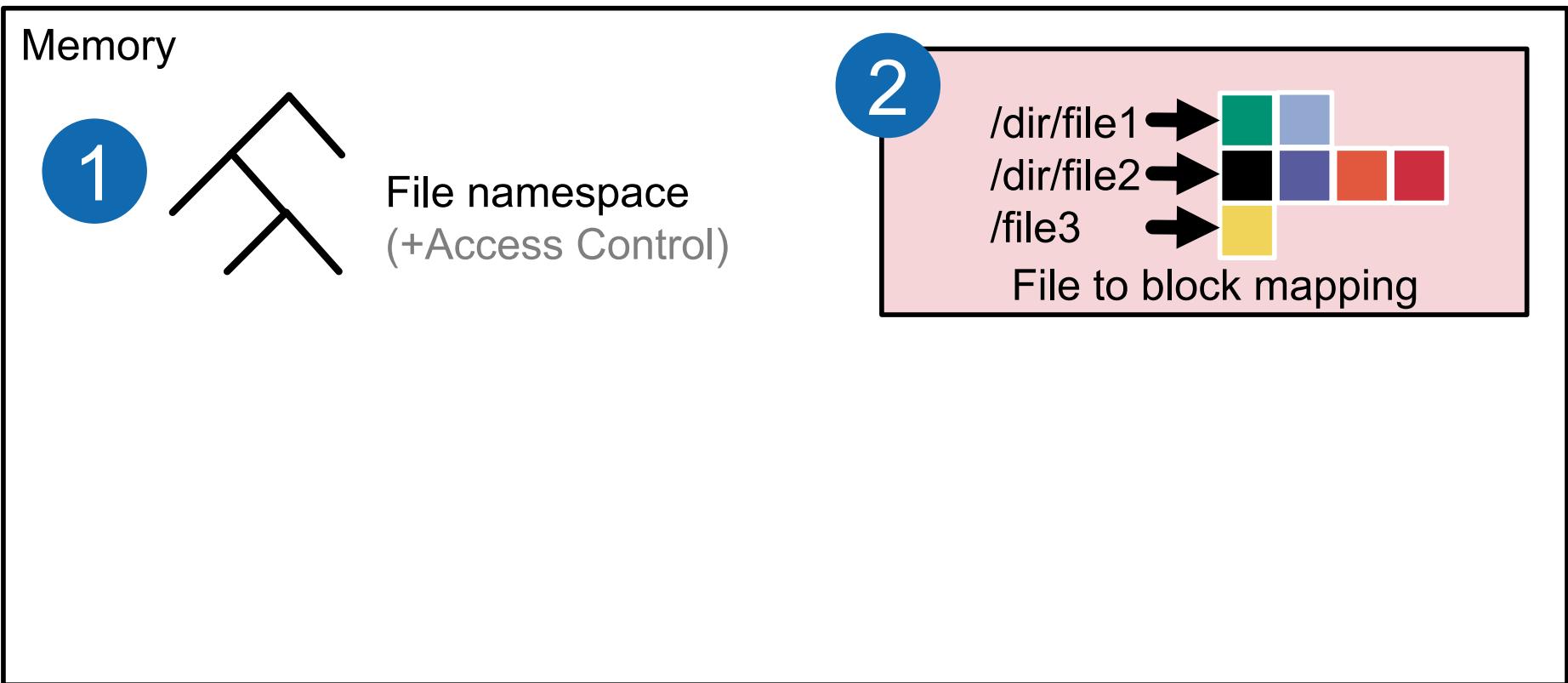
Memory

1

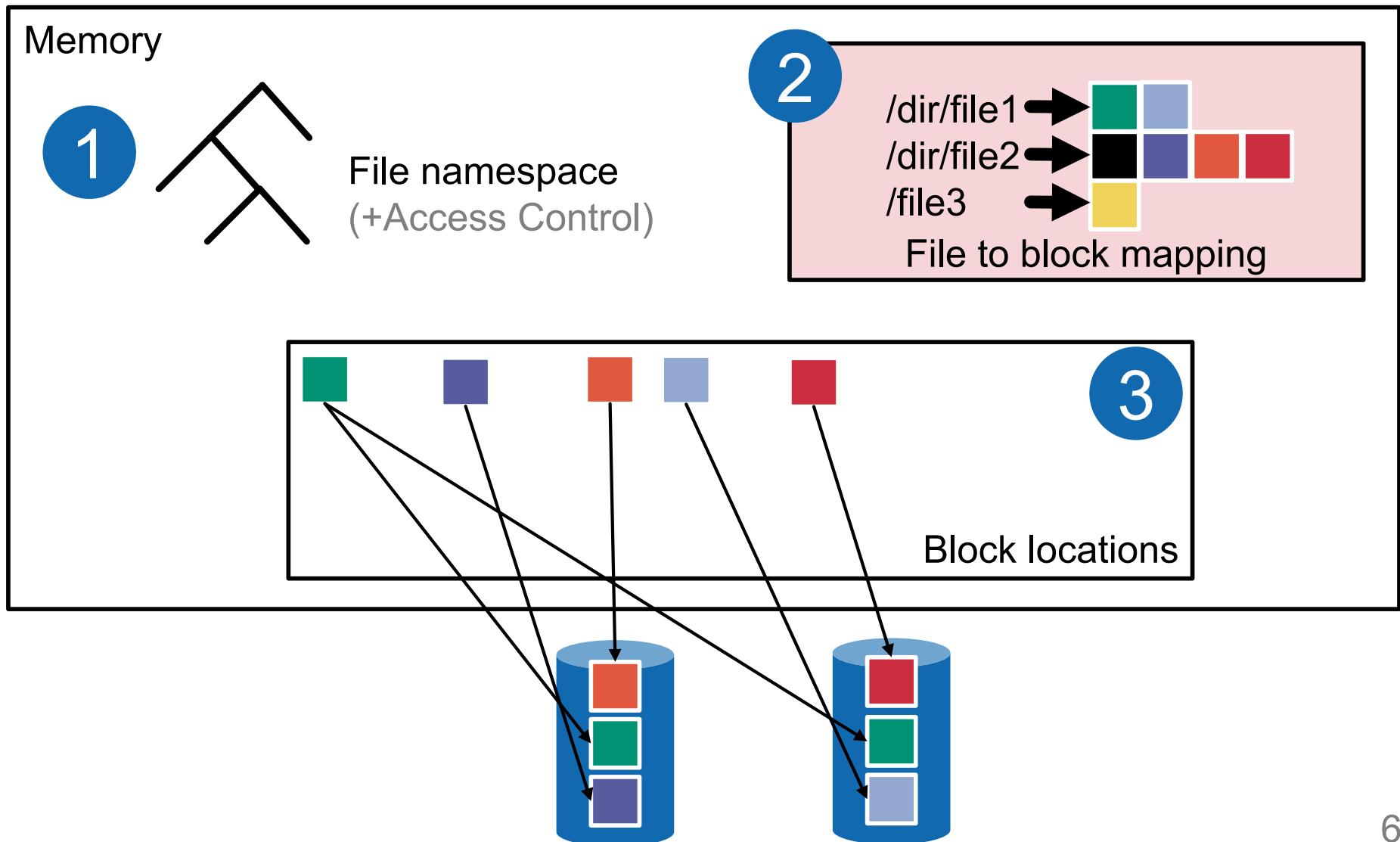


File namespace
(+Access Control)

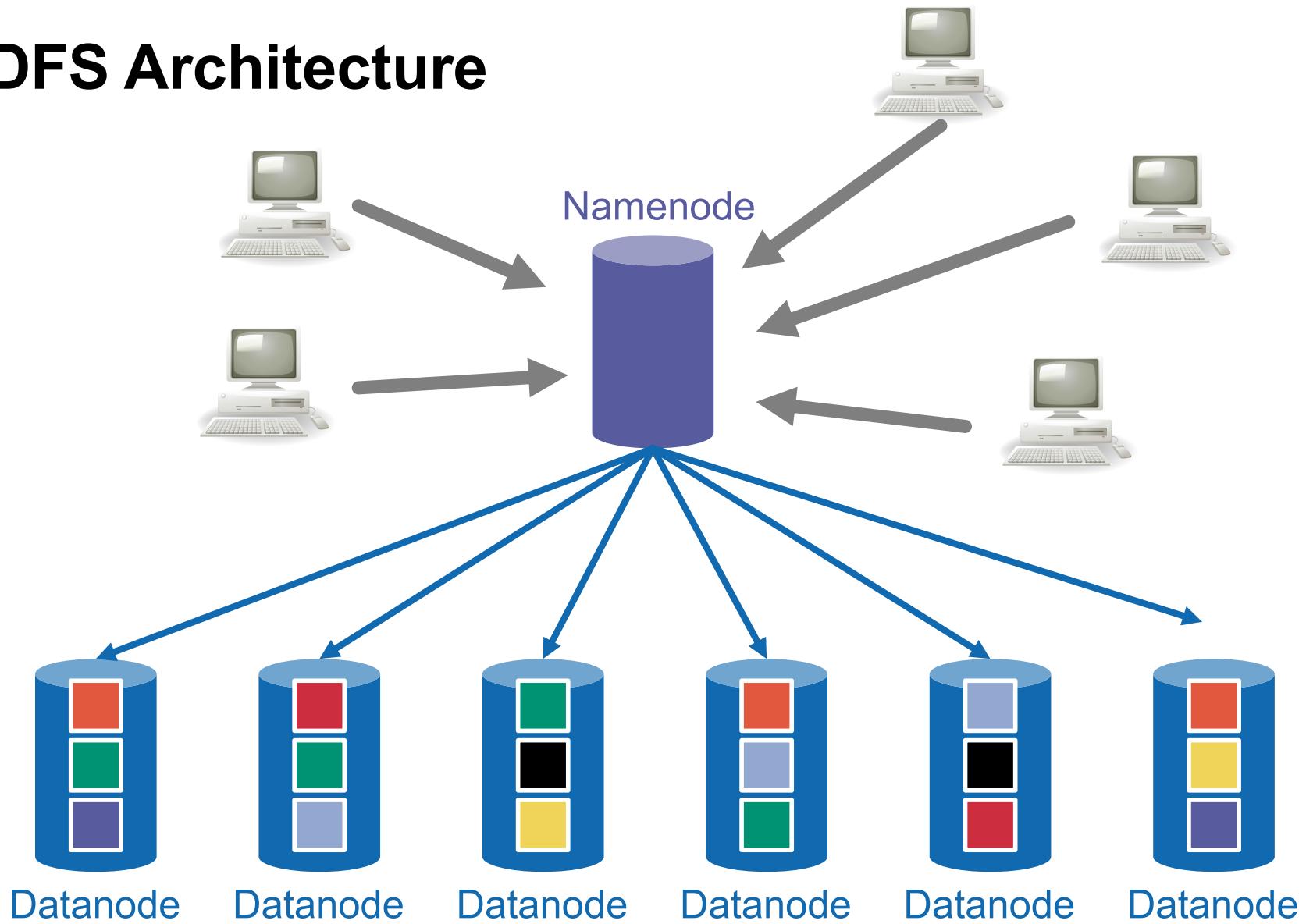
NameNode: all system-wide activity



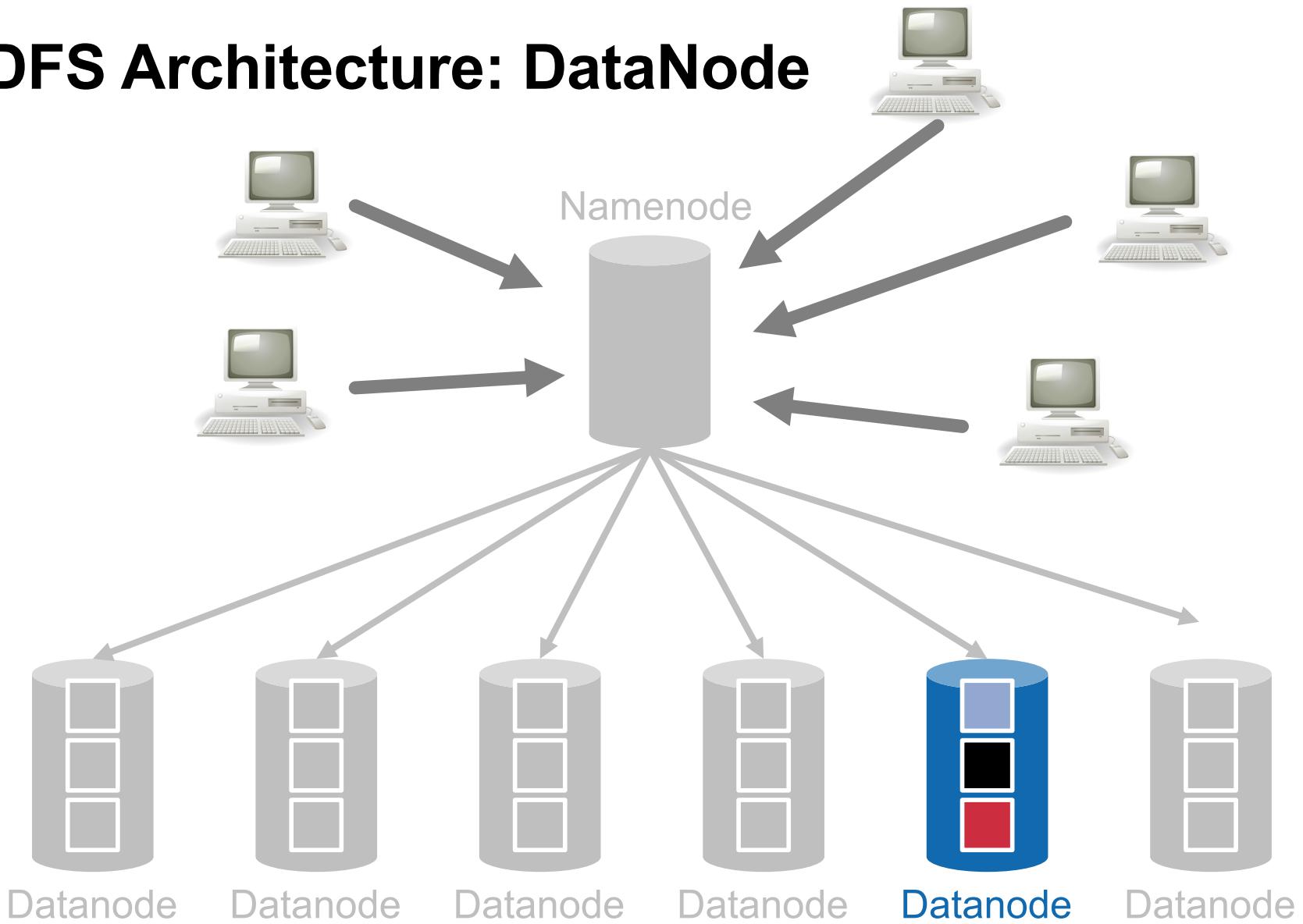
NameNode: all system-wide activity



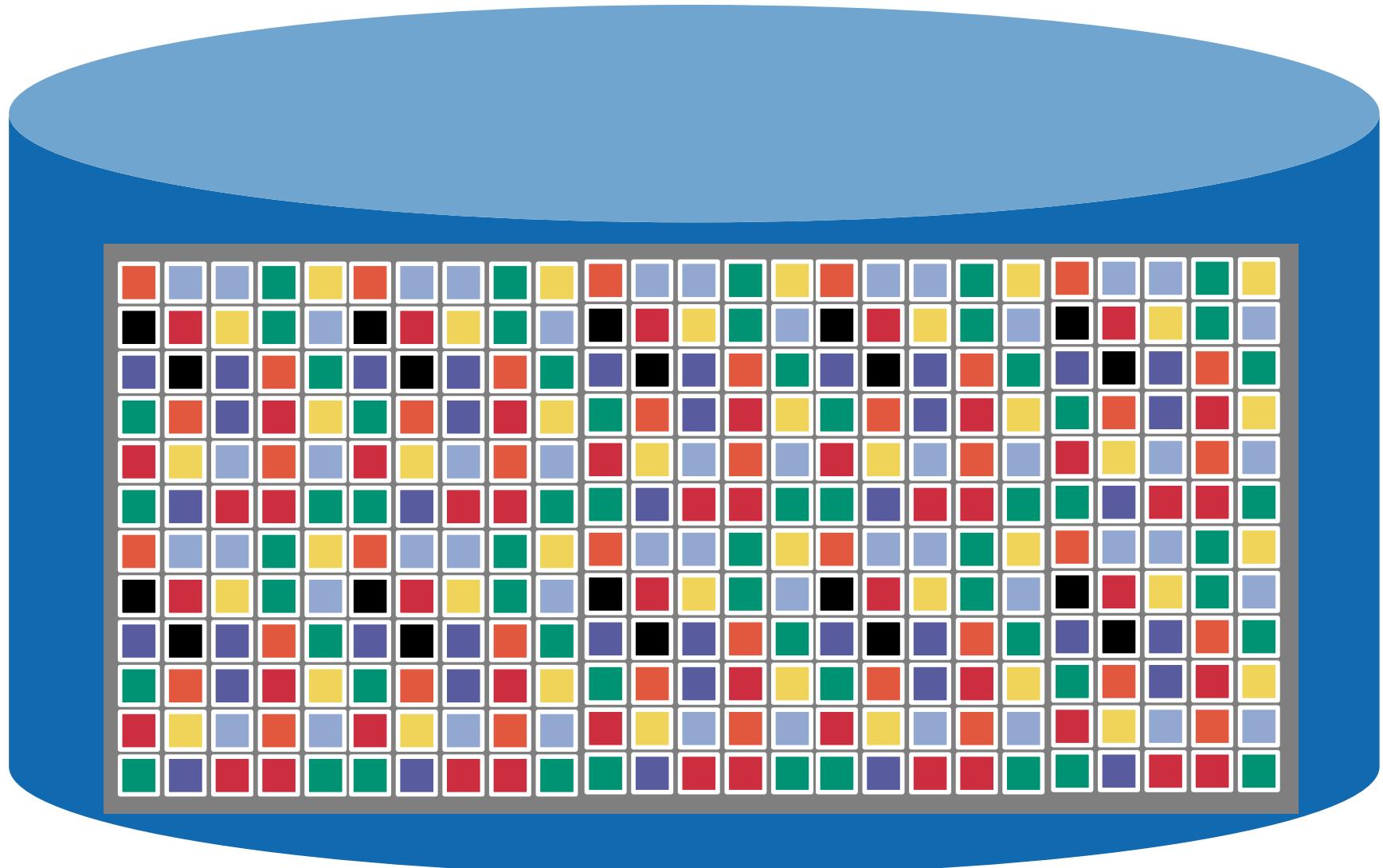
HDFS Architecture



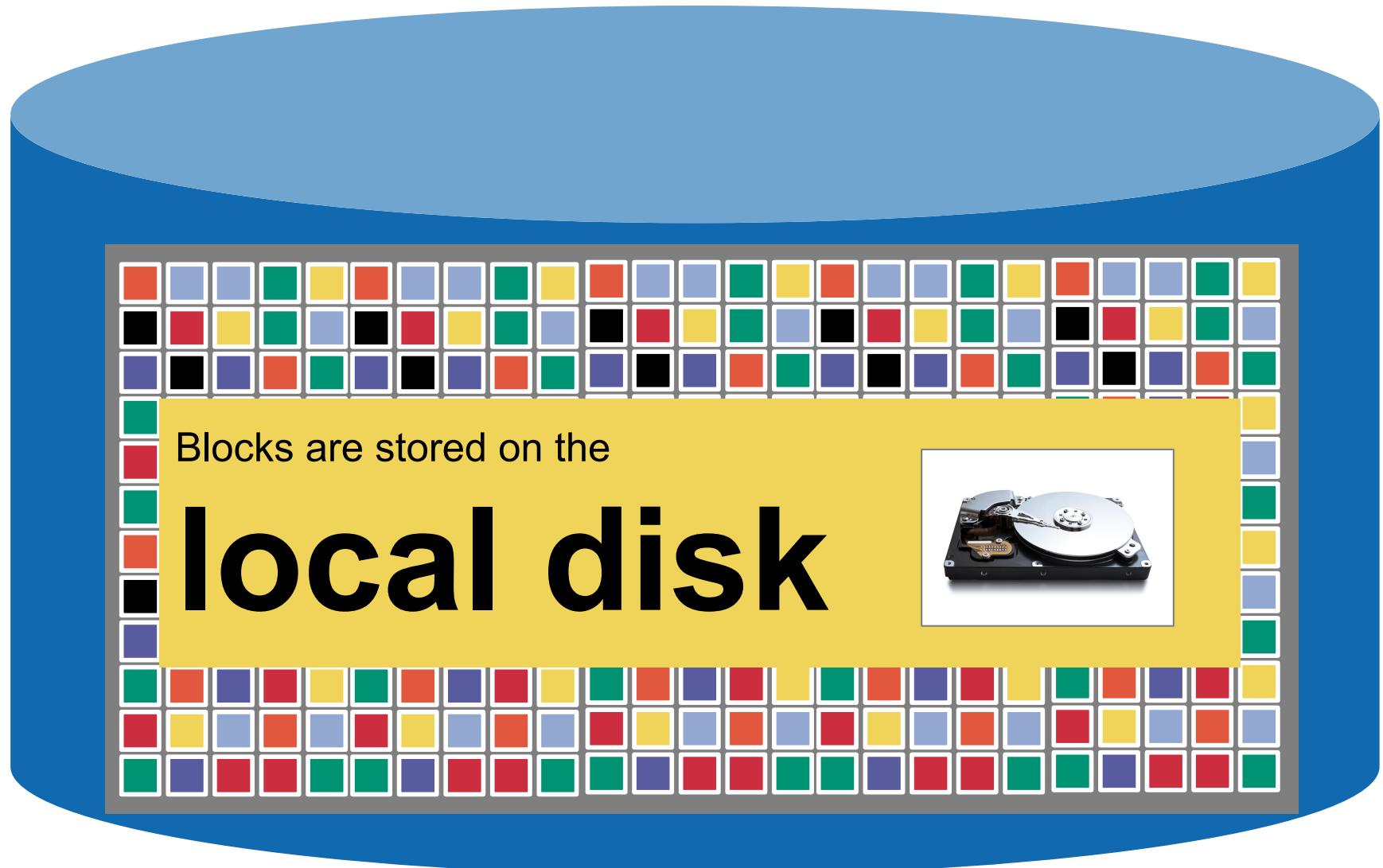
HDFS Architecture: DataNode



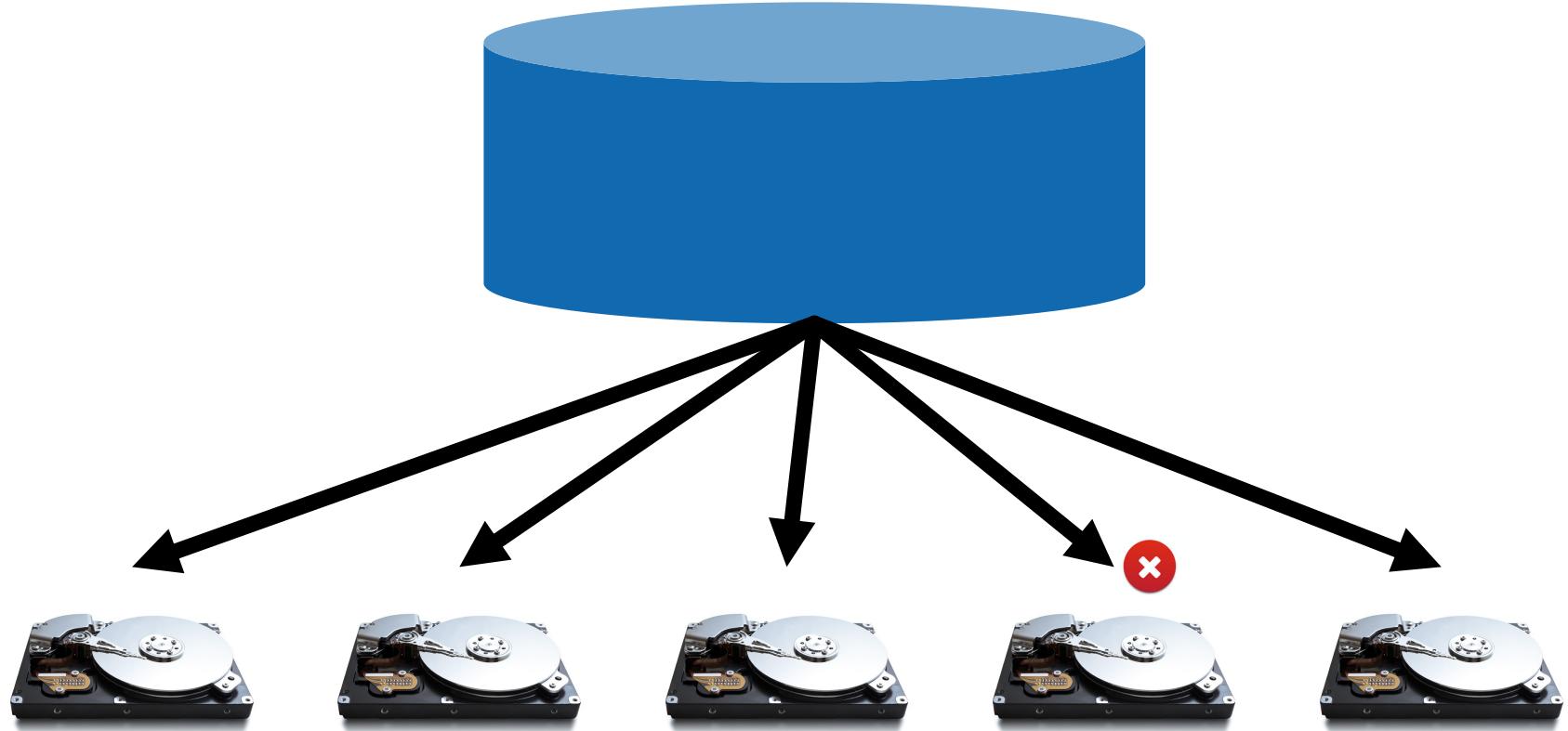
DataNode



DataNode



DataNode

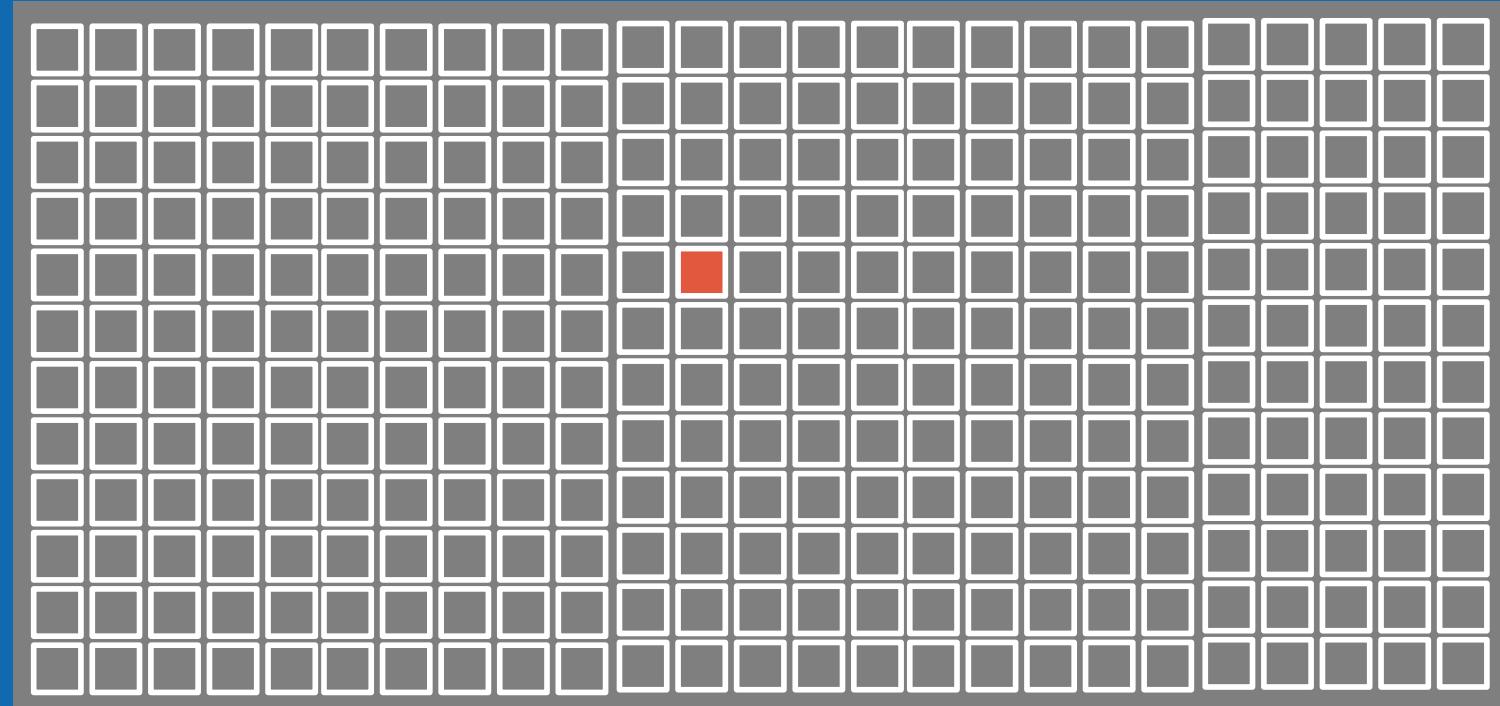


Proximity to hardware facilitates **disk failure detection**

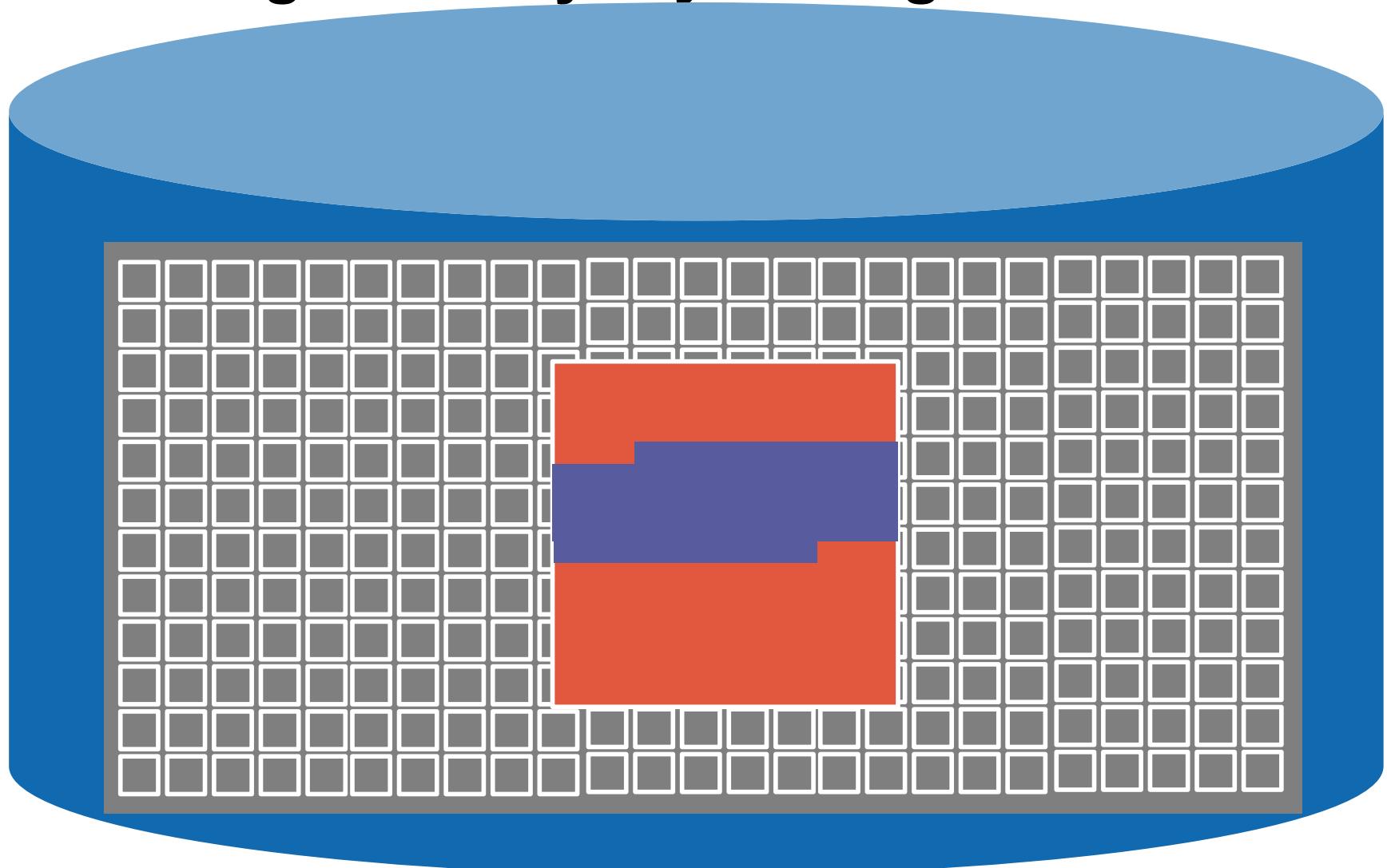
Block IDs

64 bits

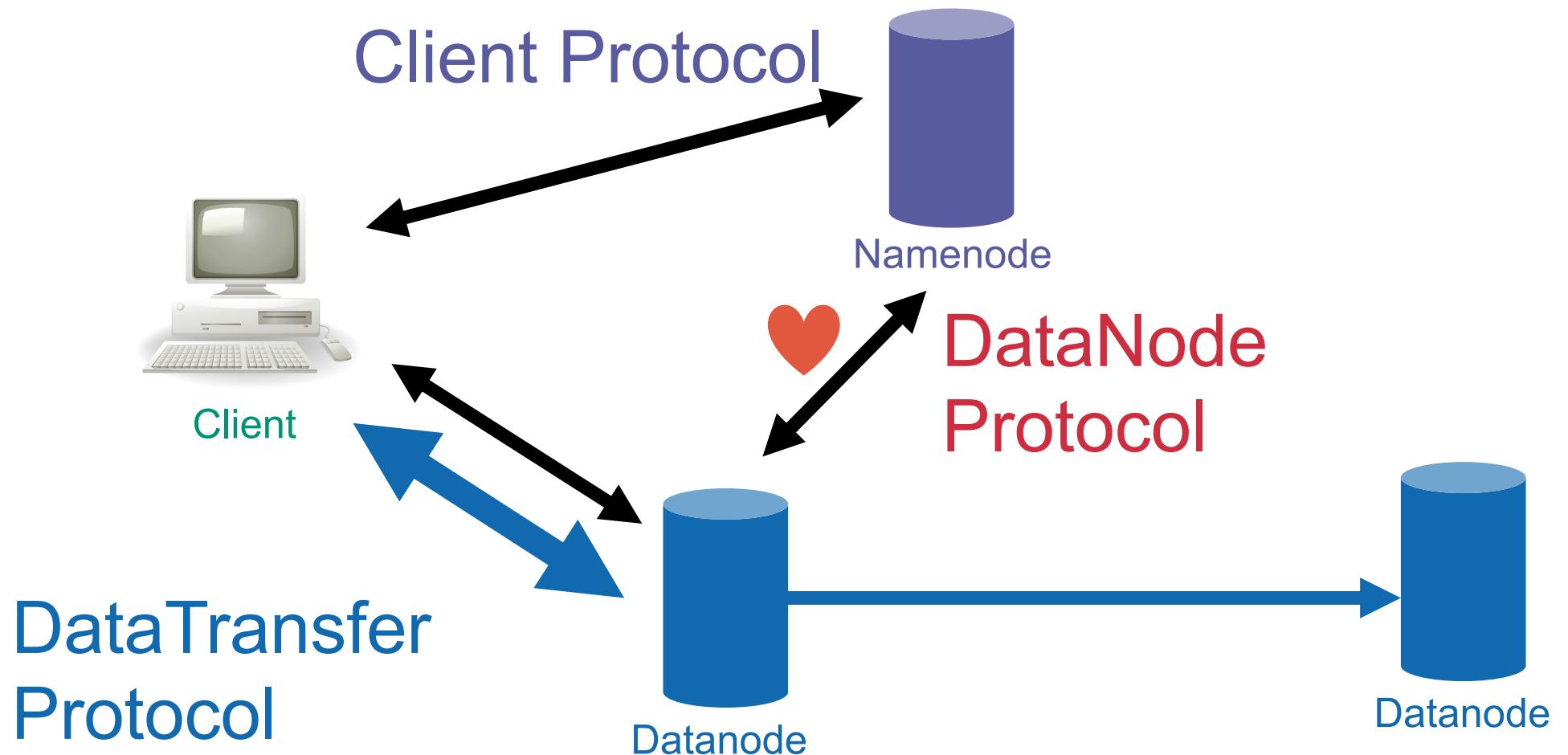
e.g., 7586700455251598184



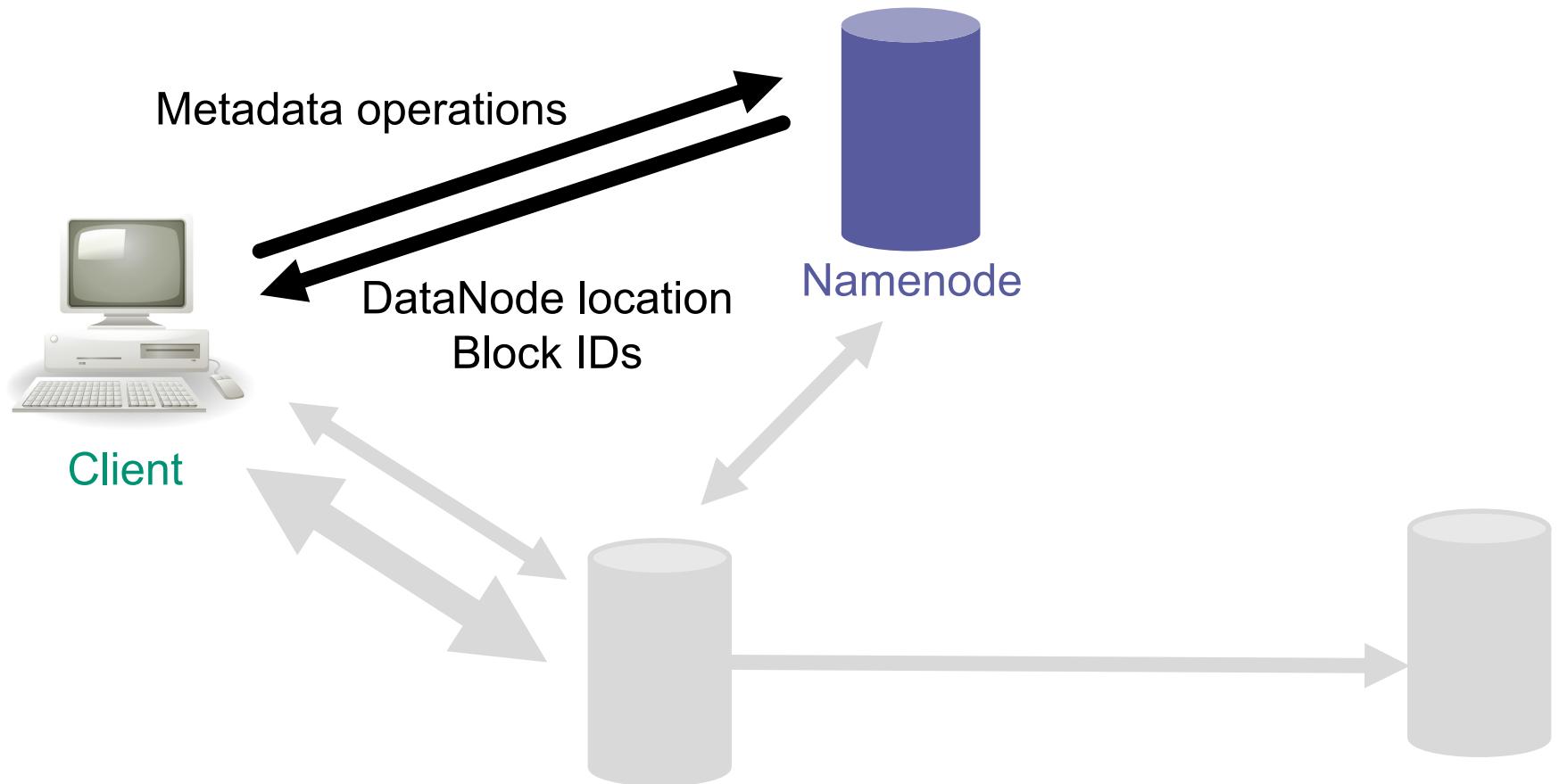
Subblock granularity: Byte Range



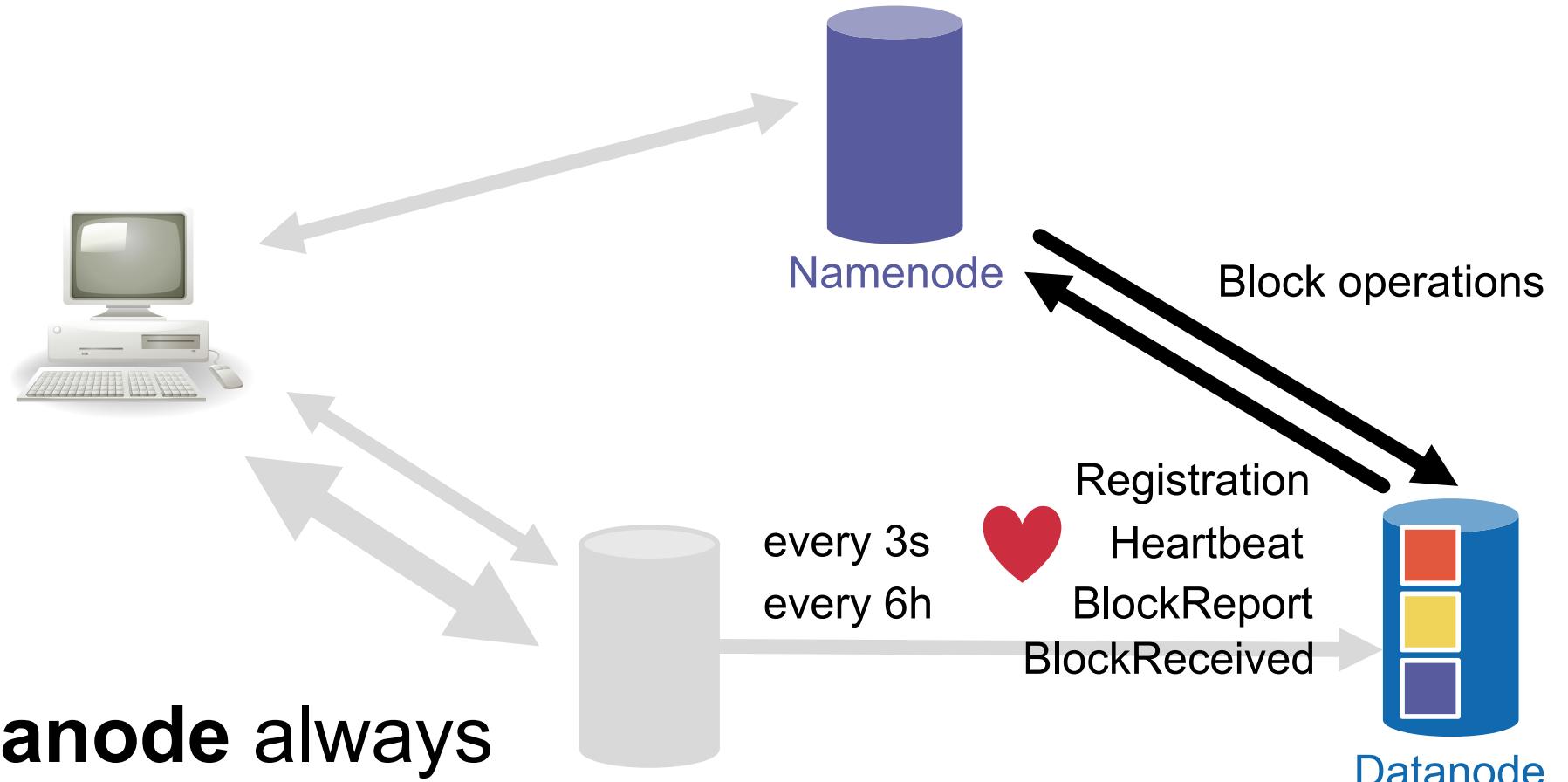
Communication



Client Protocol

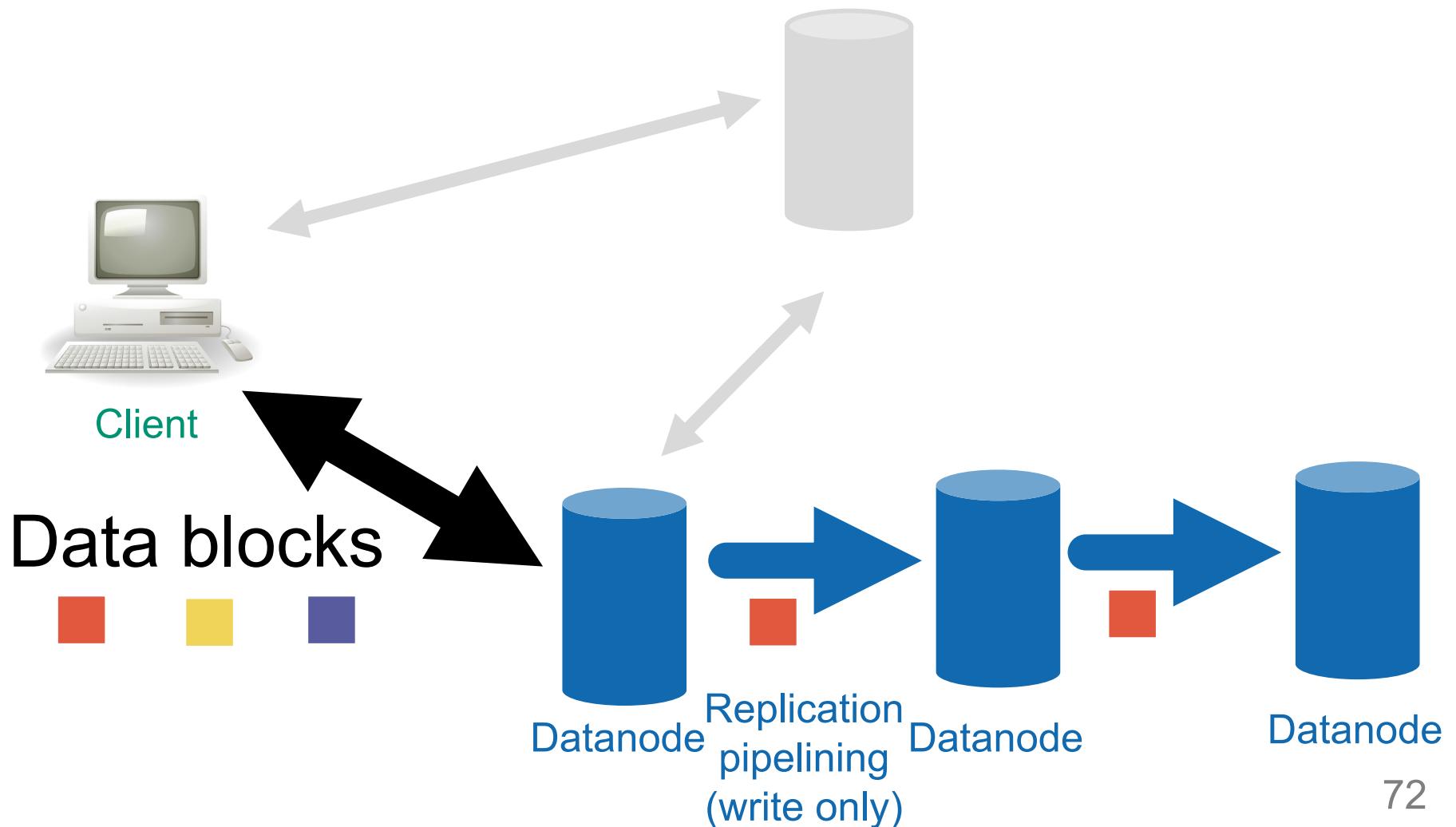


DataNode Protocol



Datanode always initiates connection!

Data Transfer Protocol



Metadata functionality

Create directory

Delete directory



Write file

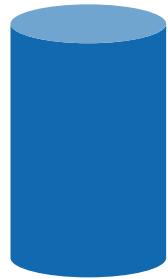
Append to file



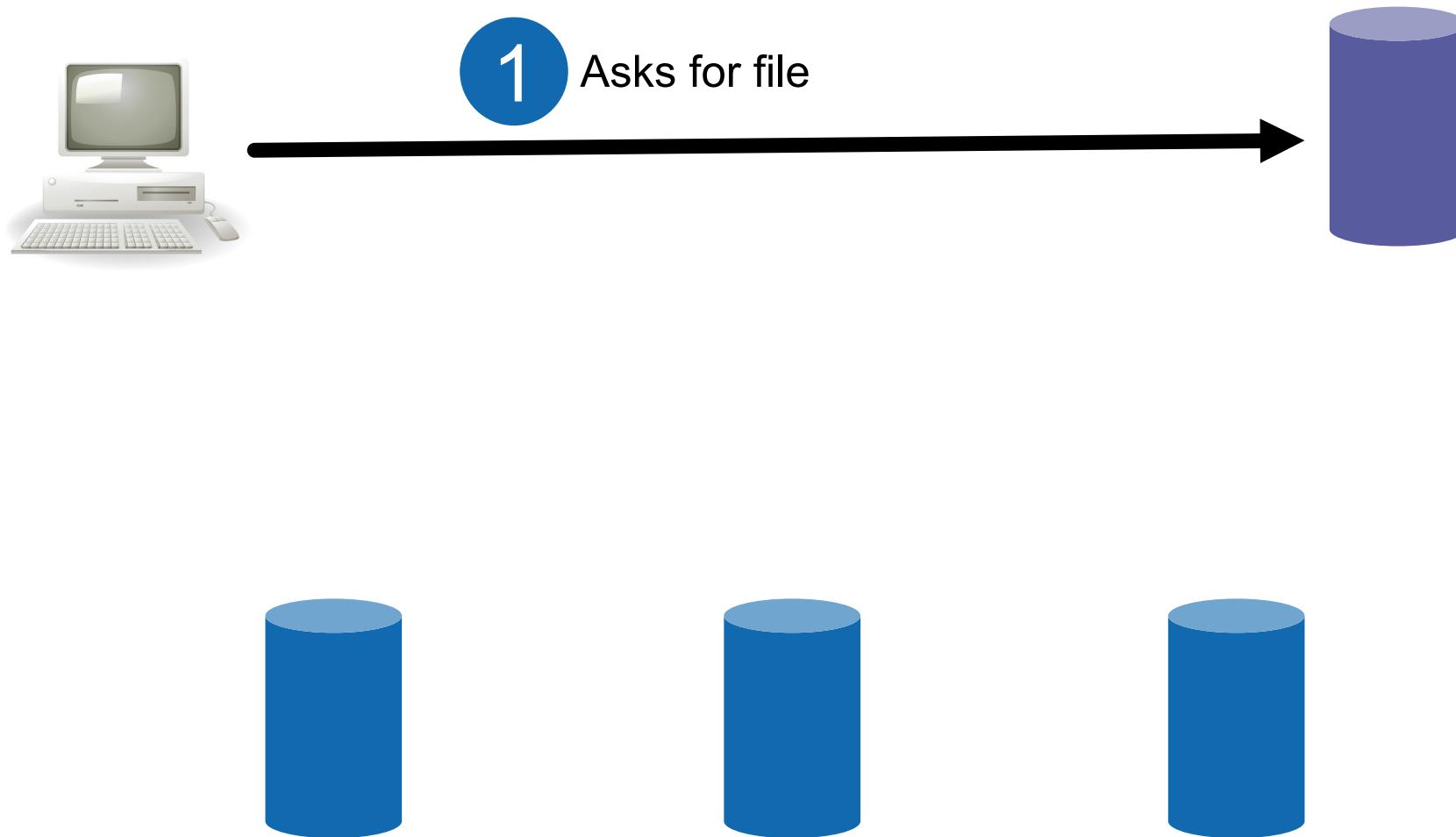
Read file

Delete file

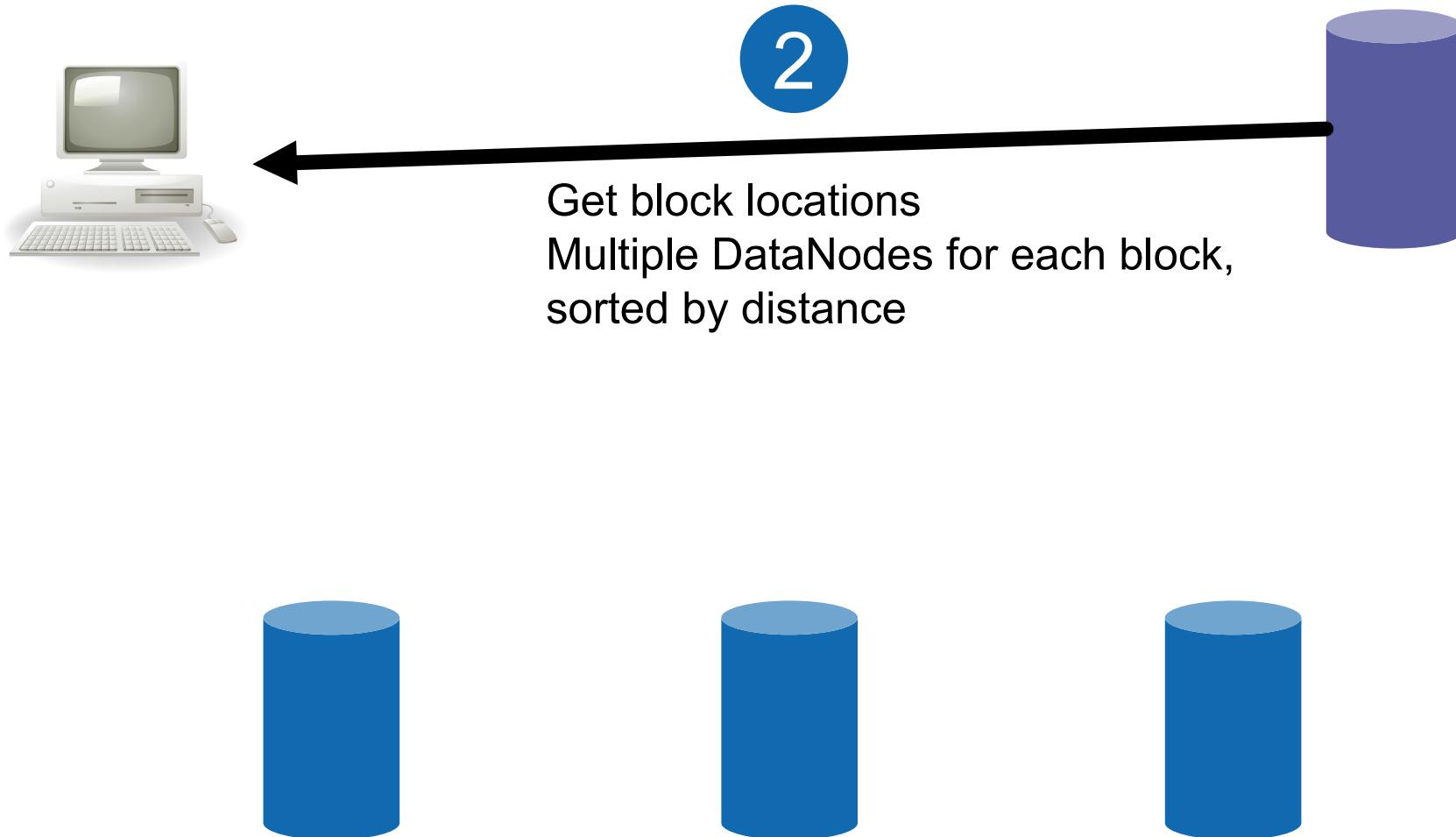
Client reads a file



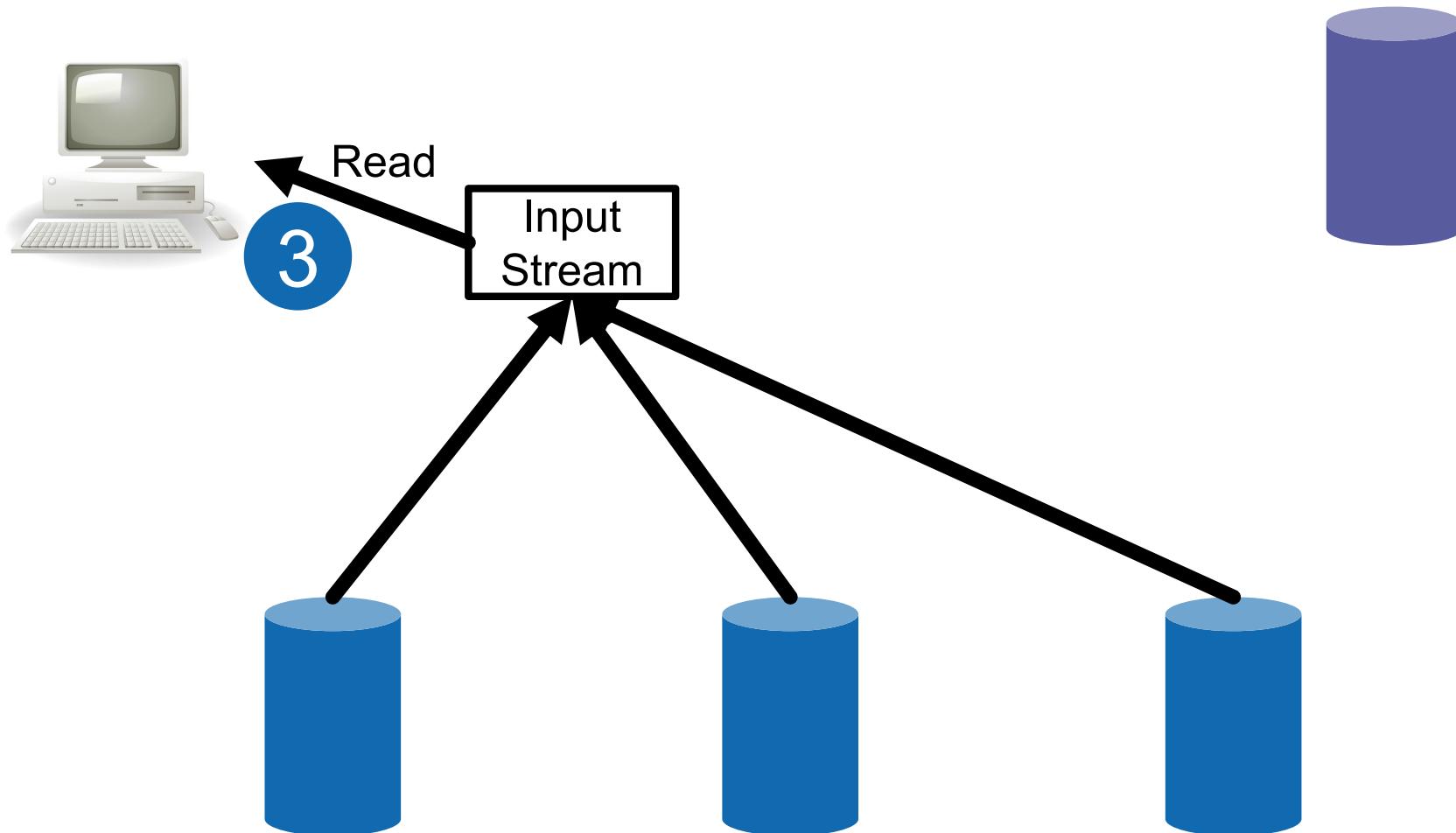
Client reads a file



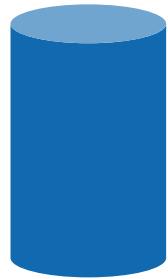
Client reads a file



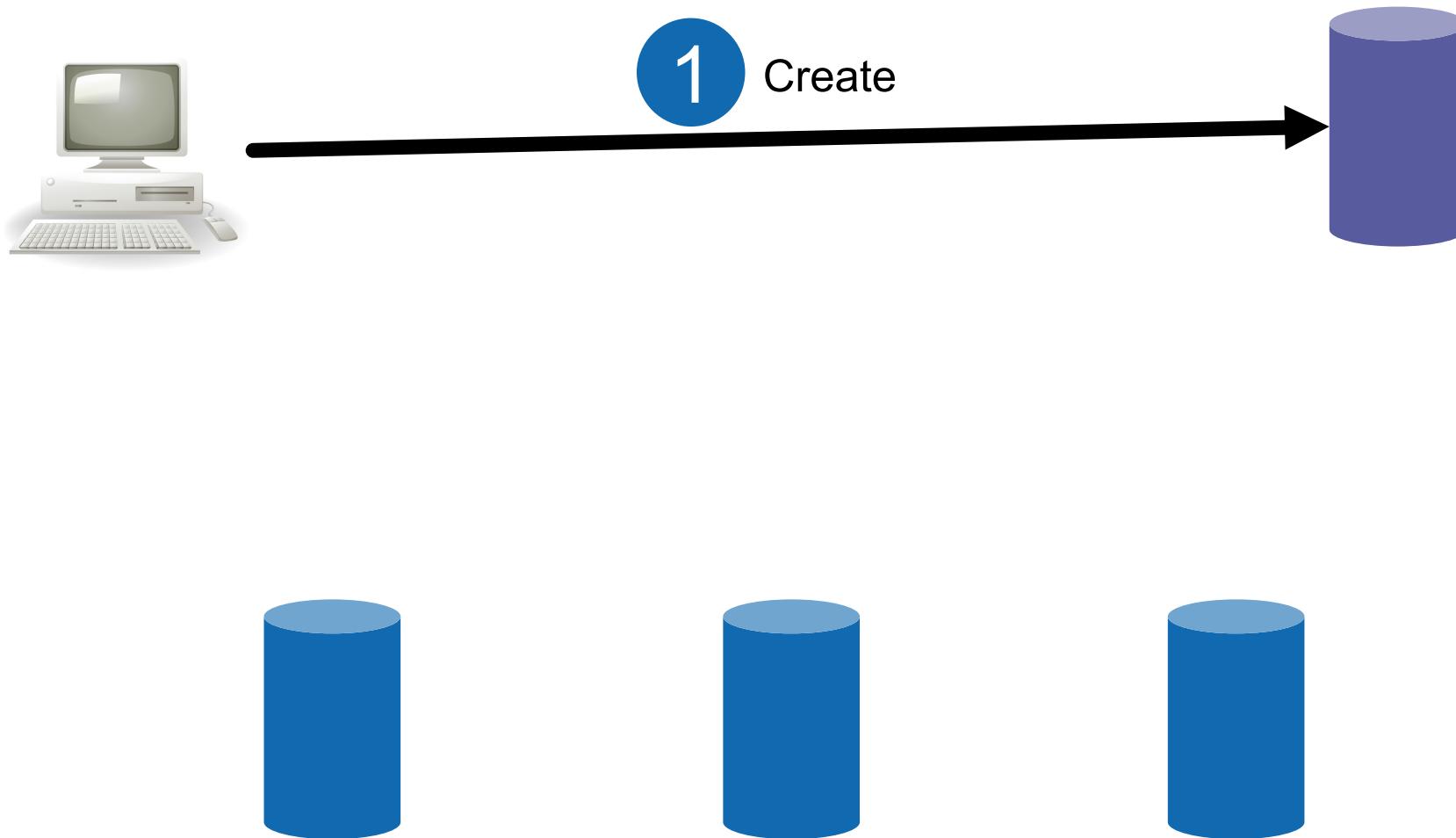
Client reads a file



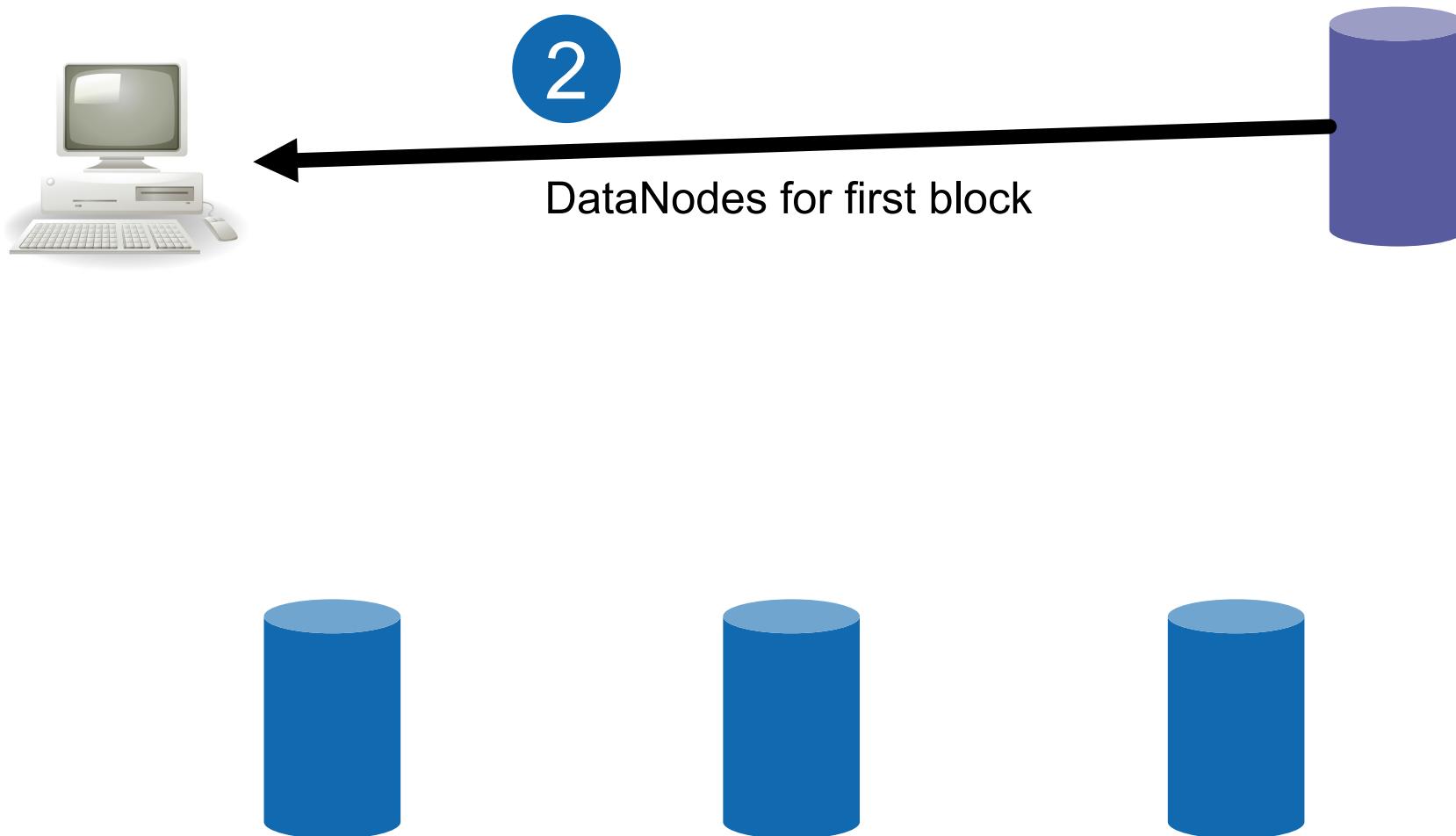
Client writes a file



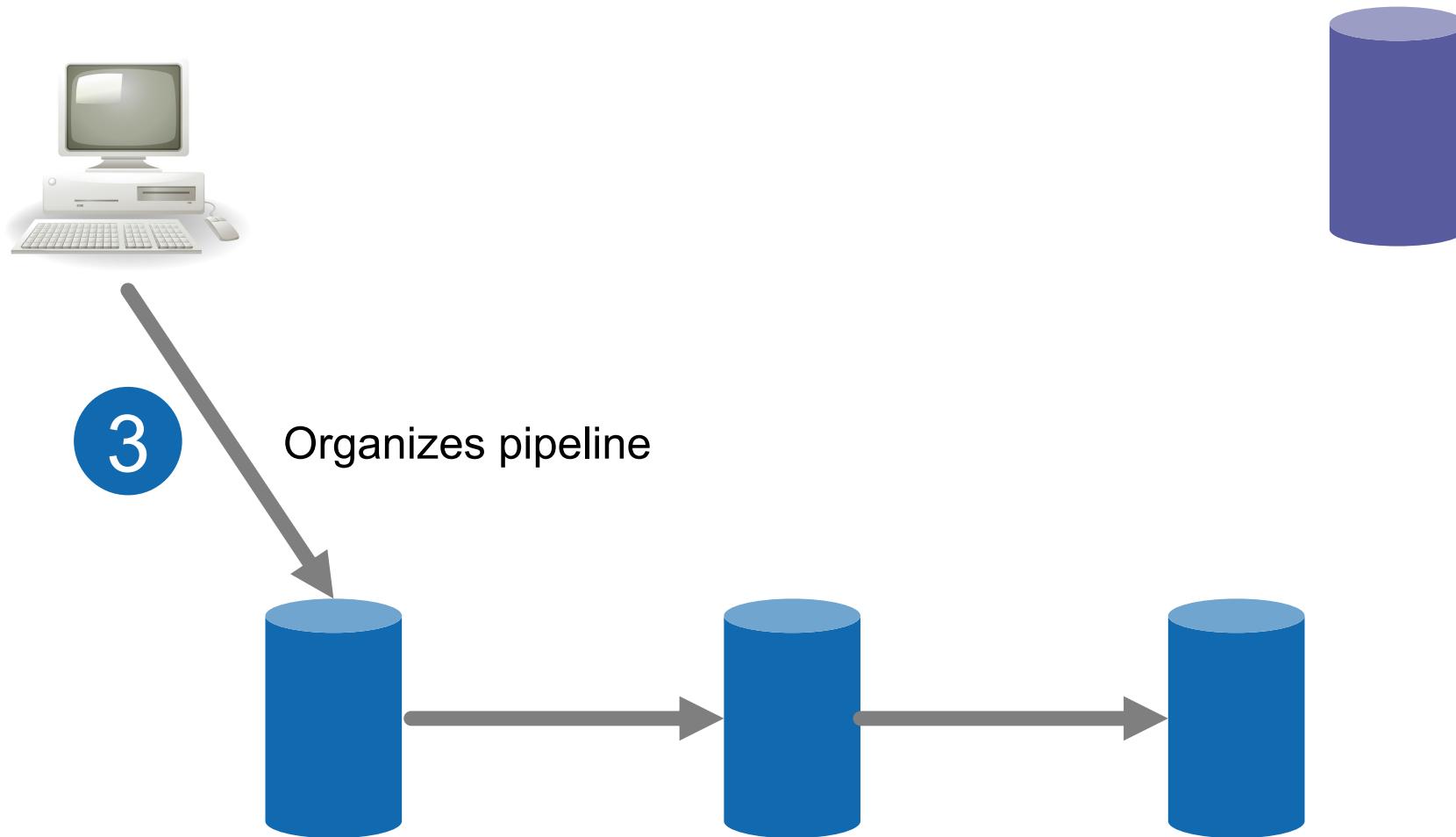
Client writes a file



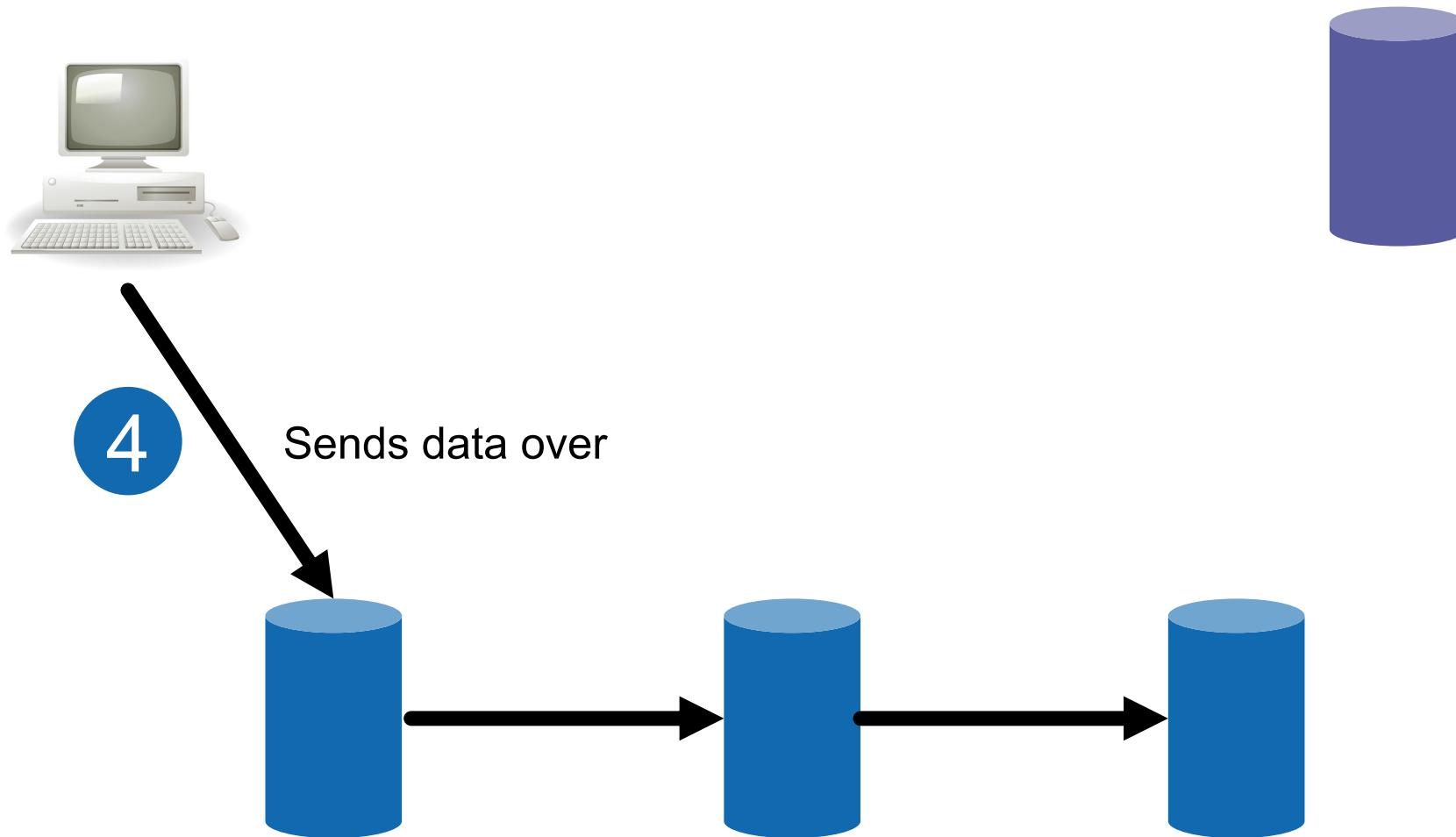
Client writes a file



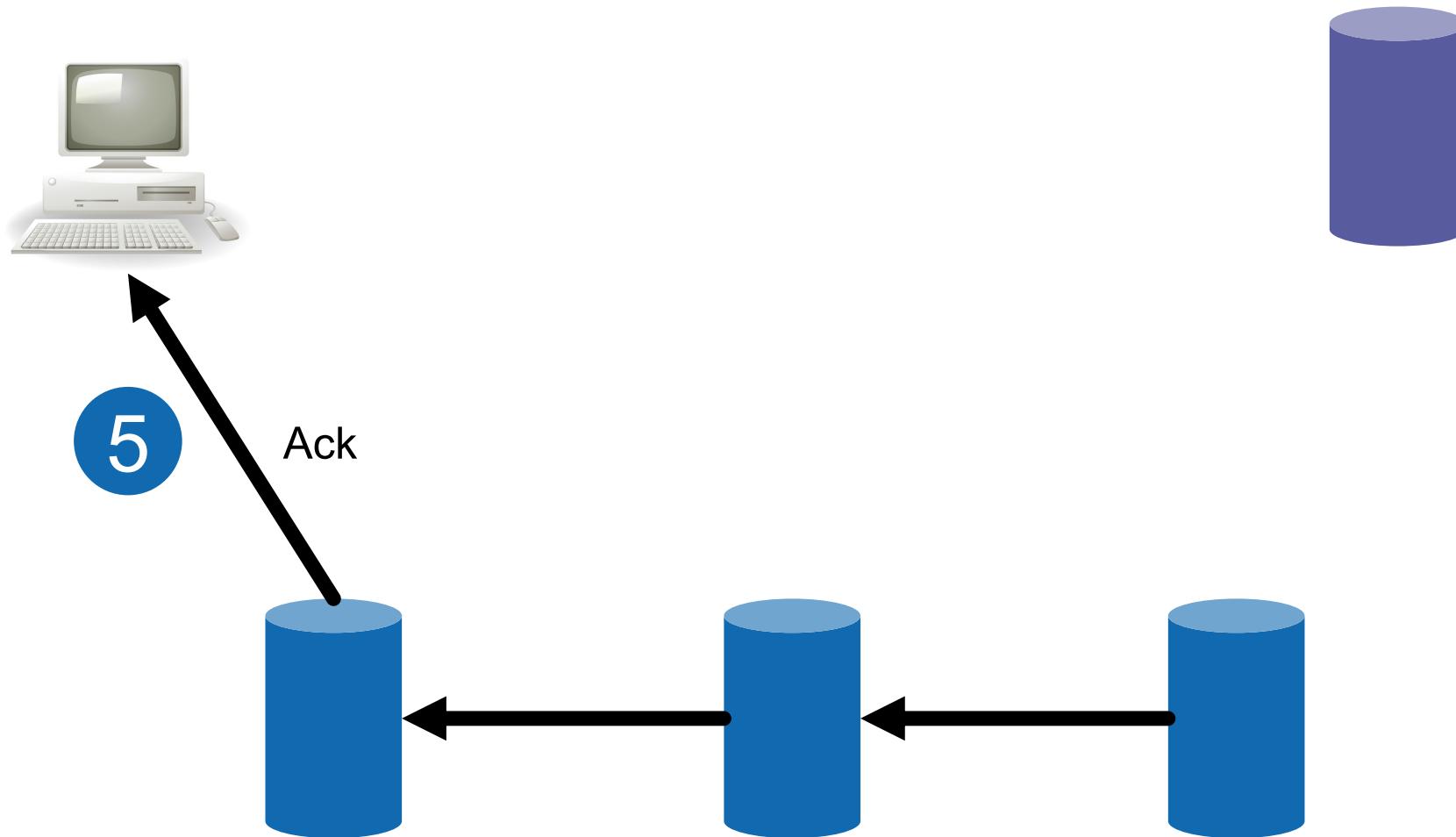
Client writes a file



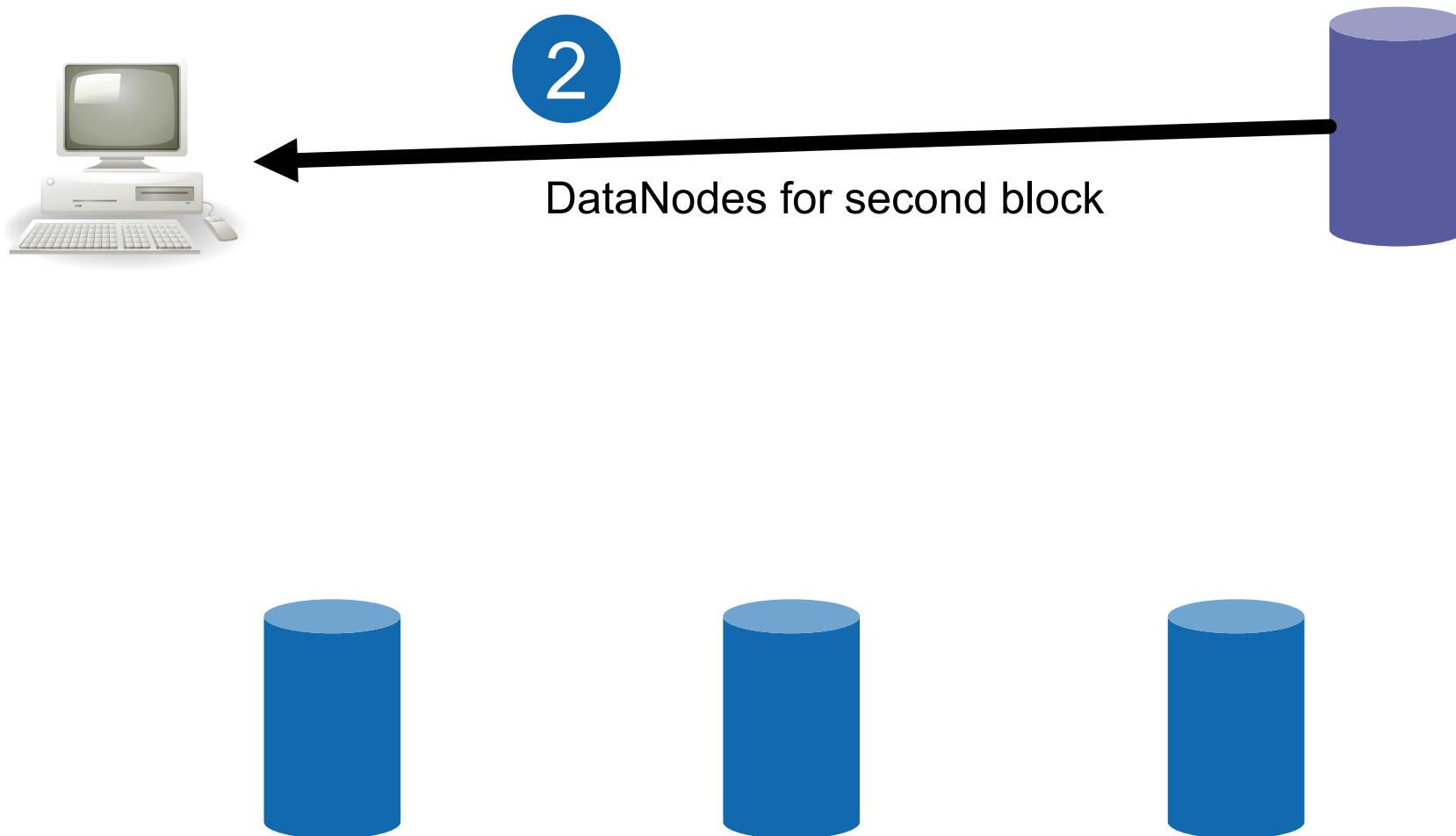
Client writes a file



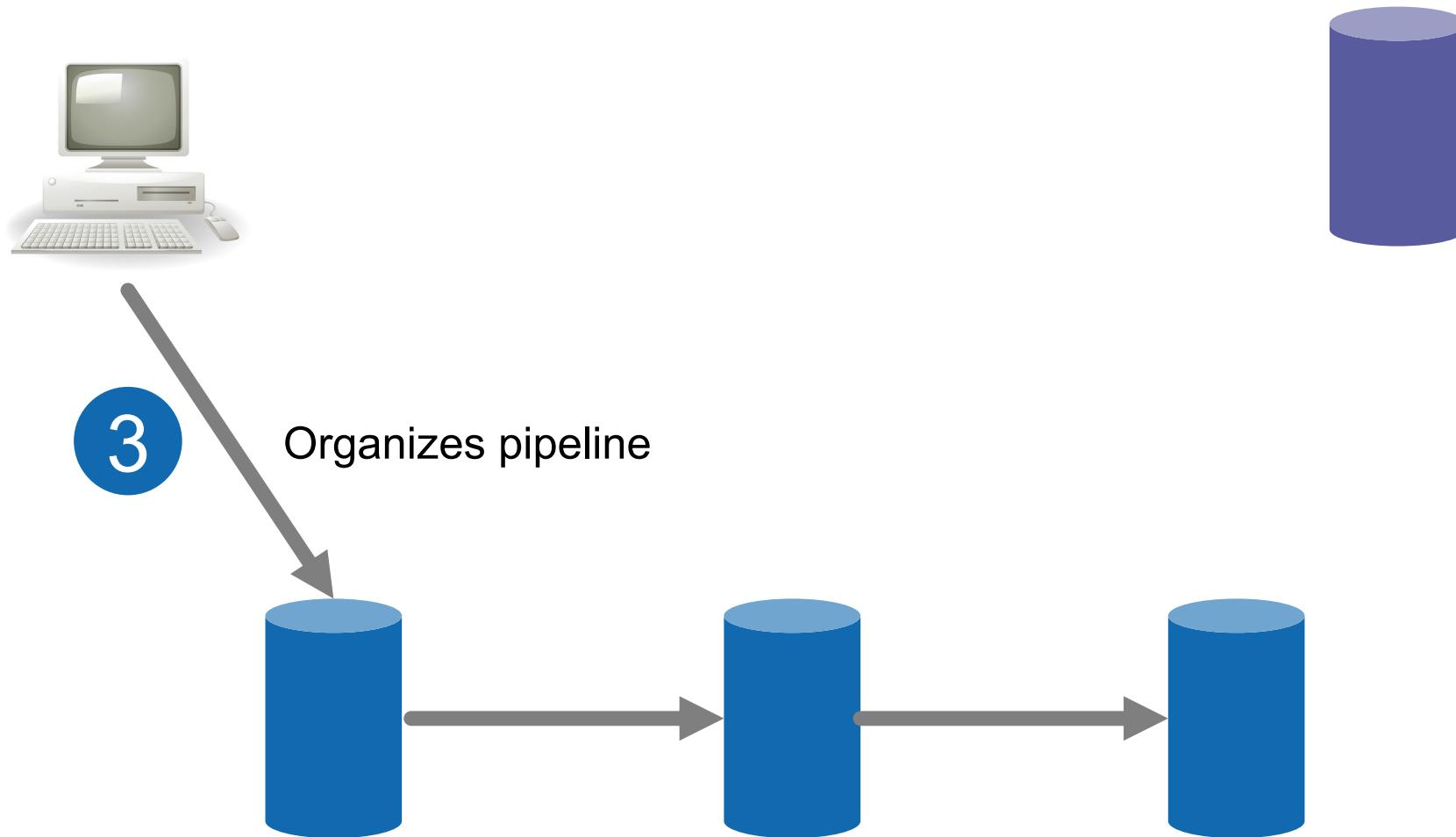
Client writes a file



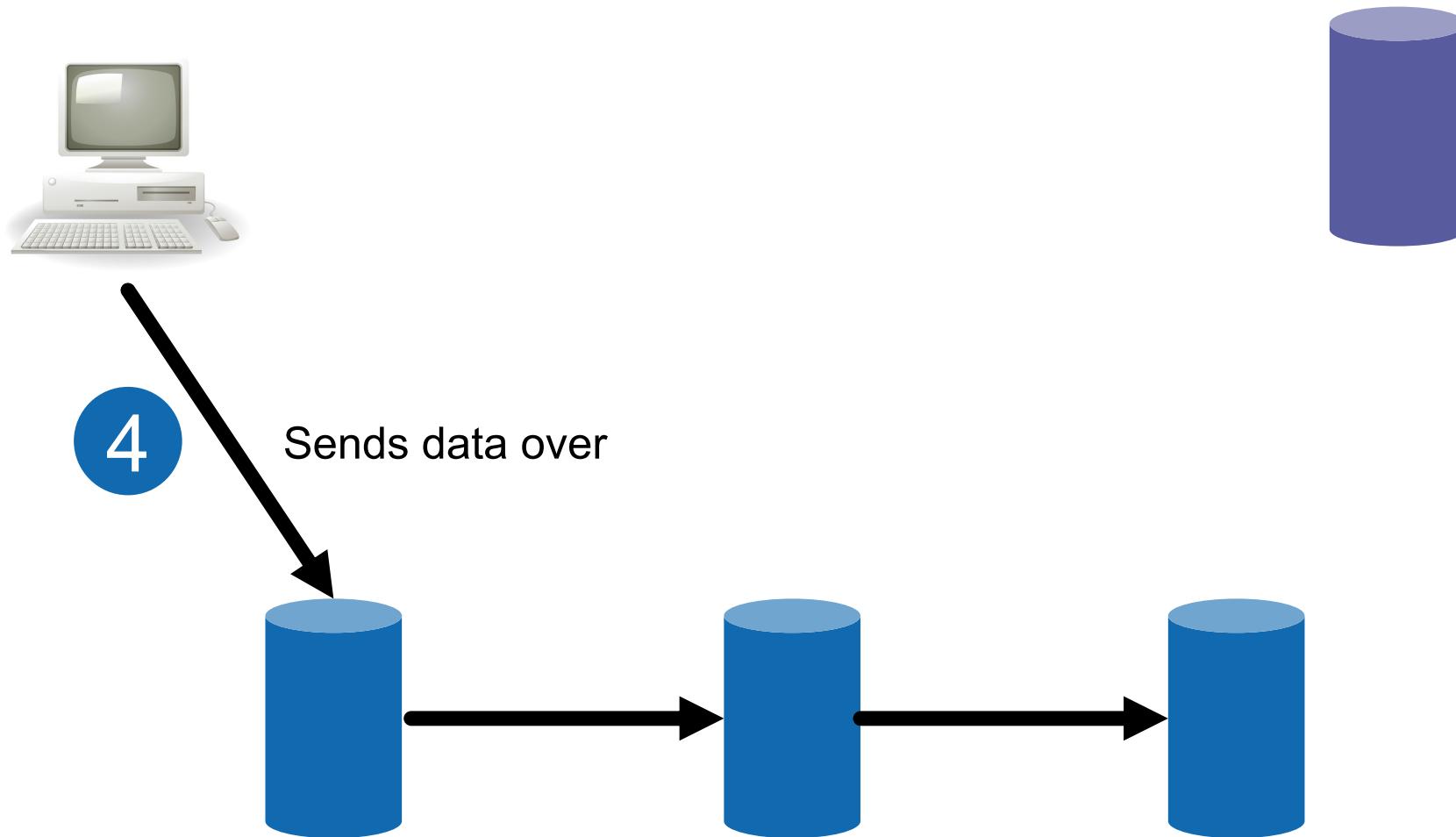
Client writes a file



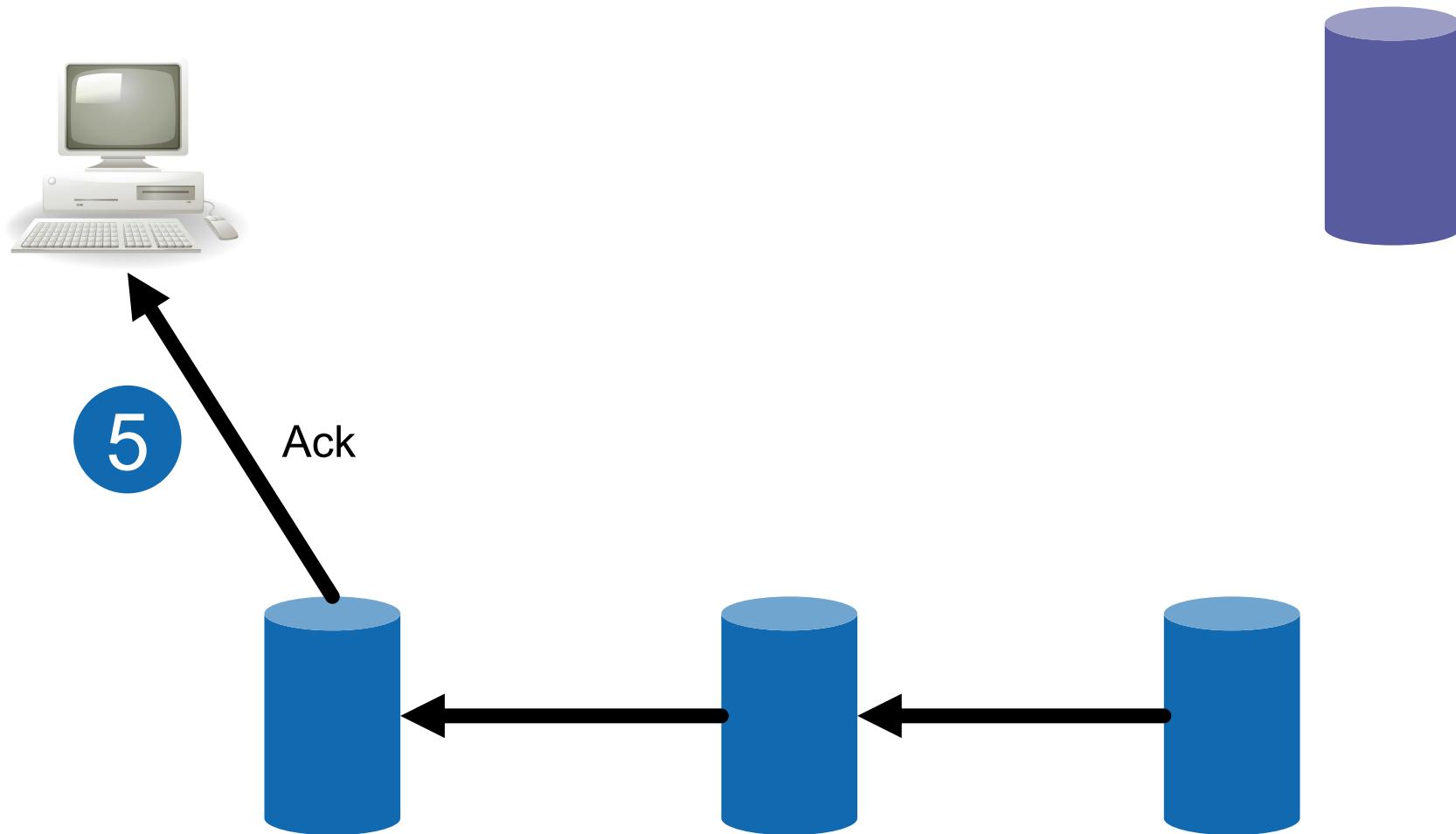
Client writes a file



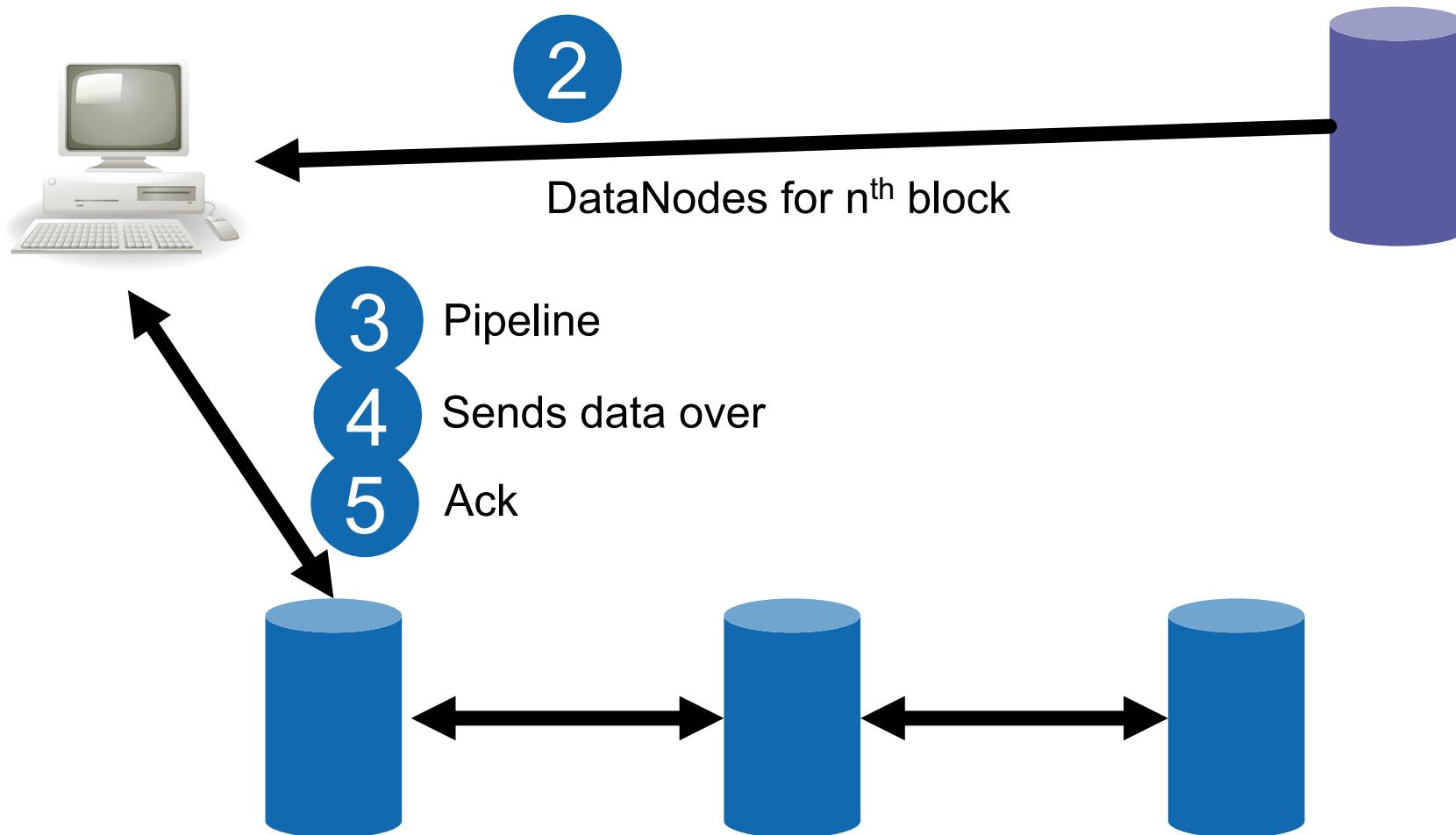
Client writes a file



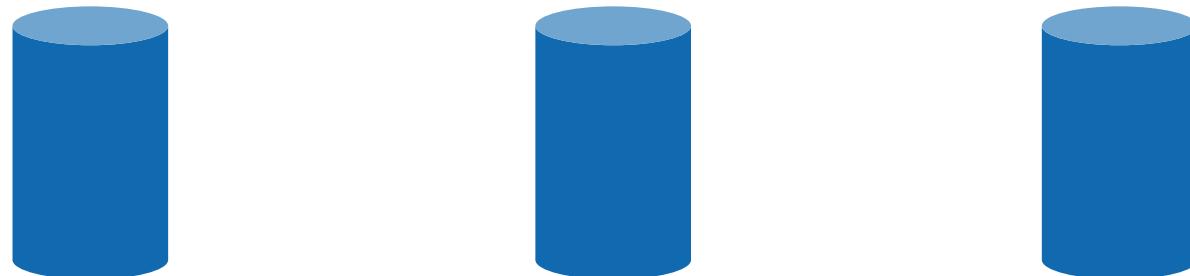
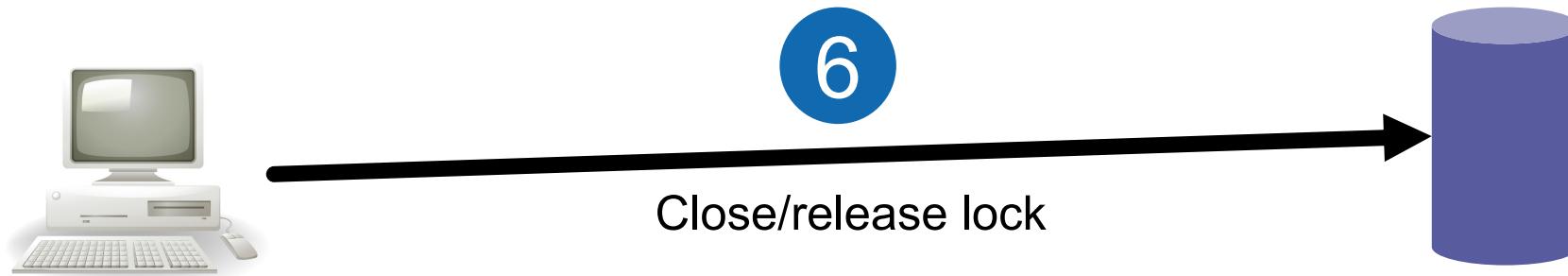
Client writes a file



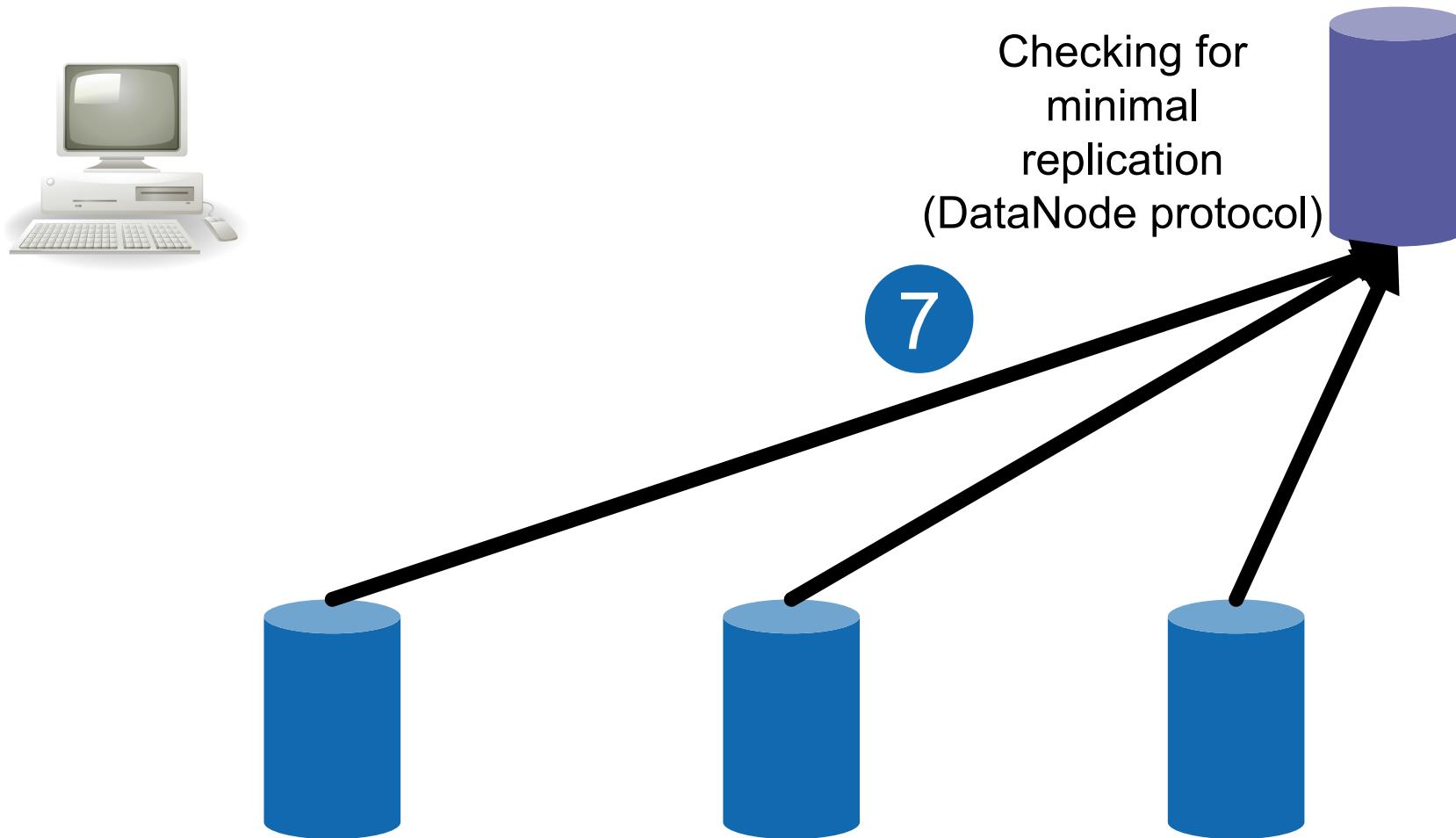
This is all done simultaneously under
DFSOutputStream (streaming through)



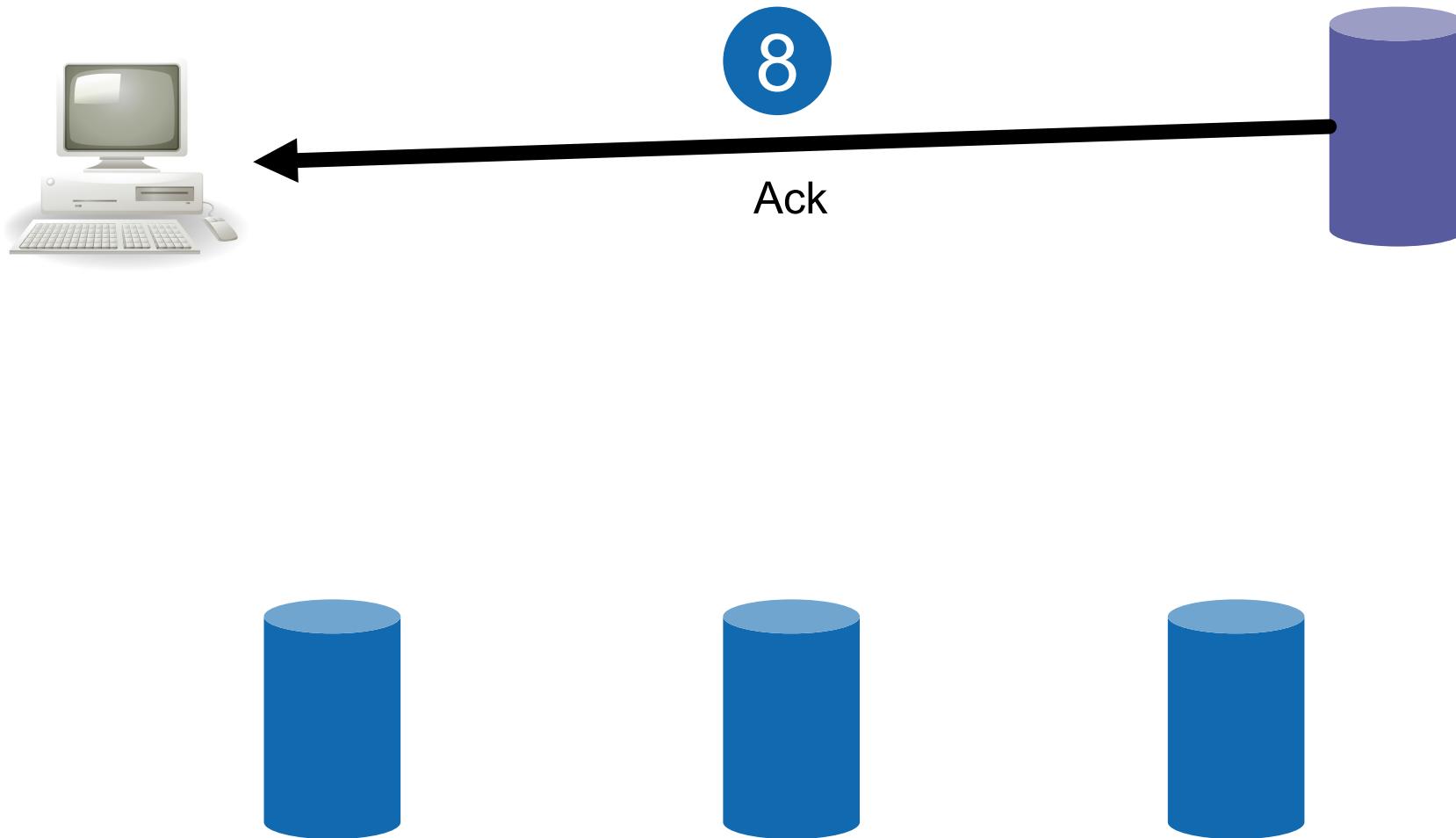
Client writes a file



Client writes a file



Client writes a file



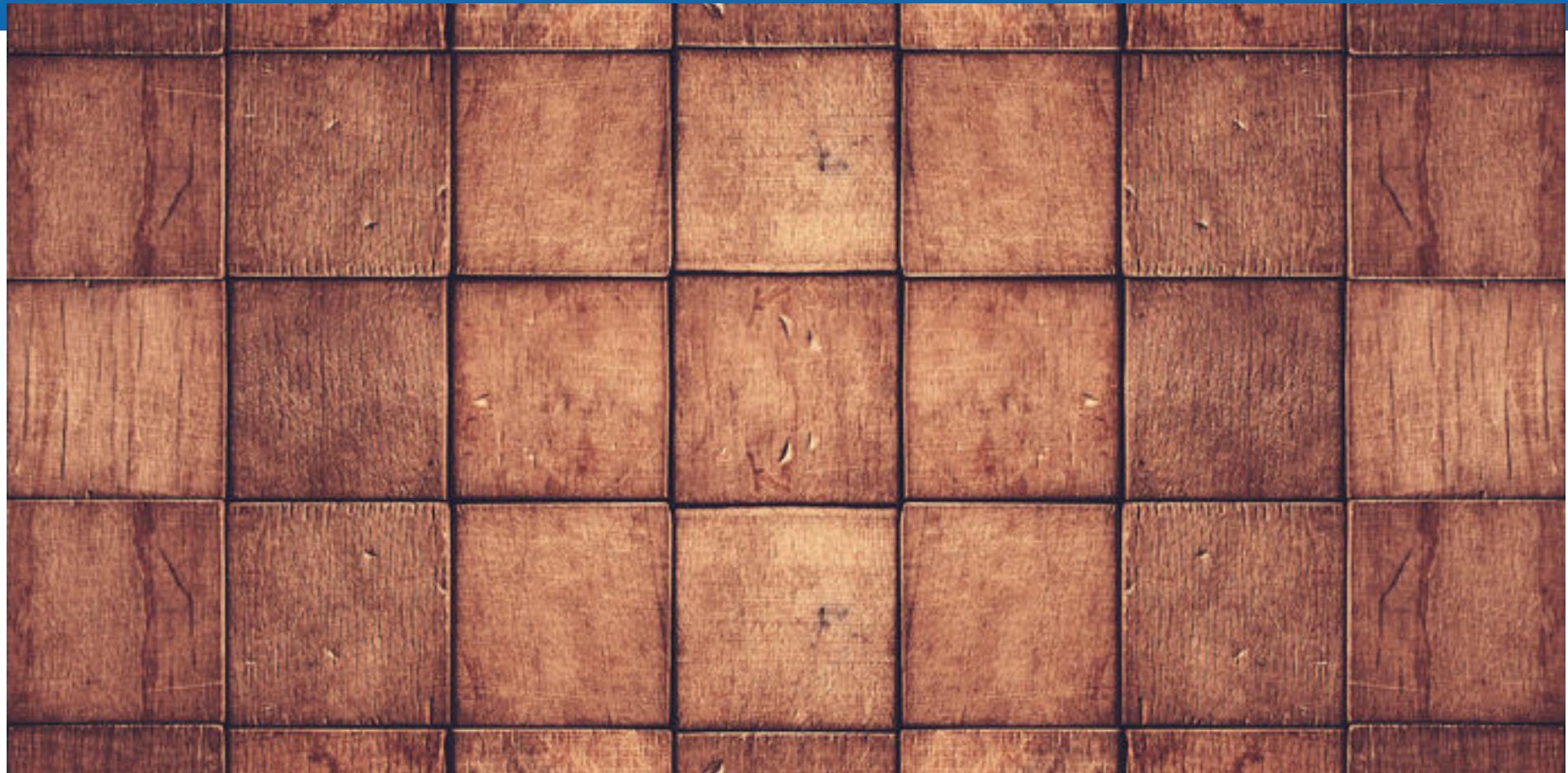
Client writes a file



replicates further asynchronously

9





Replicas

Replicas



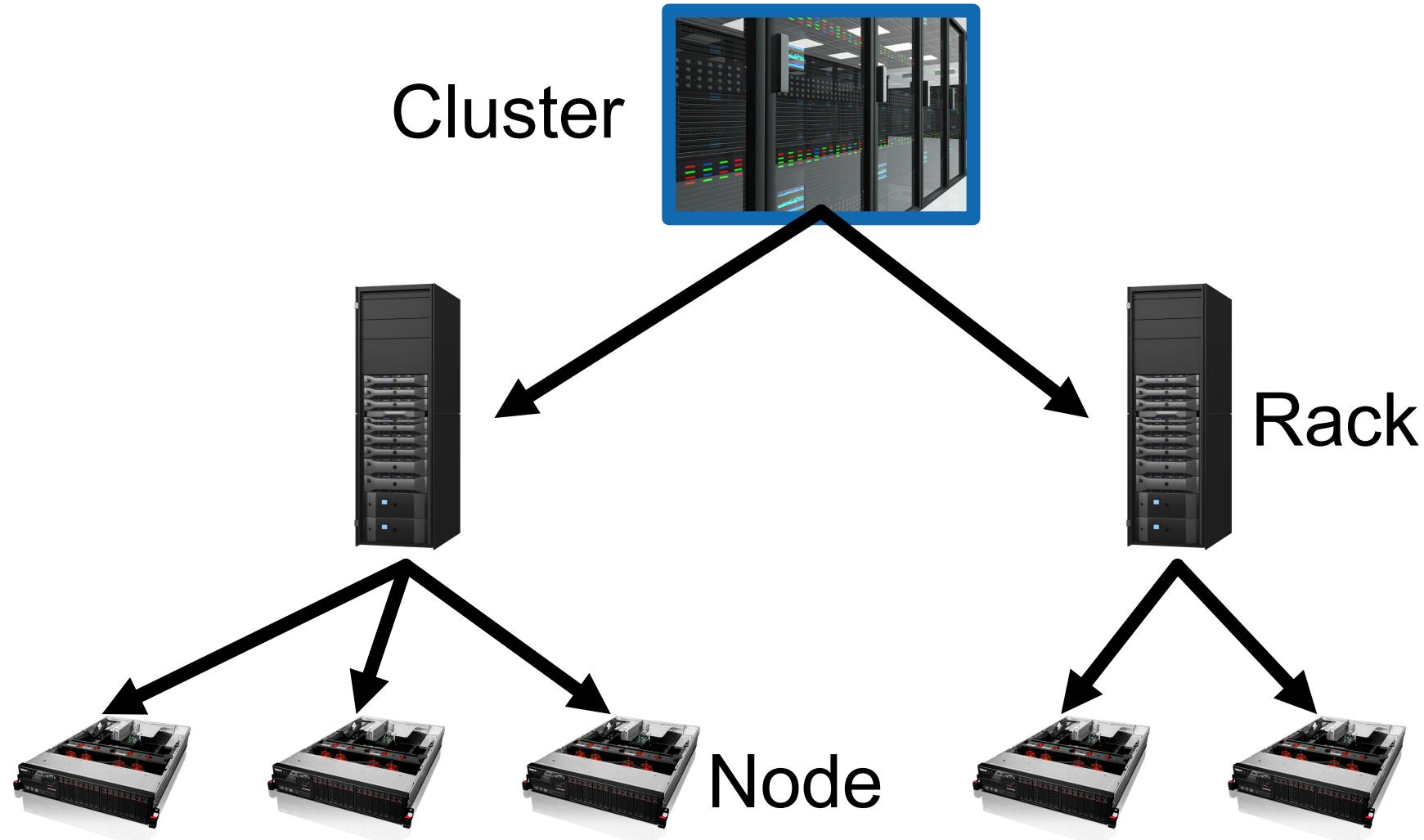
Number of replicas
specified

per file

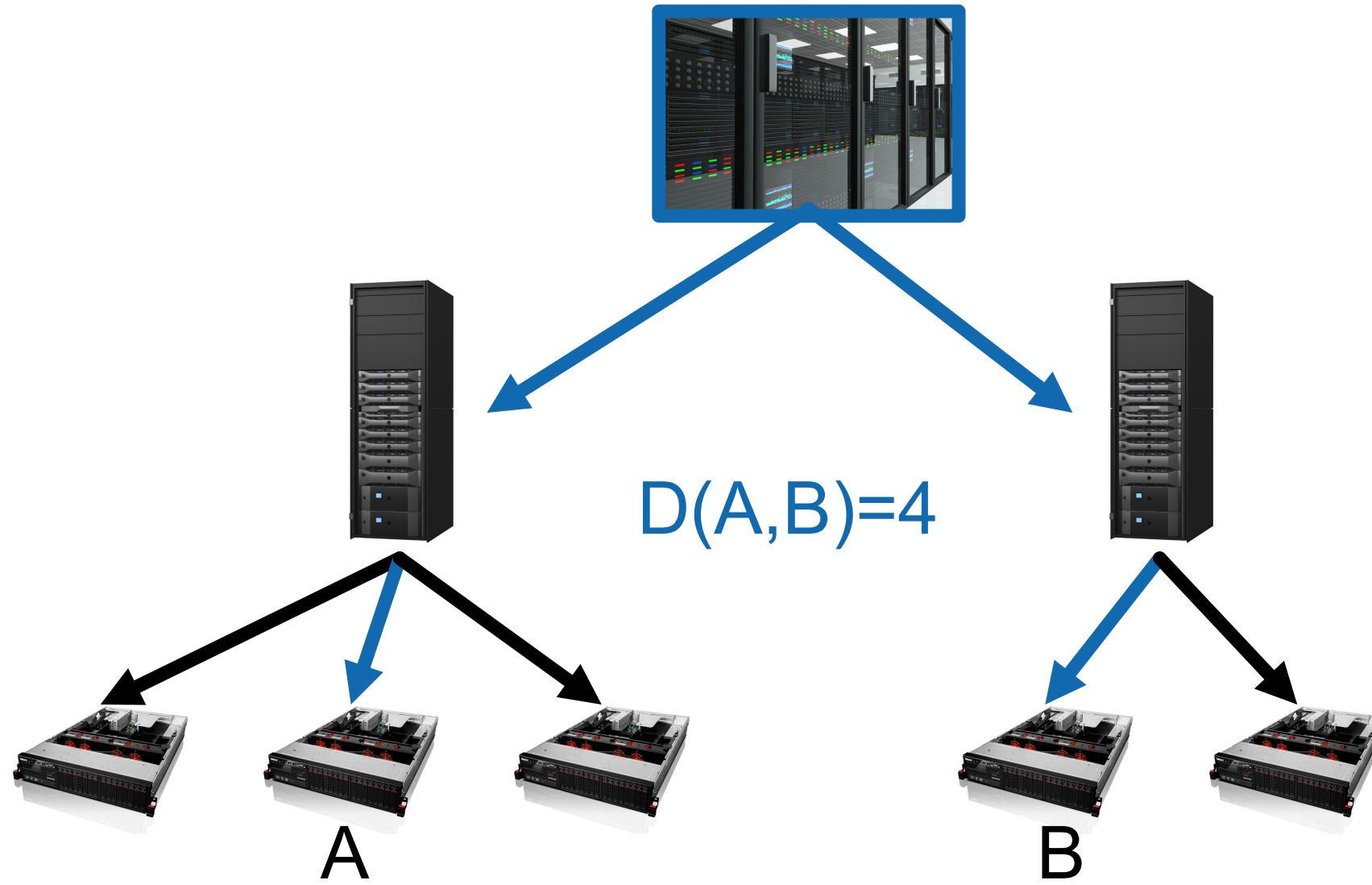
default:3



Replica placement: Reminder on topology



Replica placement: Distance



Replica placement



Replica 1: same node as client (or random), rack A

Replica placement



Replica 1: same node as client (or random), rack A



Replica 2: a node in a different rack B

Replica placement



Replica 1: same node as client (or random), rack A



Replica 2: a node in a different rack B



Replica 3: a node in same rack B

Replica placement



Replica 1: same node as client (or random), rack A



Replica 2: a node in a different rack B



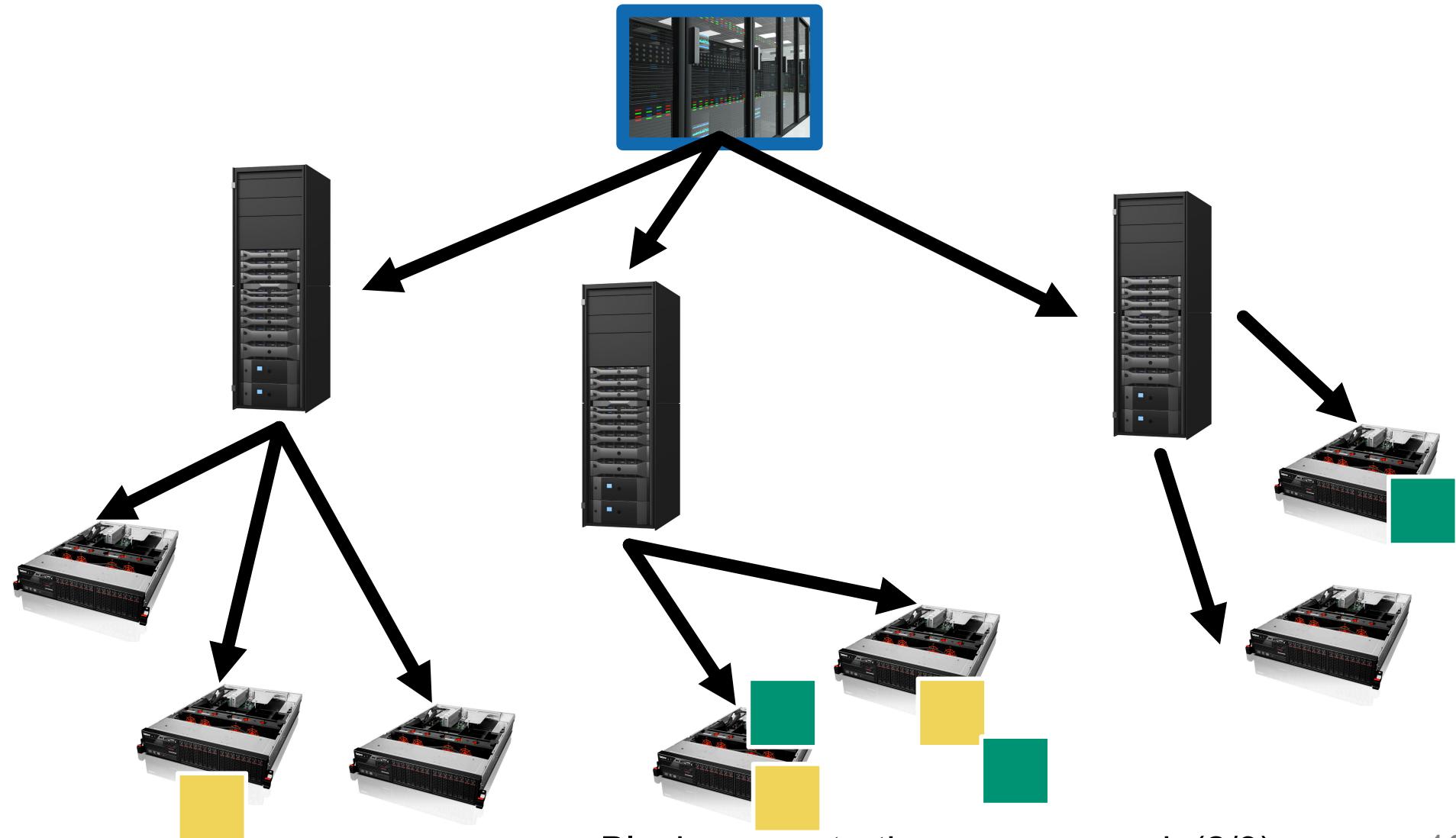
Replica 3: a node in same rack B



Replica 4 and beyond: random, but if possible:

- at most one replica per node
- at most two replicas per rack

If replicas 1+2 were on same rack...

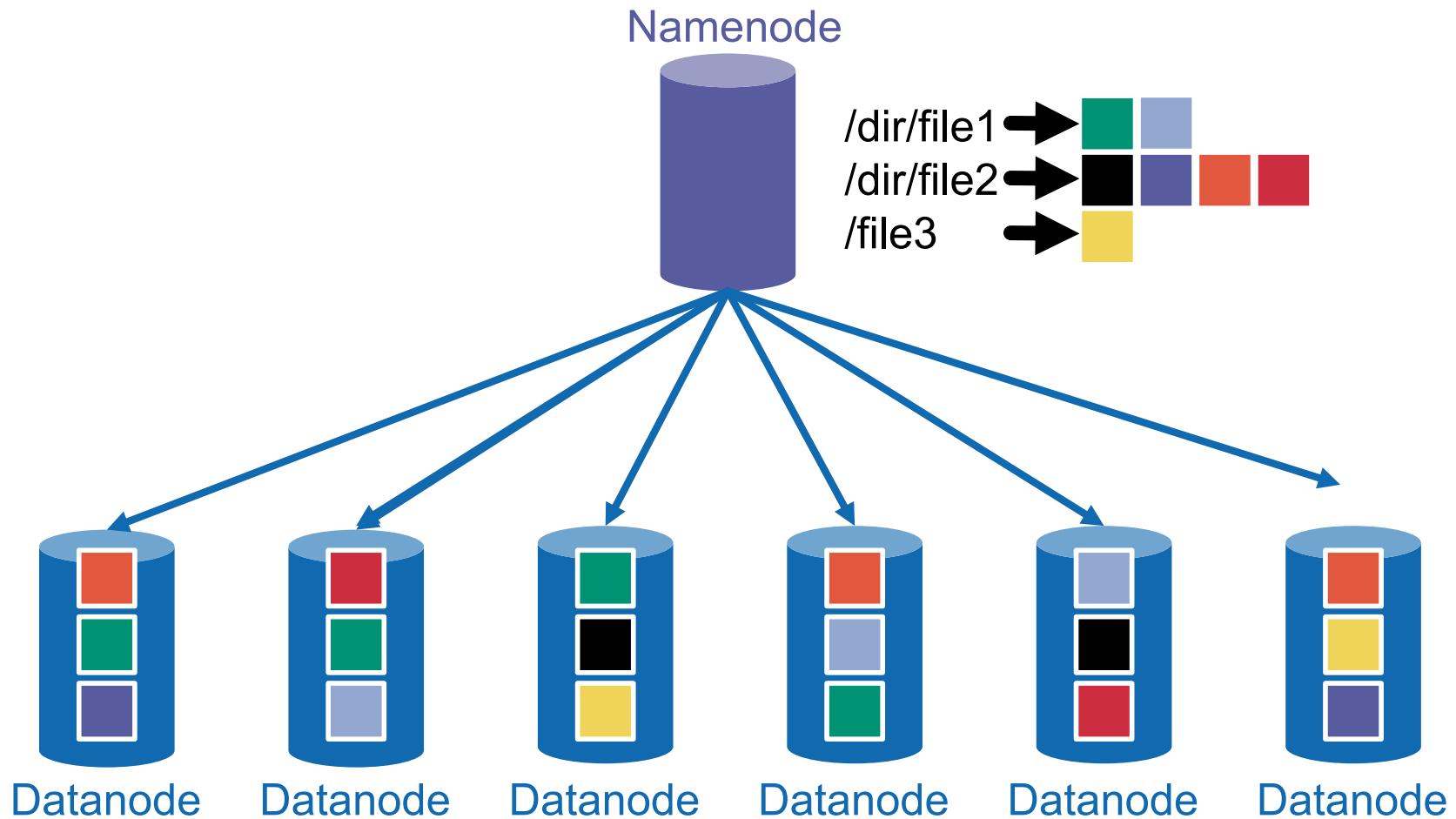


Block concentration on same rack (2/3)

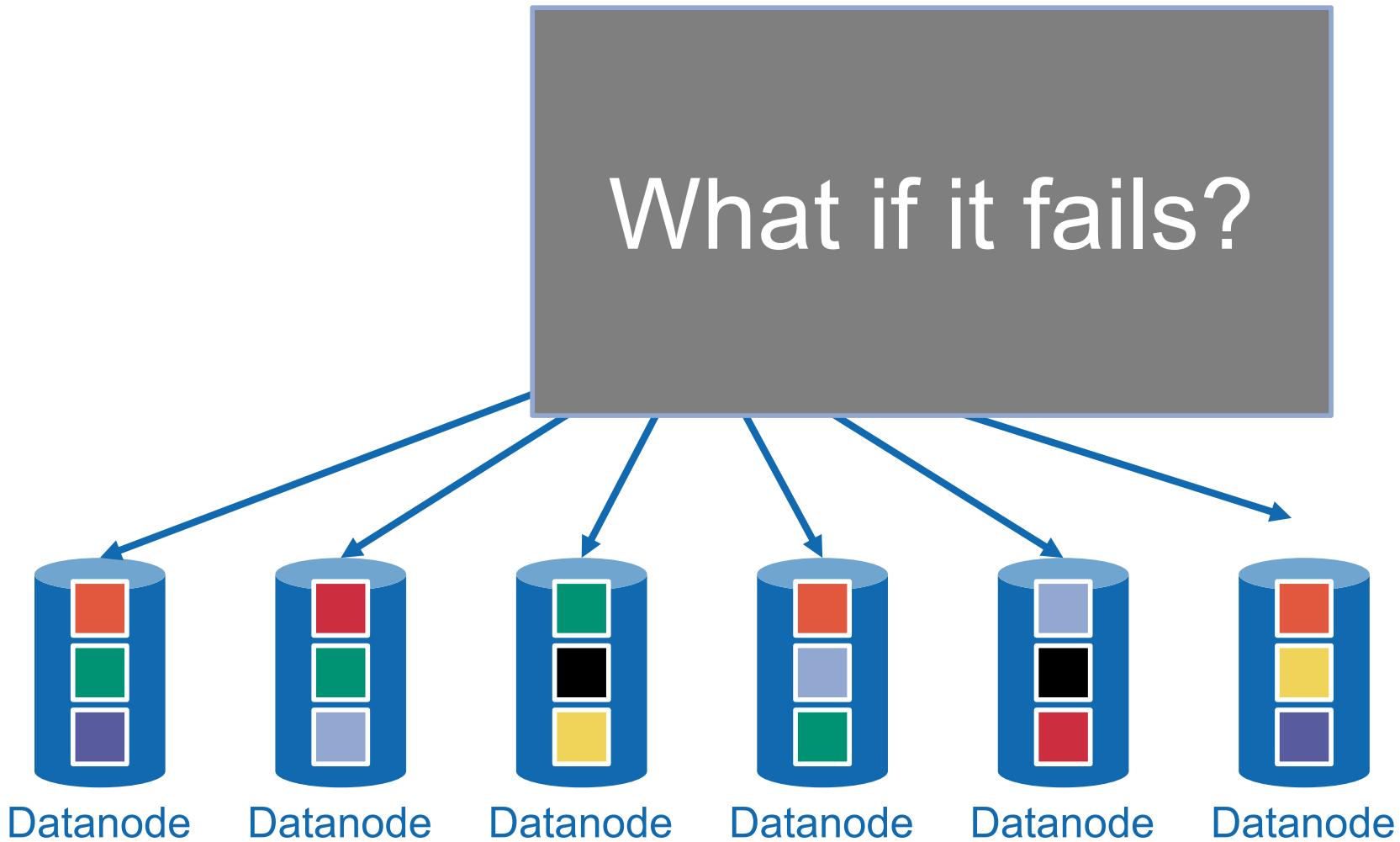


Performance and availability

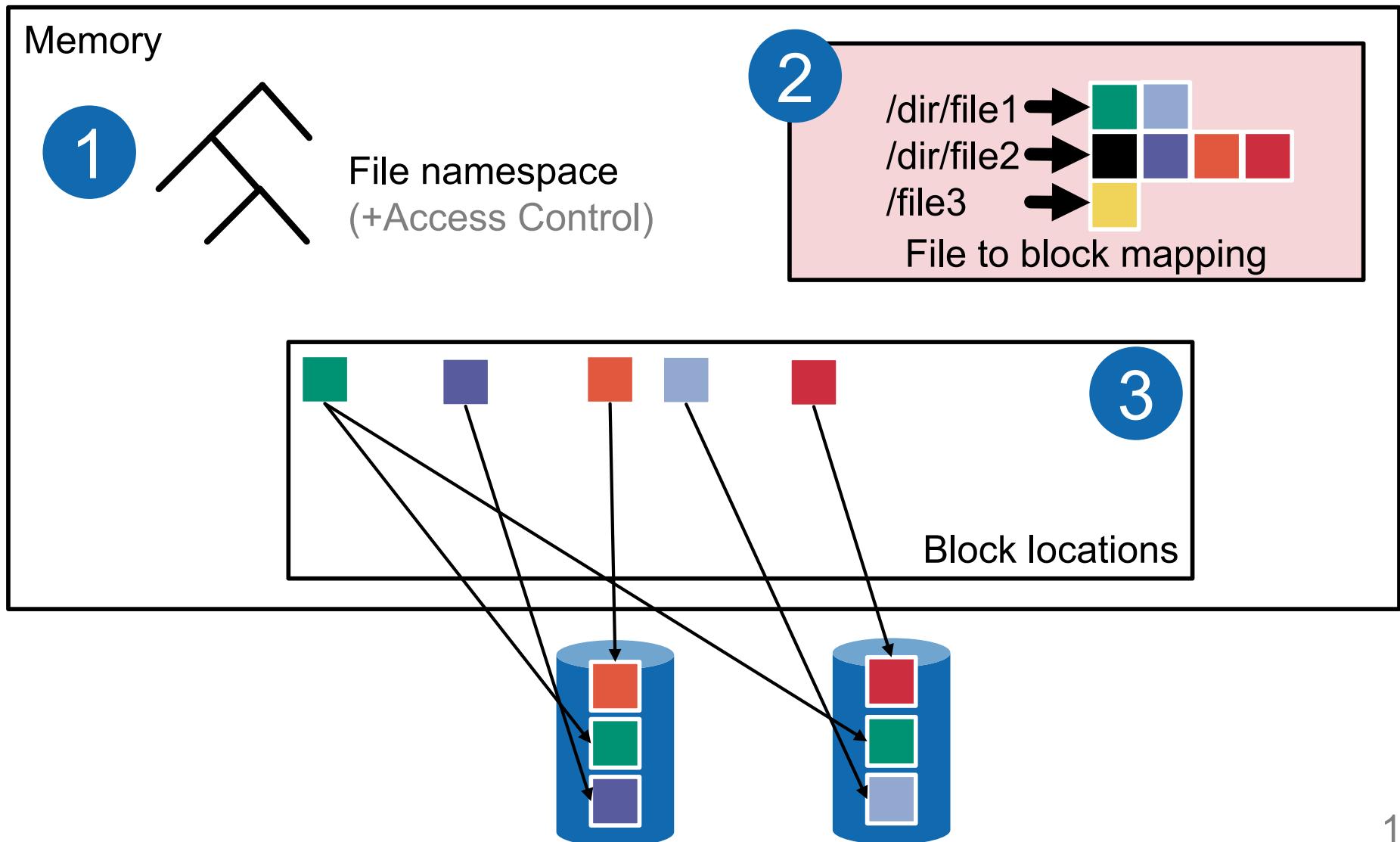
The NameNode is a single point of failure



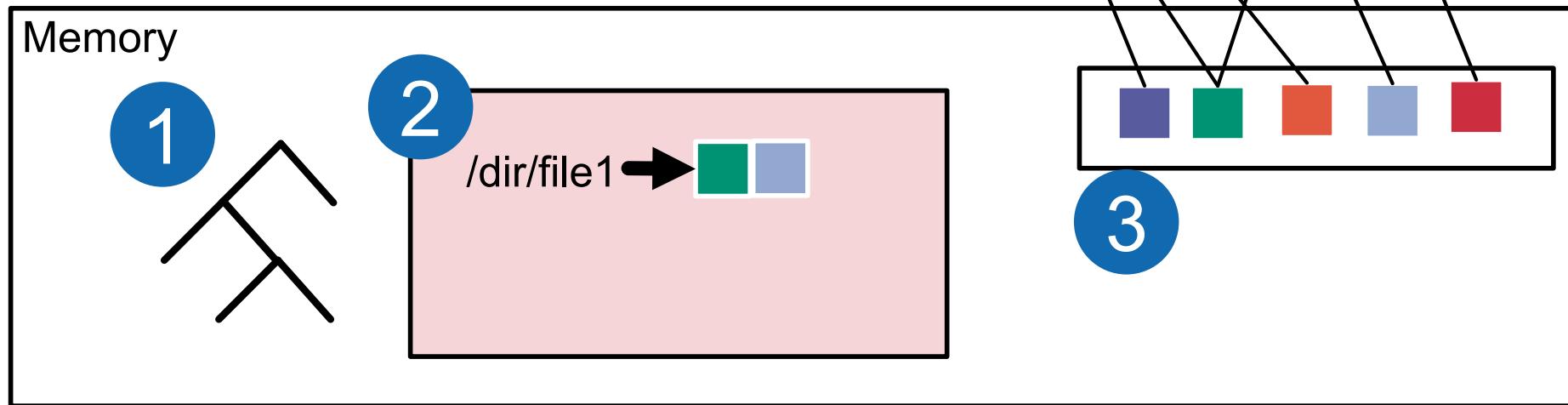
The NameNode is a single point of failure...



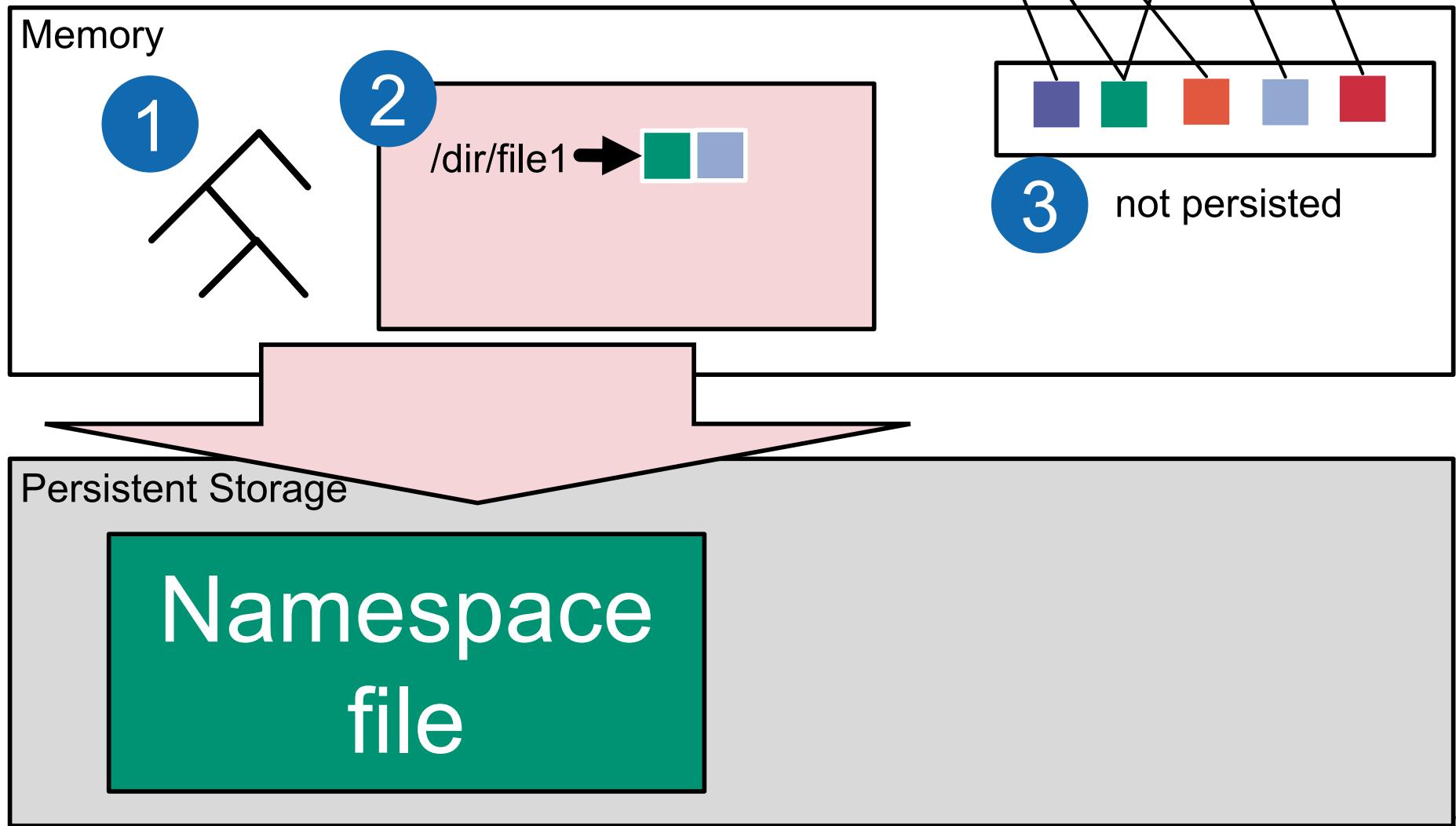
NameNode: all system-wide activity



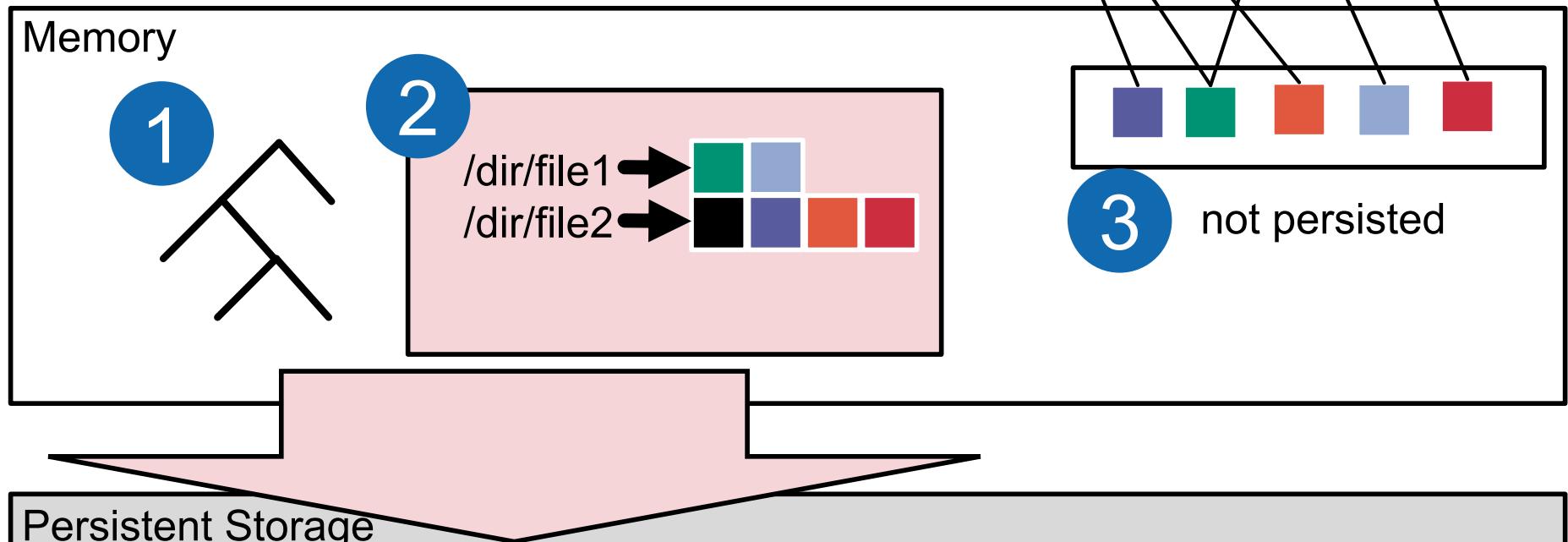
1. You want to persist



1. You want to persist

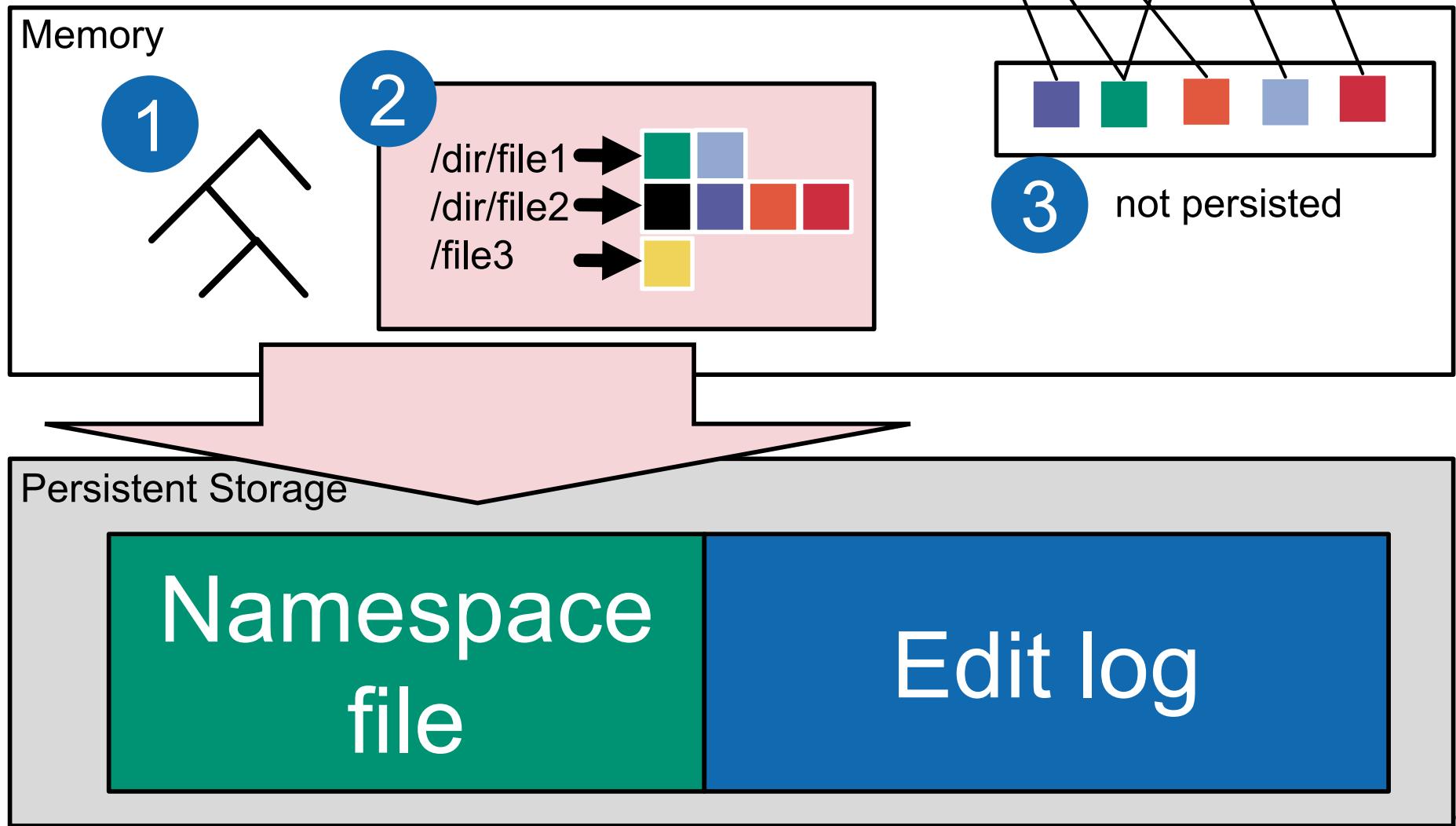


1. You want to persist

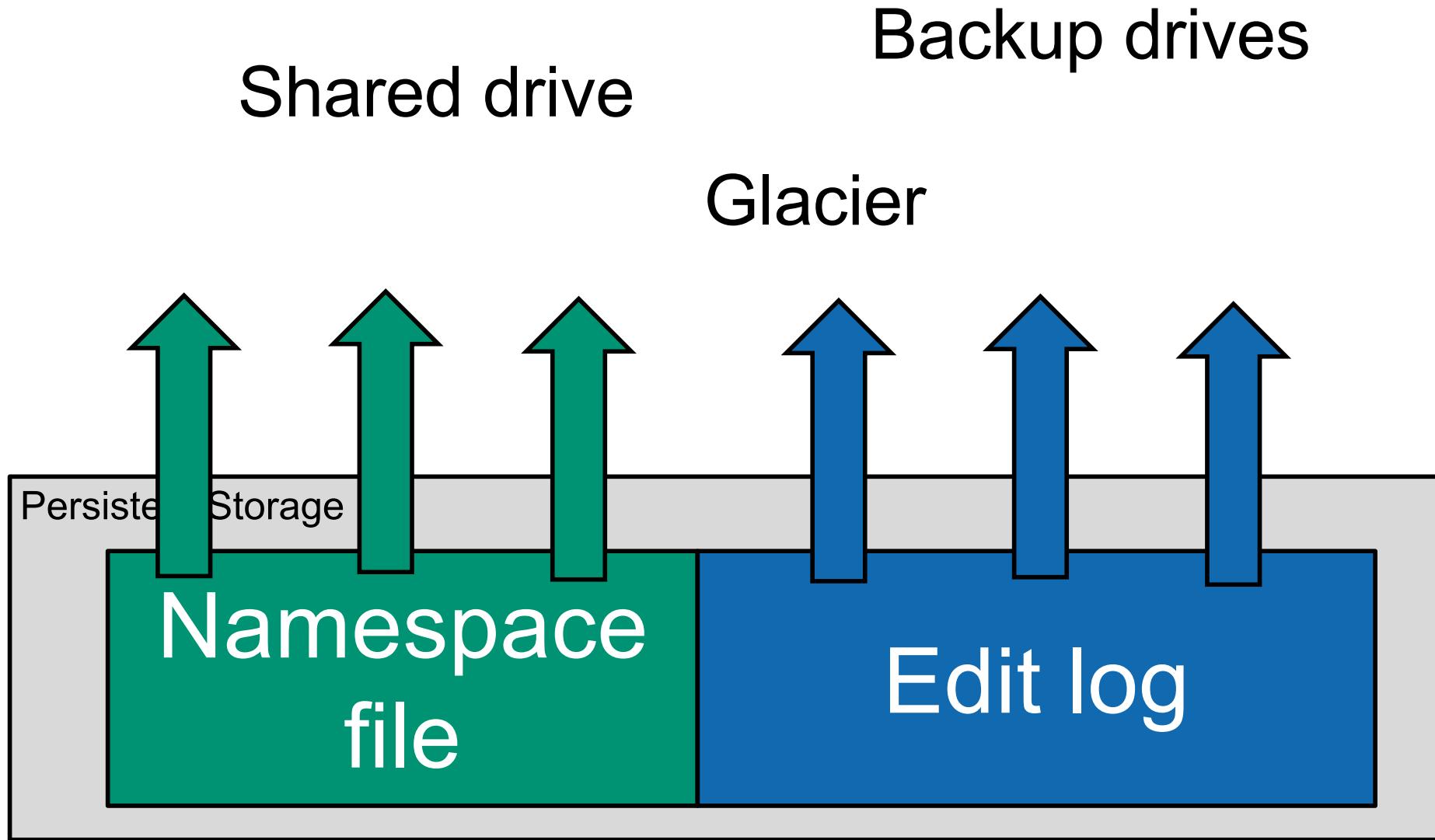


Namespace
file Edit log

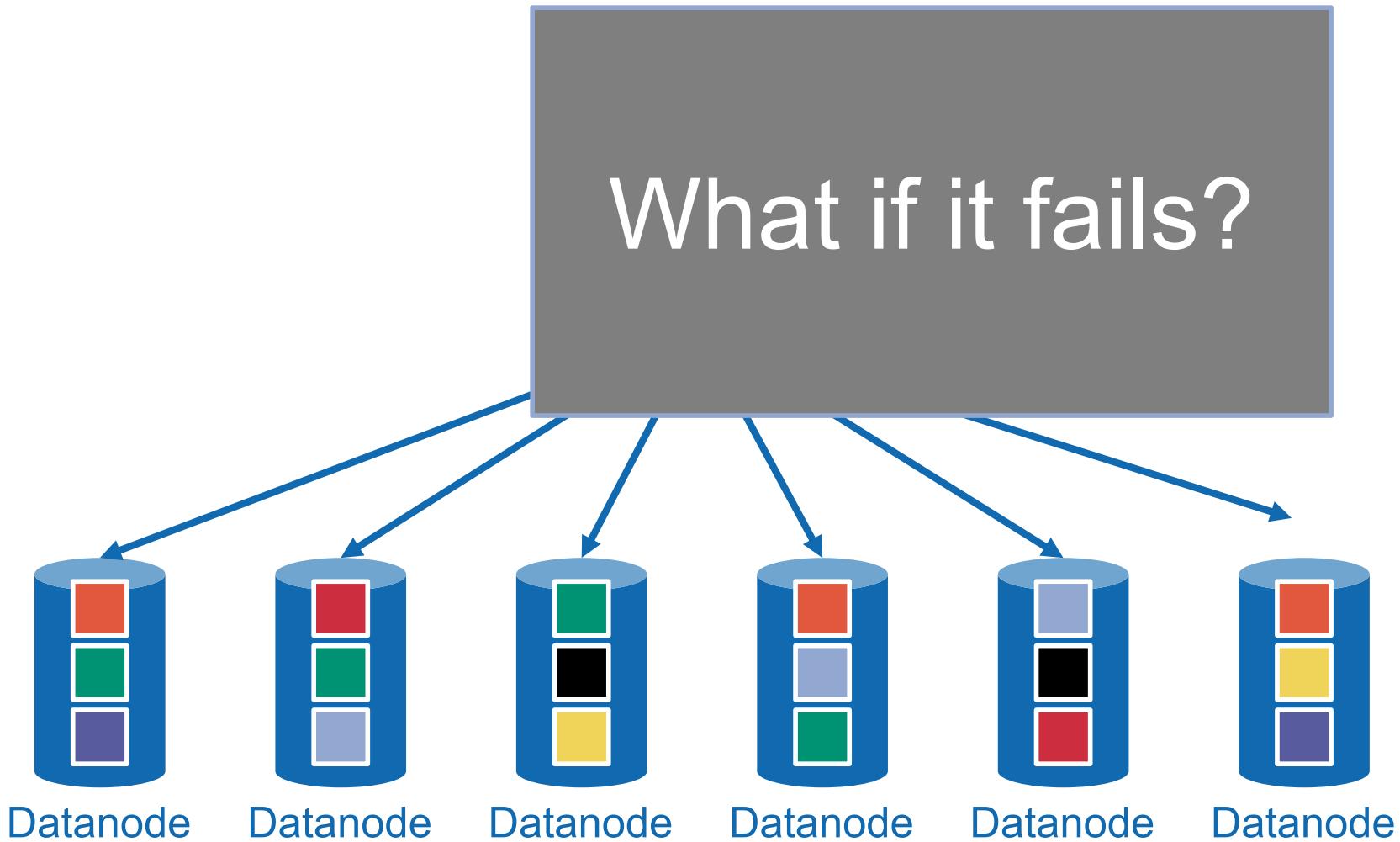
1. You want to persist



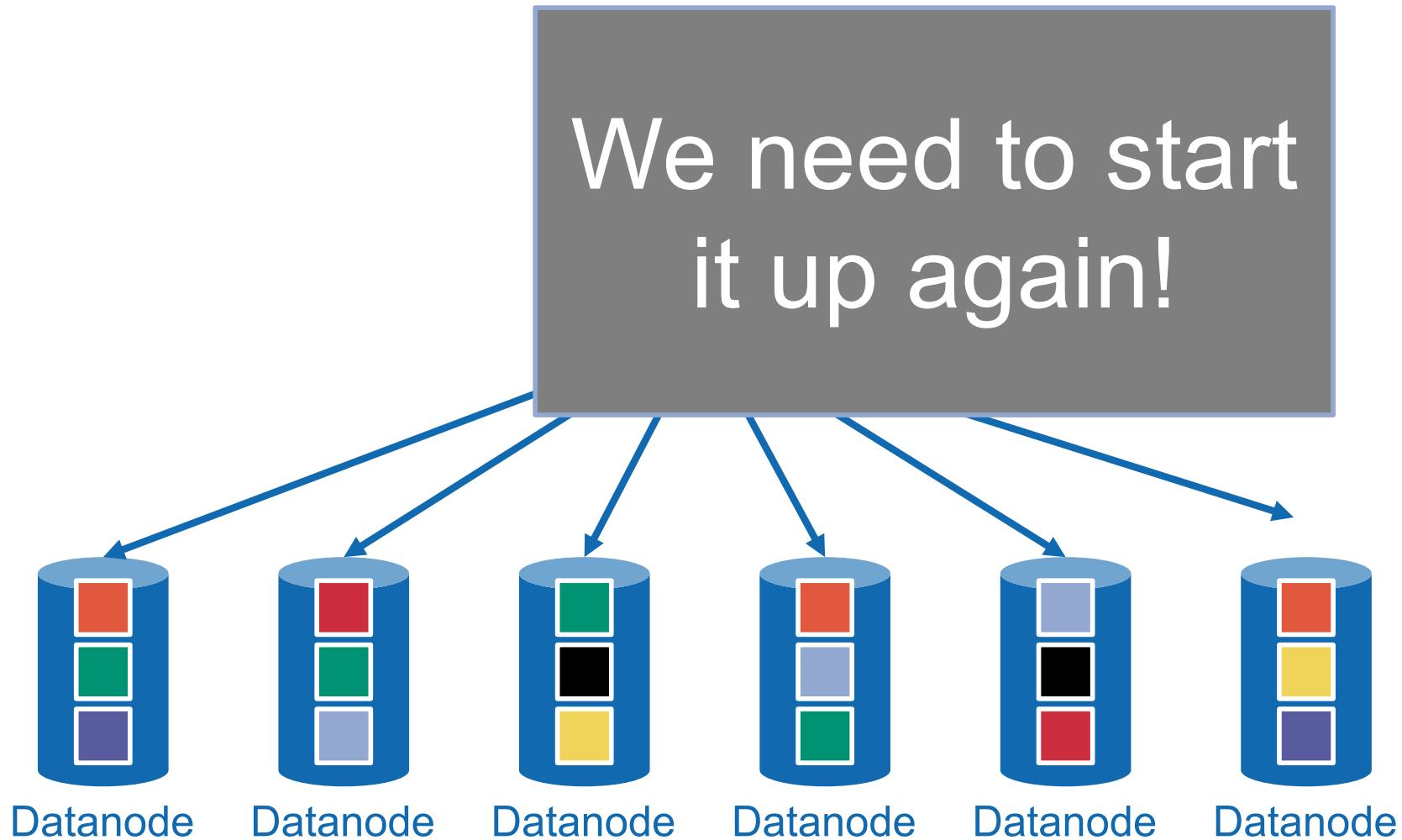
2. You want to backup



The namenode is a single point of failure...



The namenode is a single point of failure...



Namenodes: Startup

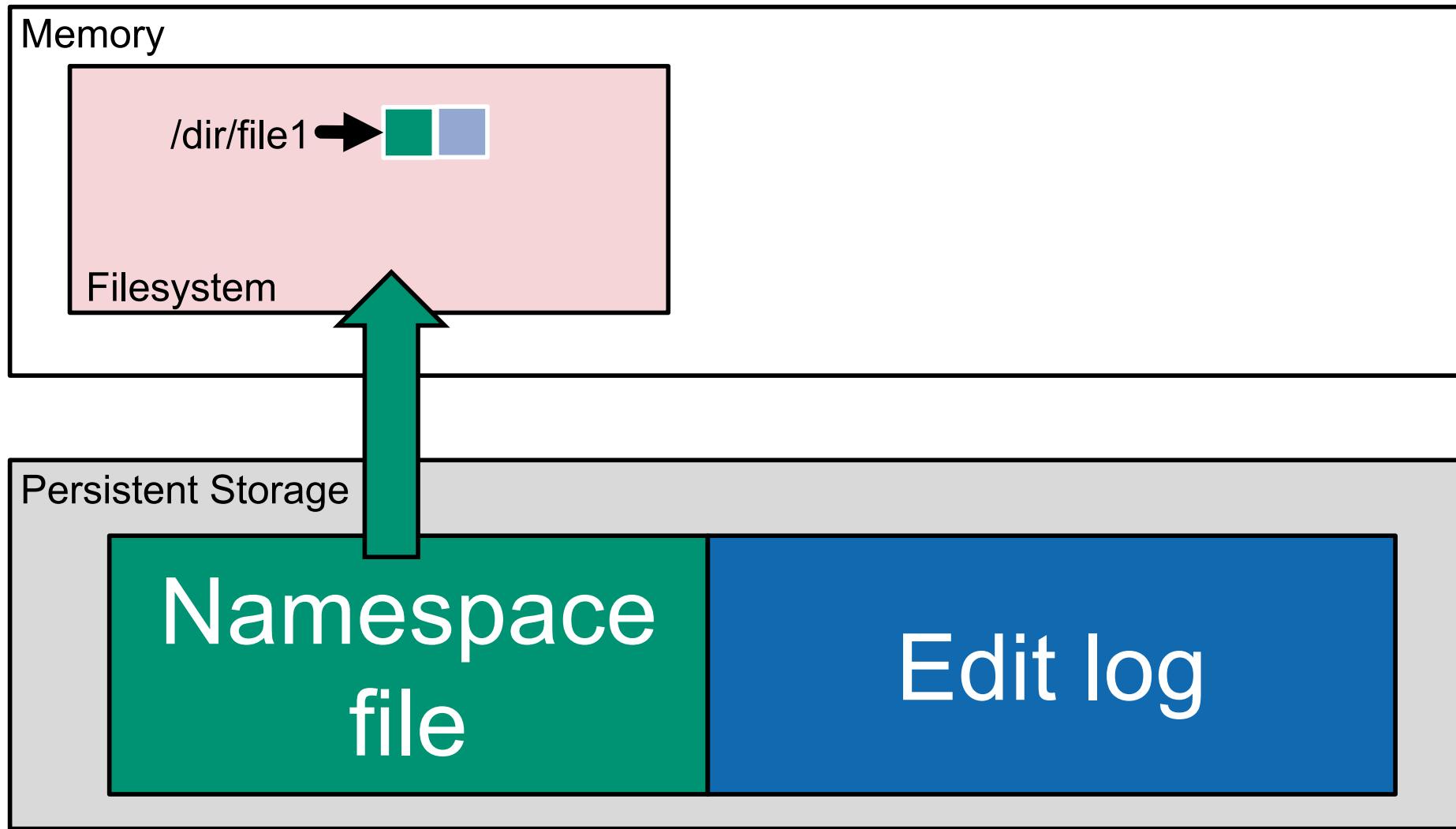
Memory

Persistent Storage

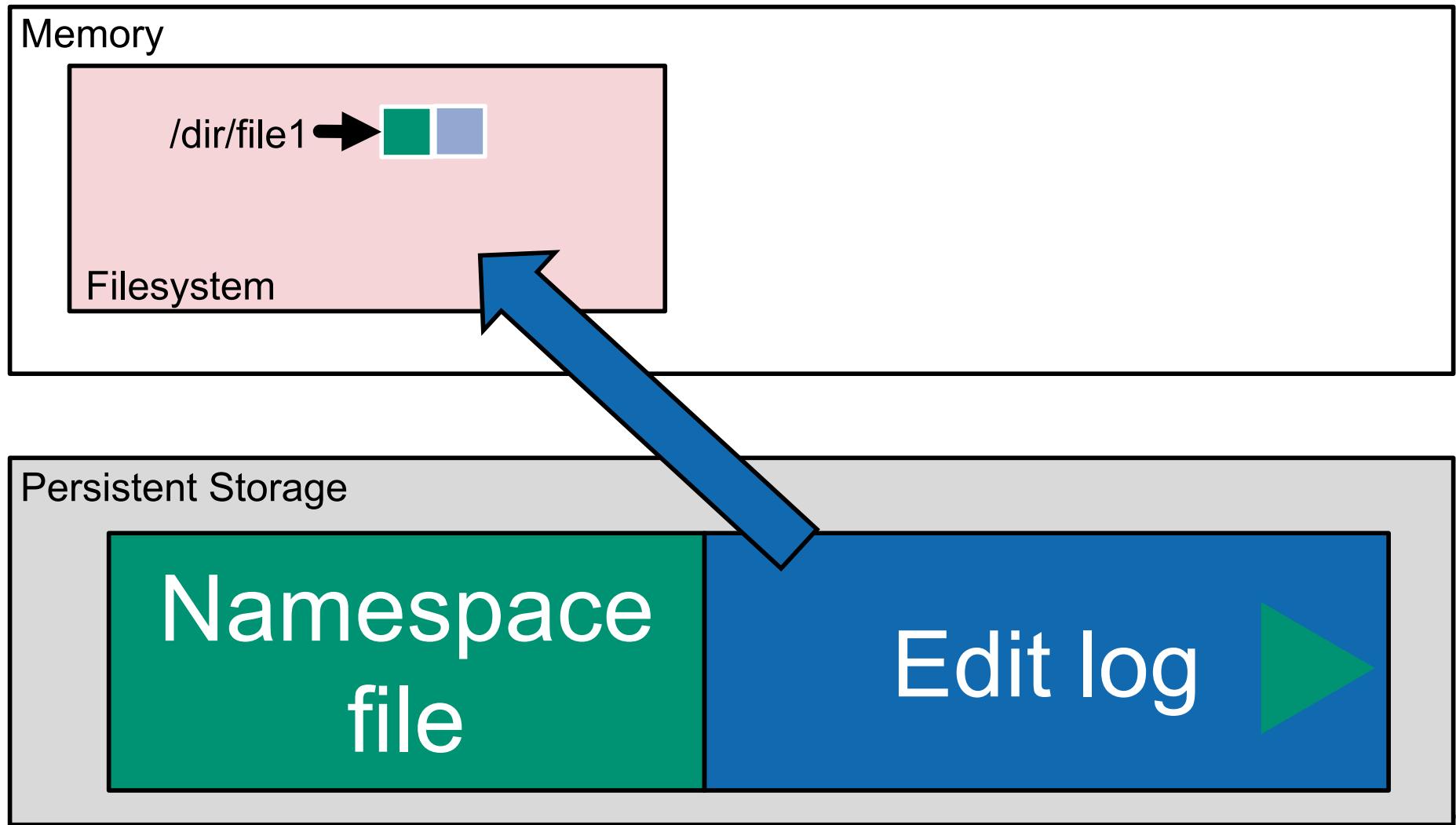
Namespace
file

Edit log

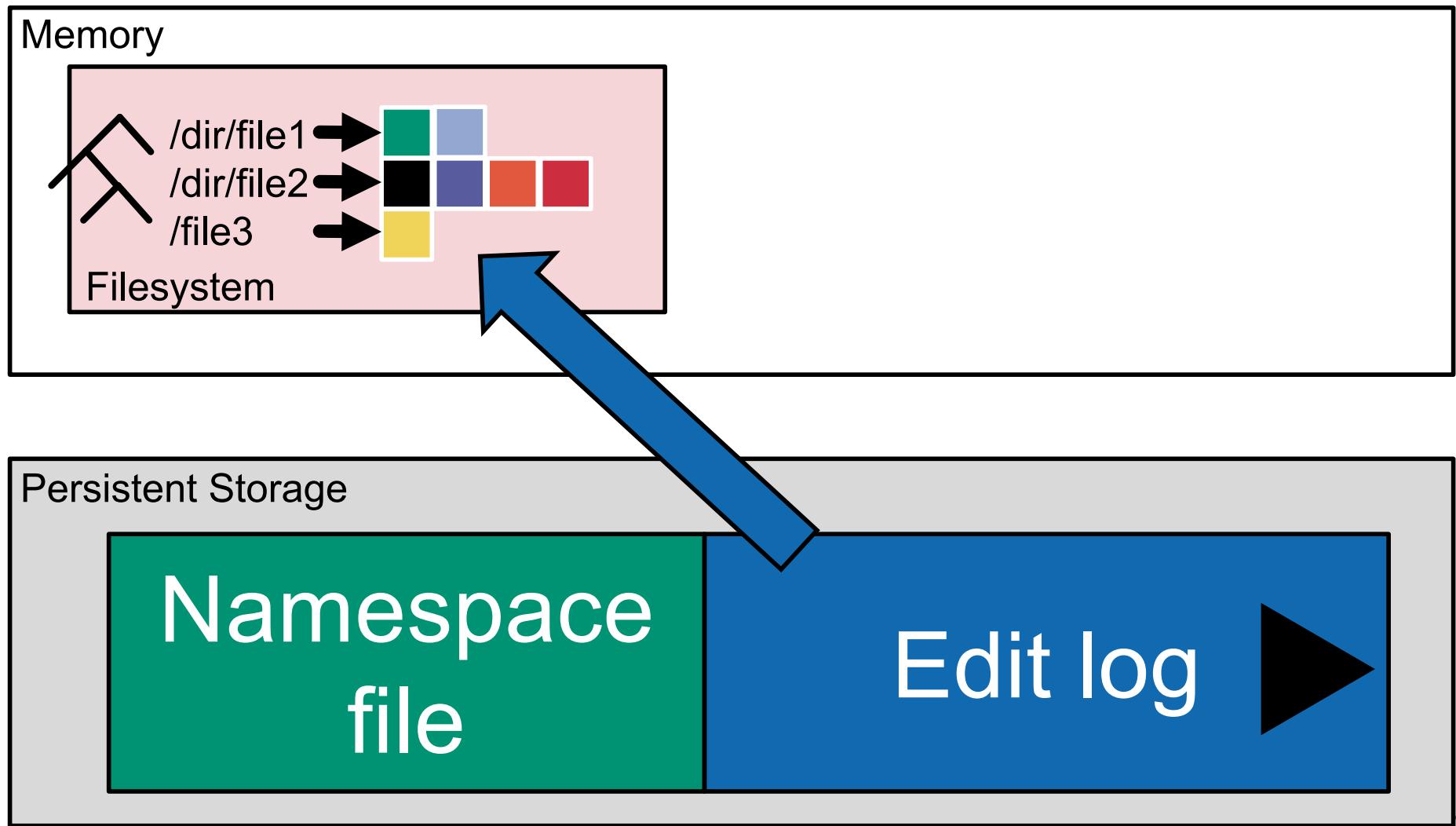
Namenodes: Startup



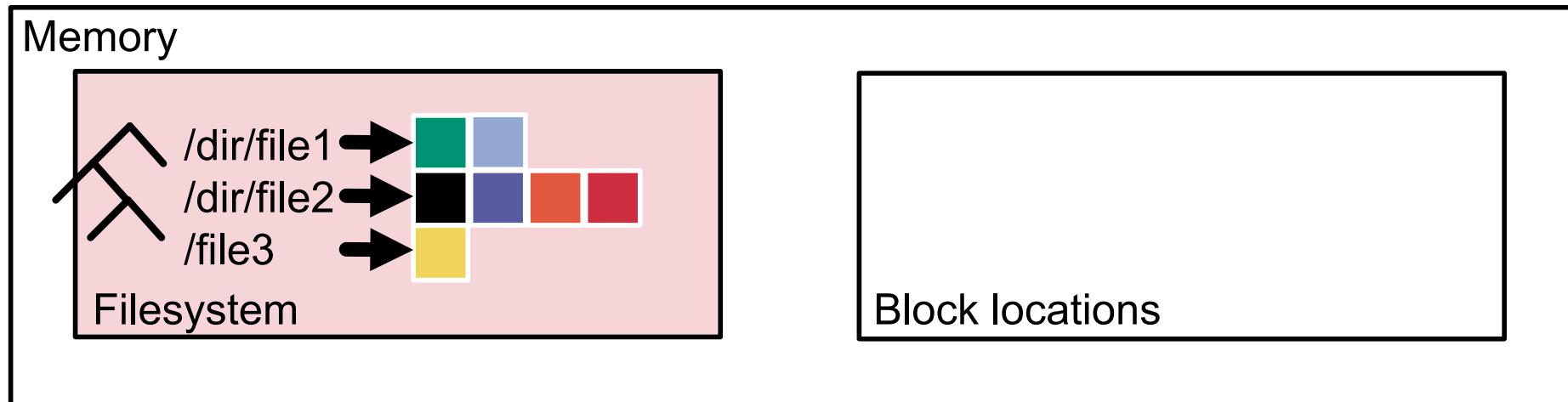
Namenodes: Startup



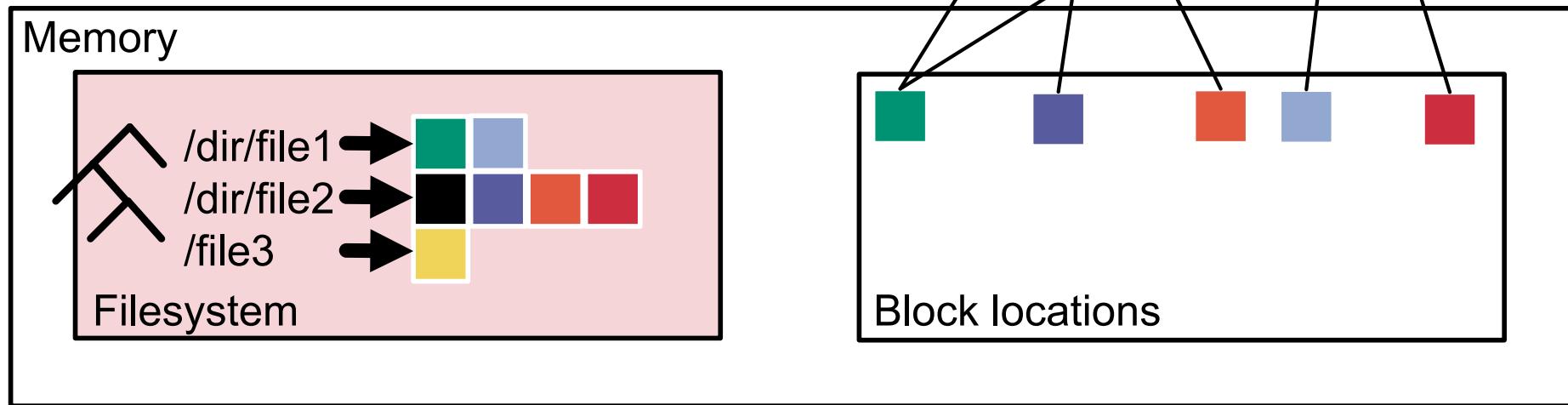
Namenodes: Startup



Namenodes: Startup



Namenodes: Startup



Starting a namenode...

**... takes
30 minutes!**

Starting a namenode...

**Can we do
better?**

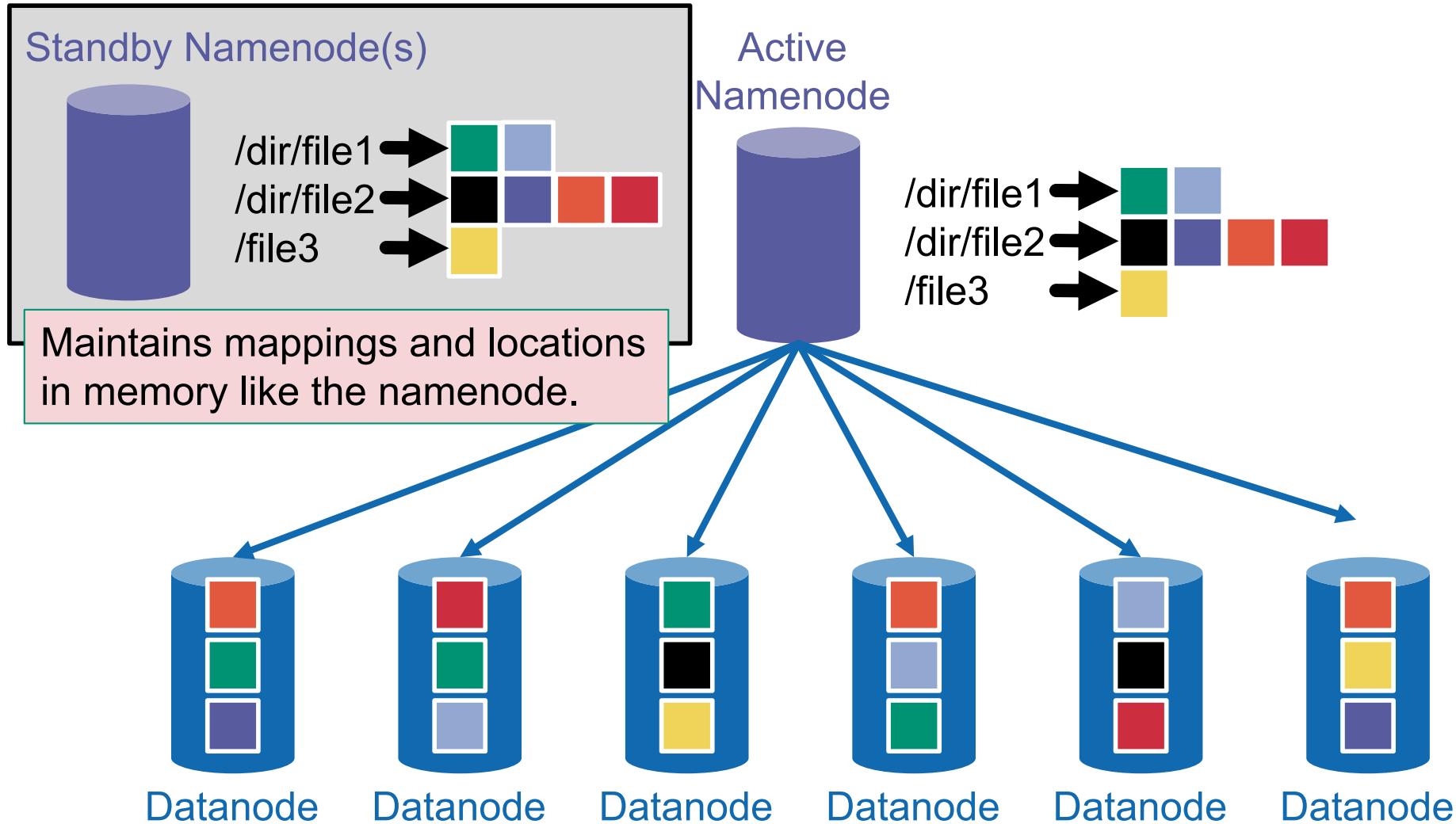
3. Checkpoints

Old namespace
file

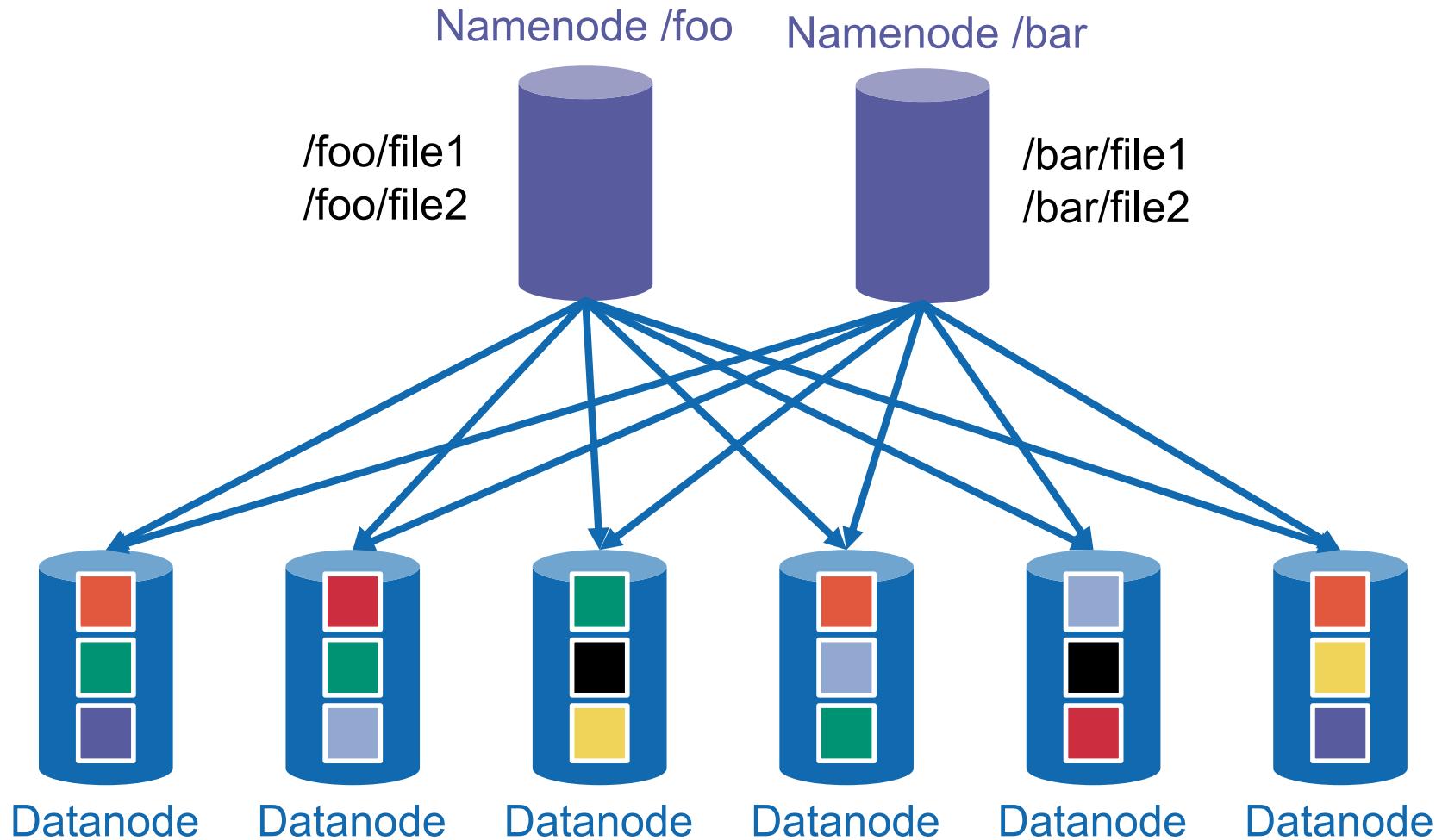
Edit log

New namespace file

4. High Availability (HA): Standby NameNodes



5. Federated DFS





Using HDFS

HDFS Shell: POSIX-like

```
$ hadoop fs -ls
```

```
$ hadoop fs -cat /dir/file
```

```
$ hadoop fs -rm /dir/file
```

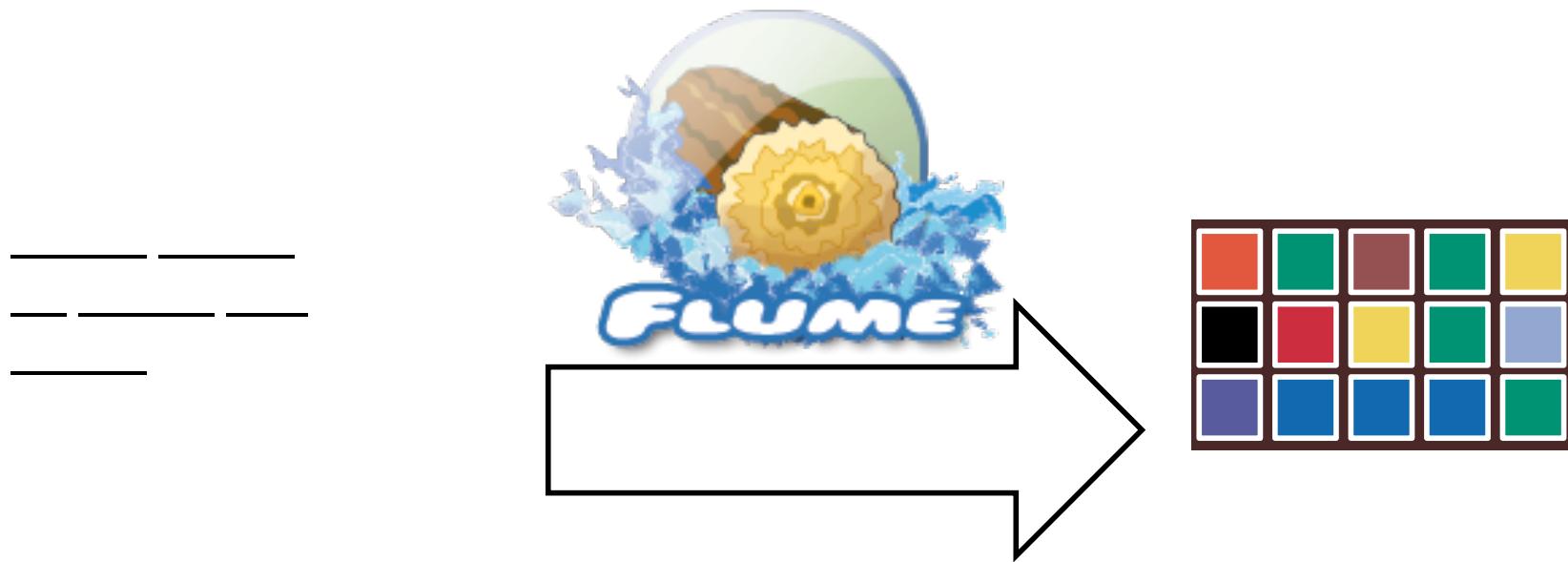
```
$ hadoop fs -mkdir /dir
```

HDFS Shell: upload and download

```
$ hadoop fs -copyFromLocal  
    localfile1 localfile2  
    /user/hadoop/hadoopdir
```

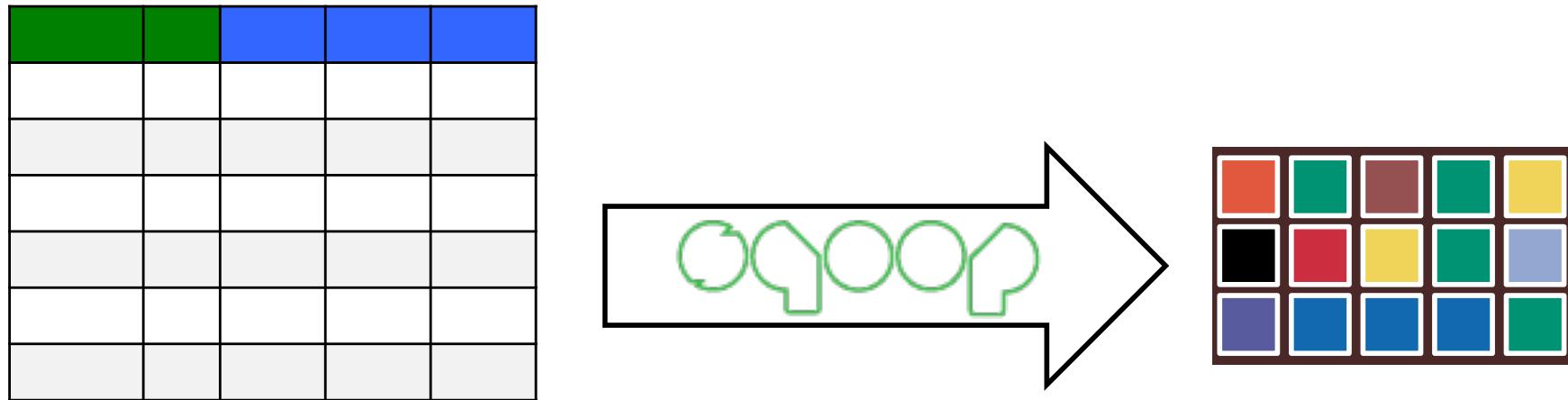
```
$ hadoop fs -copyToLocal /user/hadoop/file  
    localfile
```

Populating HDFS: Apache Flume



Collects, aggregates, moves log data
(into HDFS)

Populating HDFS: Apache Sqoop



Imports from a relational database

Pointers

Official documentation

<http://hadoop.apache.org/docs/r3.2.1/>

GFS Paper

On course website

Java API

<https://hadoop.apache.org/docs/r3.2.1/api/index.html>