Introduction to parallel filesystems

Philippe.Wautelet@idris.fr

CNRS** - **IDRIS

PATC Training session
Parallel filesystems and parallel IO libraries
Maison de la Simulation / March 5th and 6th 2015

Plan

Architecture of parallel supercomputers

Parallel supercomputer

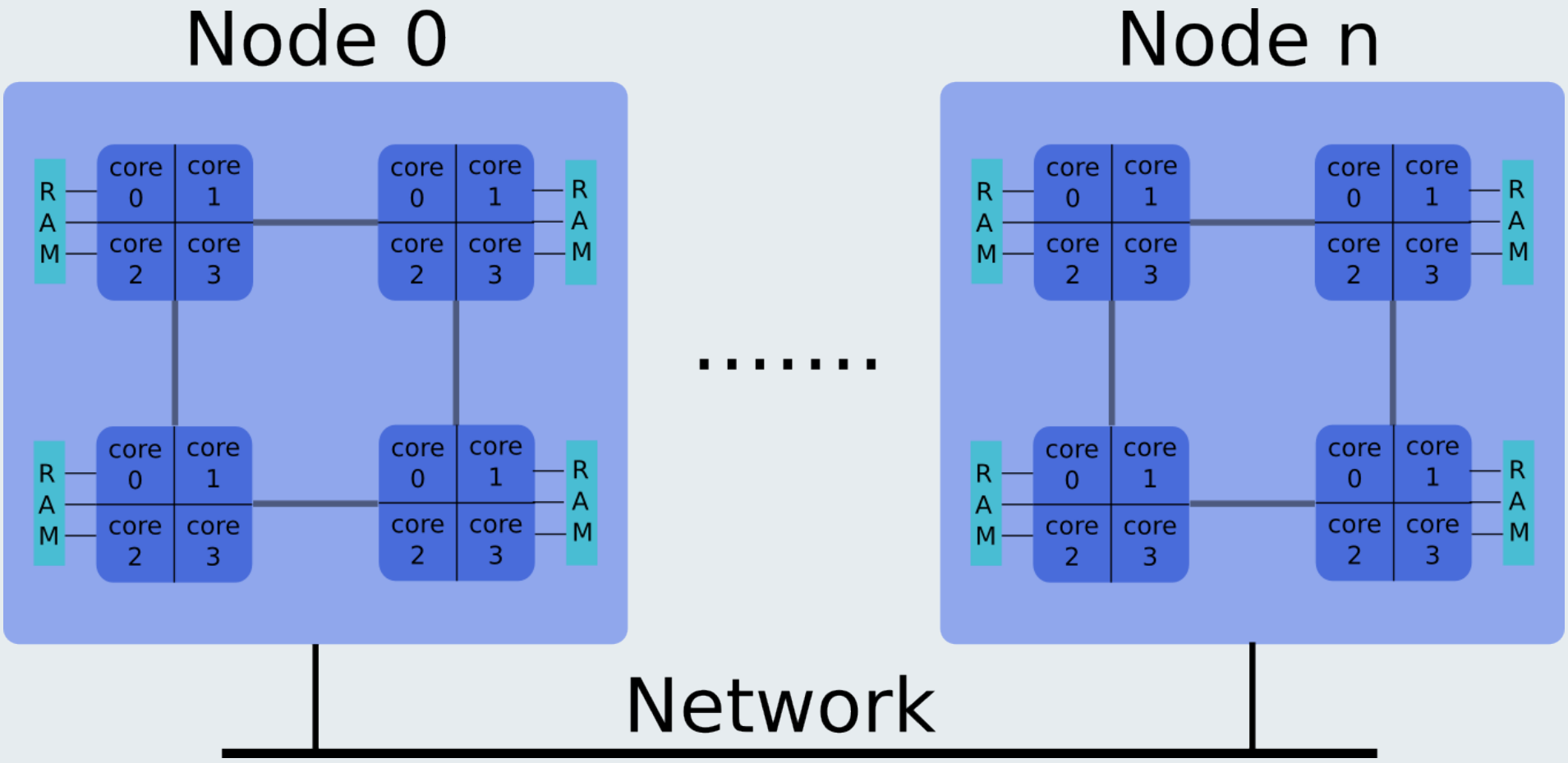
A parallel computer consists of:

- a set of computing cores having access to a local memory and grouped into nodes;
- a fast and efficient interconnection network;
- a fast storage system.

Each node contains computing cores possibly assisted of accelerators (GPGPU, Xeon Phi, FPGA...).

- All cores within a node have access to the same memory (shared memory architecture).
- However, usually, the cores of a node can not access the memory of another node (distributed memory architecture).
- Fully-shared memory machines exists in which all the cores can access the memory of any node. In this kind of machine, the memory accesses are highly non-uniform (NUMA) because, according to where the memory is relative to a given core, performance (throughput and latency) will vary. Performance can also vary within a node, but in a much less pronounced way.

Typical machine



Parallel supercomputers: examples

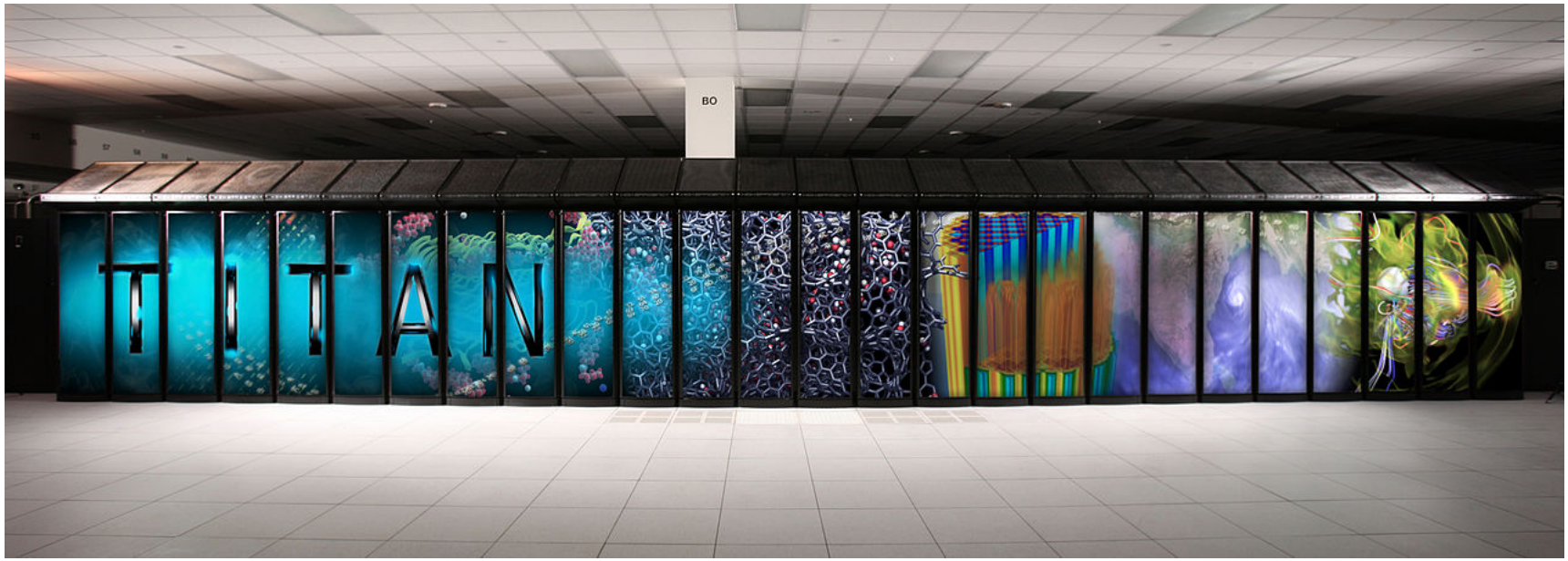
Tianhe-2 (National Super Computer Center in Guangzhou, China) (#1 top 500 2014/11)

- TH-IVB-FEP Cluster running Kylin Linux
- Peak: 54,9 Pflop/s, LINPACK performance: 33,9 Pflop/s
- 3,120,000 cores and 1,375 TiB of memory (16,000 nodes with 2 12-cores 2.2 GHz Xeon Ivy Bridge, 64 GiB of memory and 3 Xeon Phi 31S1P1 with 8 GiB of memory)
- TH Express-2 network
- H^2FS parallel filesystem with 12.4 PiB (>1 TiB/s burst, 100 GiB/s sustained)
- 17.8 MW (24 MW including cooling)

Parallel supercomputers: examples

Titan (ORNL, USA) (#2 top 500 2014/11)

- Cray XK7 running linux
- Peak: 27.1 Pflop/s, LINPACK performance: 17,6 Pflop/s
- 560,640 cores and 710 TiB of memory (18,688 nodes with 1 16-cores 2.2 GHz AMD Opteron 6274, 32 GiB of memory and 1 Nvidia Tesla K20X with 6 GiB of memory)
- Cray Gemini interconnect
- Lustre parallel filesystem with a capacity of 40 PiB (1.4 TiB/s peak)
- 8.2 MW



What is a filesystem?

What is a filesystem?

Main roles

A filesystem has two main functions:

- To organize and maintain the files namespace
- To store the contents of the files and their attributes

Data

They correspond to the actual file contents.

Metadata

Metadata is a set of information about files. They contain, for example:

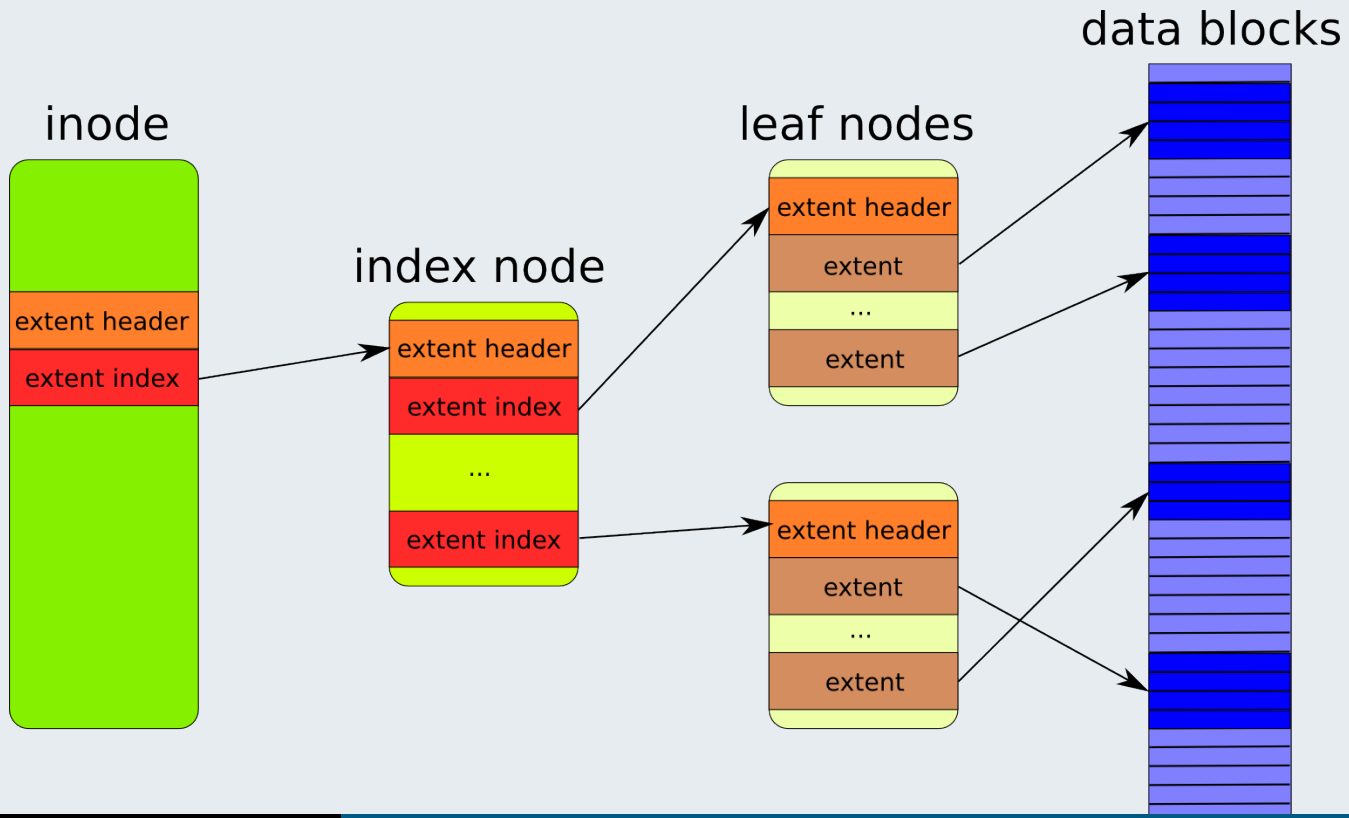
- Data position on the disks
- File sizes
- Creation, last modification and last access dates
- The owners (UID and GID) and the permissions
- ...

Sequential filesystems

Definition

- A local sequential filesystem is a filesystem that can not be directly accessed remotely.
- Only one client can access it (the operating system of the machine).
- In general, there is no parallelism (one simultaneous access at a time).

Example: structure of a *large* file on an ext4 filesystem



Parallel filesystems

Definitions

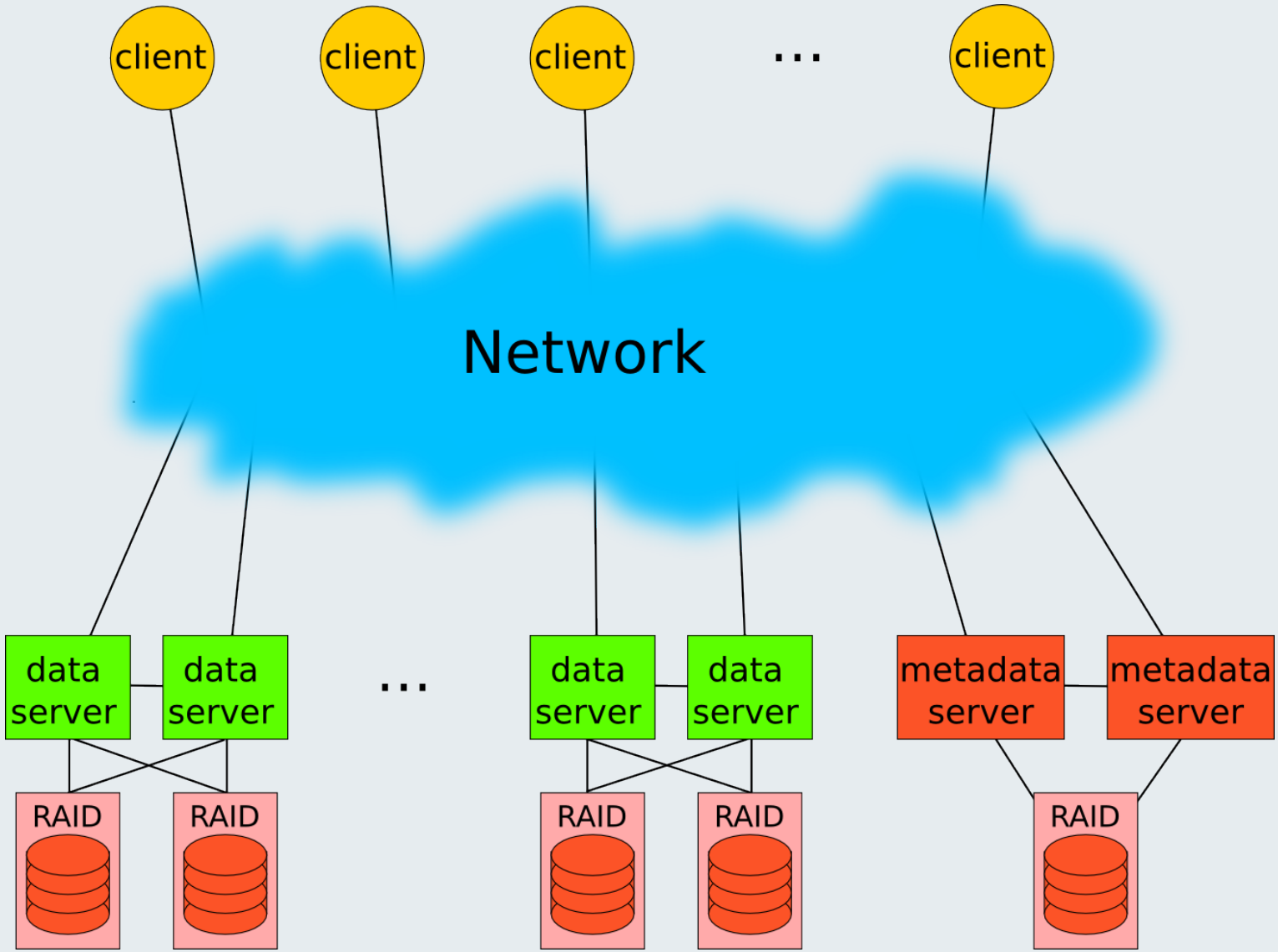
A parallel filesystem is designed to enable simultaneous accesses to a filesystem to multiple clients. The differences with a *simple* shared filesystem is the level of parallelism:

- Multiple clients can read and write simultaneously, not one at a time.
- The distribution of data. A client will get good performance if the data is spread across multiple data servers.

This parallelism is transparent to the client which sees the filesystem as if it was local.

In addition to the functions of a local filesystem, a parallel filesystem must efficiently manage potential conflicts between different clients. The preferred approach is to use locks to limit and control concurrent accesses to a given file or directory.

Typical architecture

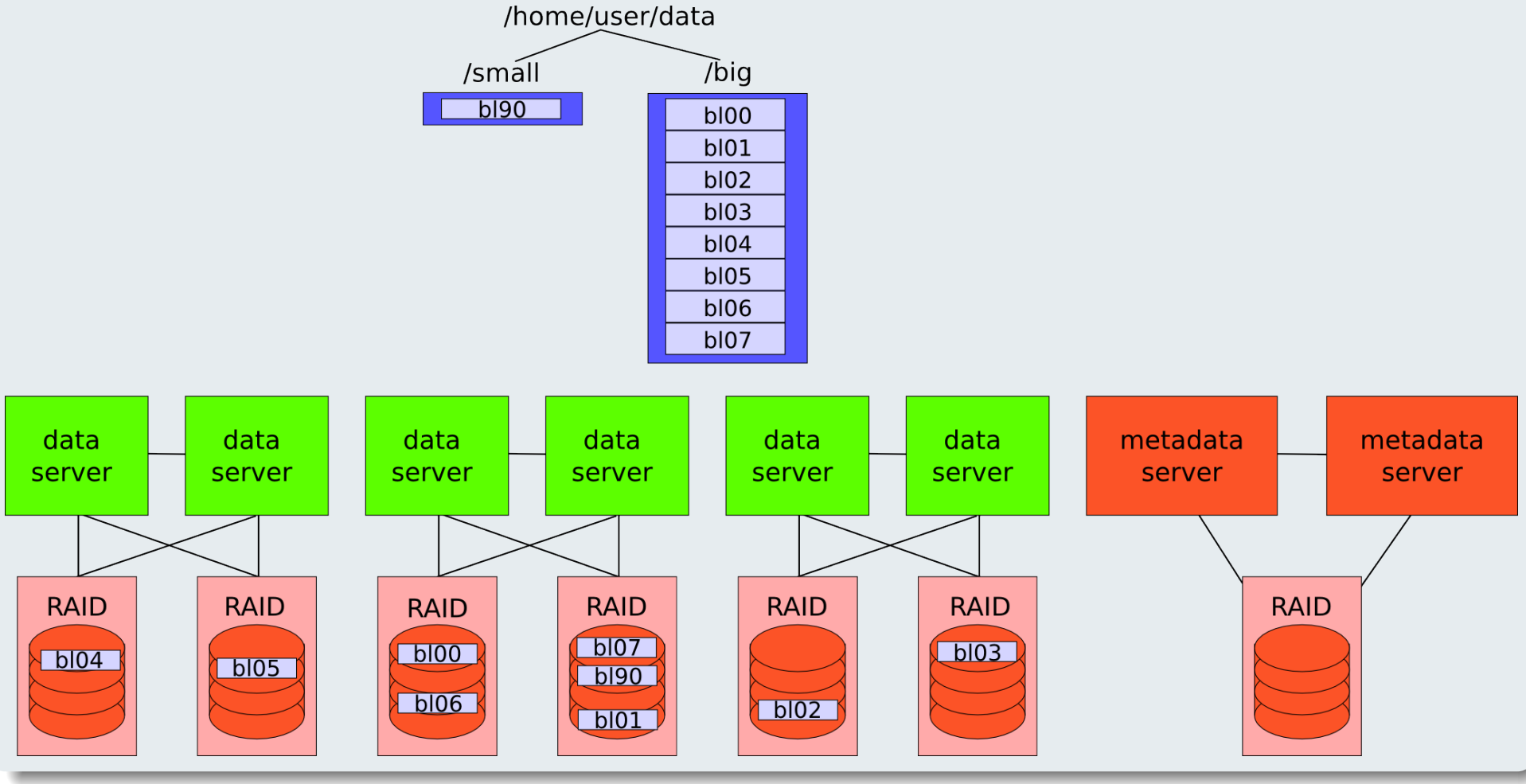


Typical architecture

A parallel filesystem is comprised of:

- clients which will read or write data to the filesystem;
- one or more metadata servers. They manage metadata and placement of data on the drives, as well as access control locks (for example to avoid that 2 clients modify the same part of a file simultaneously);
- a number of data servers. These store all the data. For some parallel filesystems, data and metadata can be handled by the same server;
- and one or more networks (dedicated or not) for interconnecting these components.

File striping



File striping

A file will usually be cut into pieces of fixed size (called stripes or chunks) and disseminated between different servers. A read or write of the file will therefore be done in parallel on different file servers. The speed of writing or reading will be the sum of the rates obtained on these servers.

Data integrity and redundancy

The parallel filesystem must also ensure data integrity and system redundancy. This can be done in several ways:

- Each data and metadata server manages several drives that use a local filesystem with RAID support ensuring data integrity in case of loss of one or more disks.
- Data can be replicated in several different places.
- A data or metadata server may be able to manage disks from another server and take over it in case of failure.
- An alternative pathway for the data may exist (two different networks, for example).

Locks and concurrency: purpose

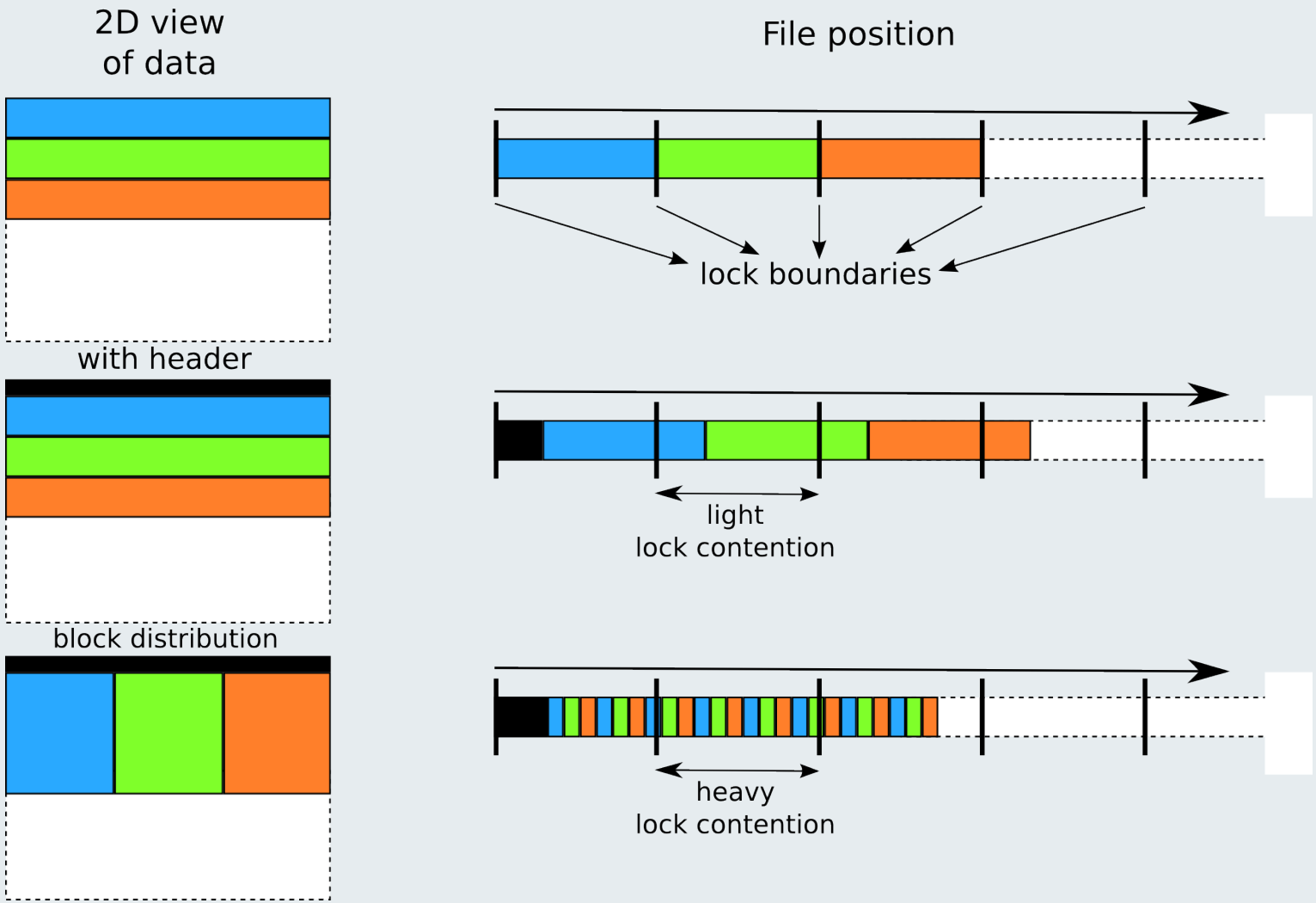
To ensure consistency of data and metadata, parallel filesystems usually use locks that limit concurrent access to this information. This allows, among others, to ensure the atomicity of read/write operations. For example, a process writes a block of data and one wants to read it at the same time. The use of a lock guarentee that the reader will read the data block as it was before or after the change (as it gets the lock before or after the writer), but never a mixture of both.

Locks and concurrency: working

Depending on the filesystem, the locks on the data are managed on a file-level or on a stripe-level basis. They are aligned to certain boundaries (eg size of memory pages for Lustre and size of the filesystem block for GPFS).
There are 2 main types of locks:

- Exclusive locks for writes limiting access to a range from a single client.
- Shared locks for read accesses to a range by any number of clients and preventing changes/concurrent writes.

Locks and concurrency: working



Data represent a two-dimensional array in the application (contiguous data along lines). Each color corresponds to a process/client.

Caches

A cache is a local copy close to the one that uses it. Its purpose is to accelerate performance. Their influence can be very important.

In a parallel filesystem, caches are mainly:

- Data servers side. Caches are prior the disks in random access memory (faster) and can be read and write (in this last case, the memory has to be powered by batteries in case of power failure);
- Clients side. Data consistency between different clients must be assured. This is done via locks. For example, a client which has write access will flush its caches to data servers if the corresponding lock is removed. Another case, if data is cached on some clients and another begins writing in the same part of the file, read caches will be invalidated (ie the data on them can no longer be used) before starting to read the newly written data from the data servers.

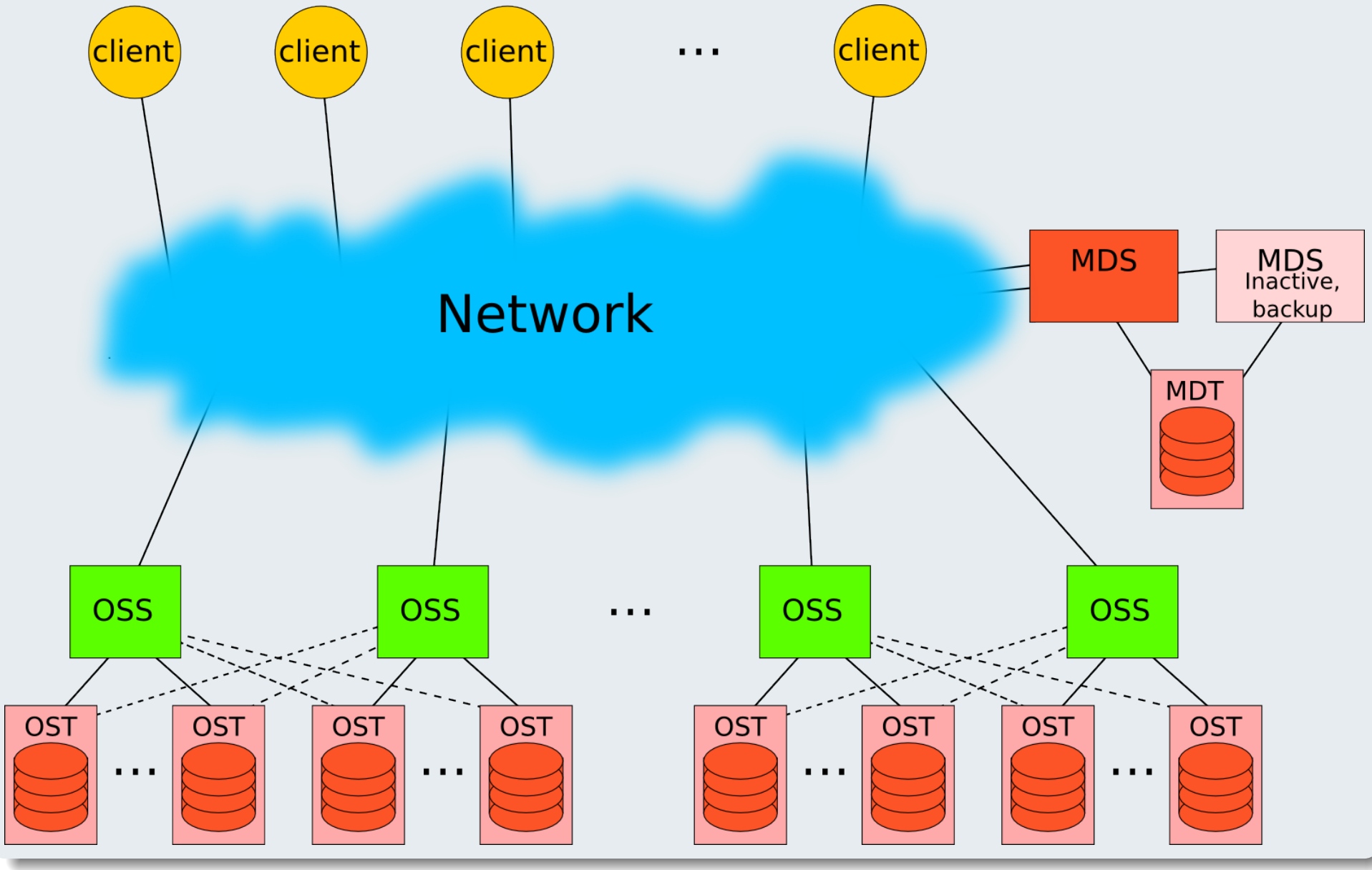
Each parallel filesystem has its own way of managing caches.

Main parallel filesystems

The most commonly used parallel filesystems in supercomputers are:

- Lustre
- GPFS
- PVFS2/OrangeFS
- PanFS

Diagram


client
client
client
...
client
Network
MDS
MDS
Inactive,
backup
MDT
OSS
OSS
...
OSS
OSS
OST
OST
OST
OST
OST
OST
OST
OST

Lustre architecture

Lustre is an open source parallel filesystem used by more than half of the Top 500 supercomputers (among others *Titan* (#2), *Sequoia* (#3) and *K computer* (#4)). It runs on all major networks (InfiniBand, Myrinet, Quadrics, TCP/IP...).

A Lustre system consists of:

- one metadata server (MDS) which manages a *Meta Data Target* (MDT) filesystem,
- possibly a backup metadata server that can take control in case of failure on the primary MDS,
- a series of data servers (called *Object Storage Servers*, OSS) that manage each several *Object Storage Targets* (OST),
- and clients.

MDT and OST use *ldiskfs* local filesystems (based on the ext3 filesystem) (ZFS for the machine *Sequoia*) and can use LVM and RAID.

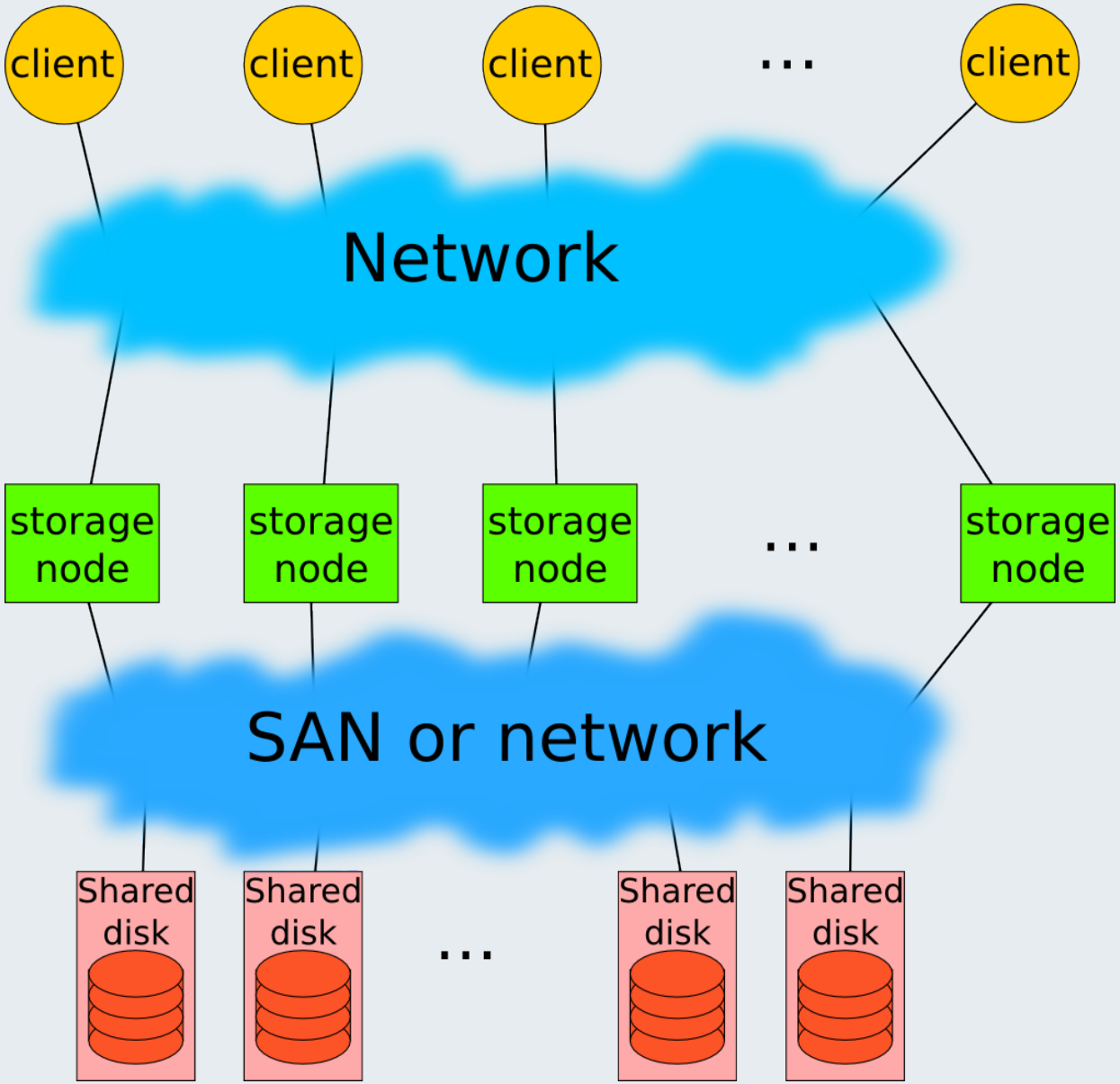
How Lustre works

When a client wants to access a file,

- it contacts the MDS which provides information on the OSTs that hold the data, or on which it will be able to write;
- the MDS changes metadata if necessary;
- then, the client communicates directly with the OSSs to read or write data.

The locks are set by intervals of bytes on each OST and are managed by the OSSs.

Diagram


client
client
client
...
client
Network
storage node
storage node
storage node
...
storage node
SAN or network
Shared disk
Shared disk
...
Shared disk
Shared disk

GPFS architecture

GPFS is a parallel filesystem developed by IBM. It is commercially licensed and used in many supercomputers (including *Mira*, the #5 of the Top 500 and at IDRIS).

A GPFS system consists of:

- a series of storage servers that deal with data and metadata (which can be separated or not),
- a series of shared disks (SAN-attached or network block devices) and accessible through any storage server,
- and clients.

Metadata is distributed on different storage servers with a single node responsible for the metadata of a given file.

Locks on a file (in bytes intervals) are, depending on circumstances, distributed among the various nodes or managed by a specific node.

Architecture and functioning of PVFS2/OrangeFS

PVFS2 and OrangeFS are open source parallel filesystems. They are very similar and vary only in the details. They run on the major networks (InfiniBand, Myrinet, Portals, TCP/IP...).

A PVFS2 or OrangeFS system consists of:

- one or several metadata servers,
- a series of data servers,
- and clients.

They have some special features:

- Optimized for MPI with support for derived datatypes;
- Designed without locks (lockless or stateless). This greatly simplifies things, but, for example, atomicity is not guaranteed if 2 clients write to the same location at the same time (the end results can be a mixture of the 2!);
- Do not follow the POSIX semantics.

As Lustre, when a client wants to access a file, it contacts a metadata server which provides information on the data servers to use and then the client communicates directly with these servers to read or write.

PanFS architecture

PanFS is a parallel filesystem developed by Panasas under commercial license. This is a turnkey solution coming with all the necessary hardware and software. *Cielo* #32 in the Top 500 uses it.

A PanFS system consists of:

- a series of metadata servers (*DirectorBlades*). Each metadata server manages only one volume corresponding to a filesystem directory. They are also involved in the management of the locks;
- a series of data servers (*StorageBlades*)
- and clients.

Data can be automatically migrated between data servers for load balancing.

Instead of using traditional RAID, PanFS uses Object RAID. Each file is divided into objects (like most other parallel filesystems). And these objects are protected individually by using the RAID-5 algorithm (user configurable). Parities sums are computed directly by the clients (which can verify the integrity of data during reading).