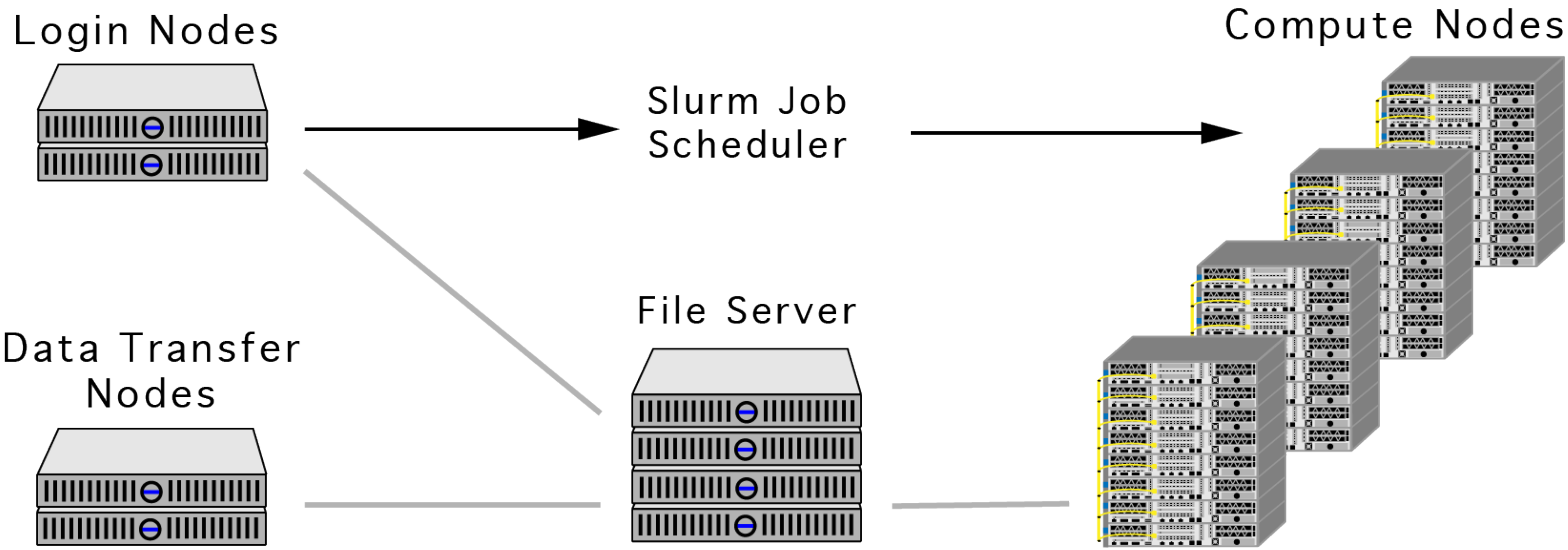
Lecture 8: Anatomy of High Performance Computing

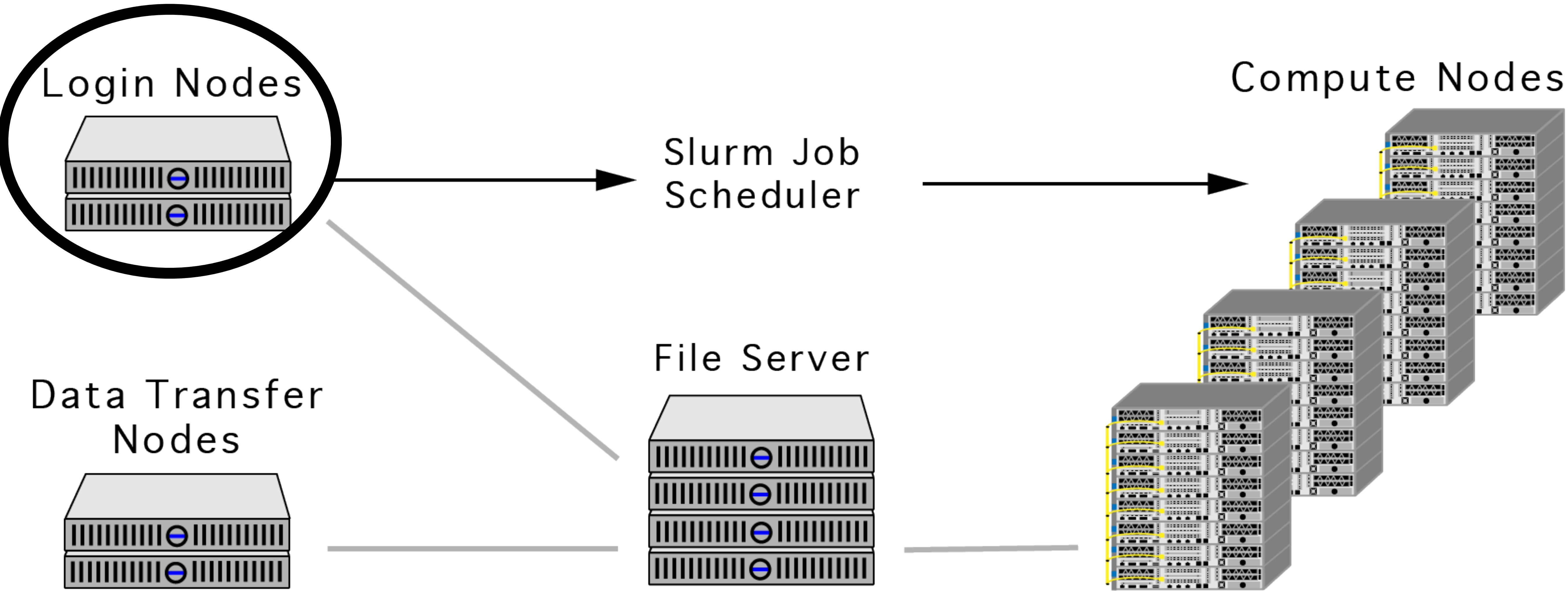
CEE 690

What these machines look like

How are they structured?



How are they structured?

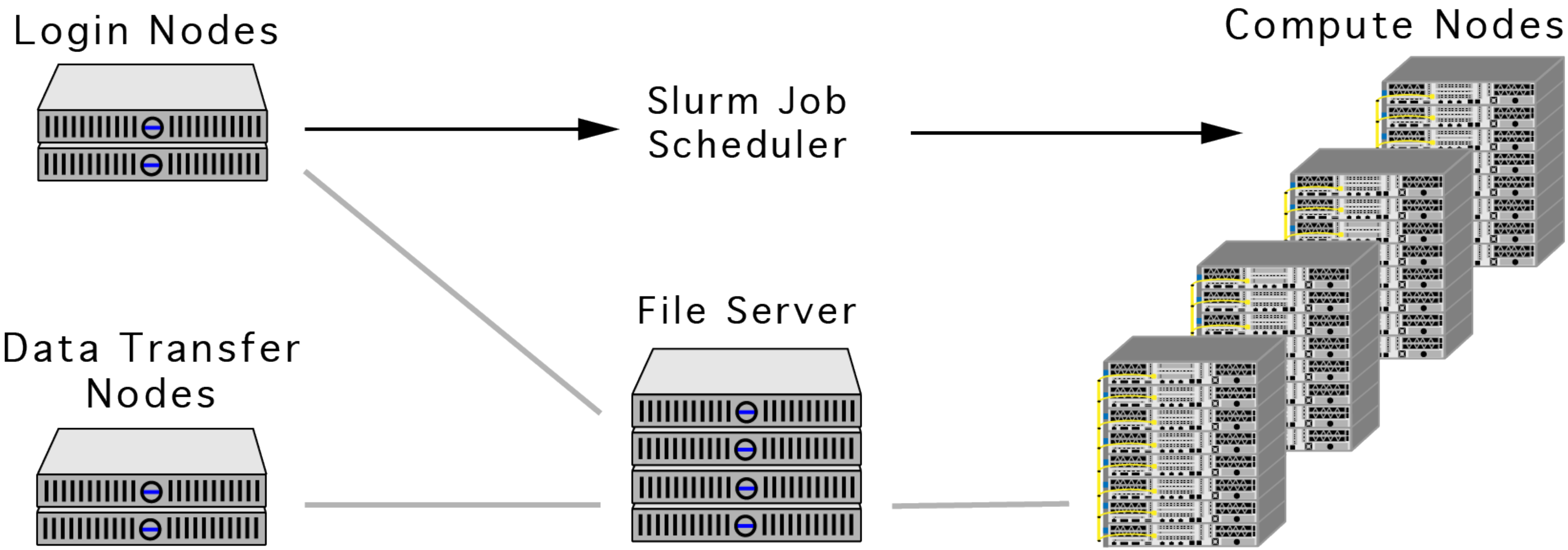


Login node

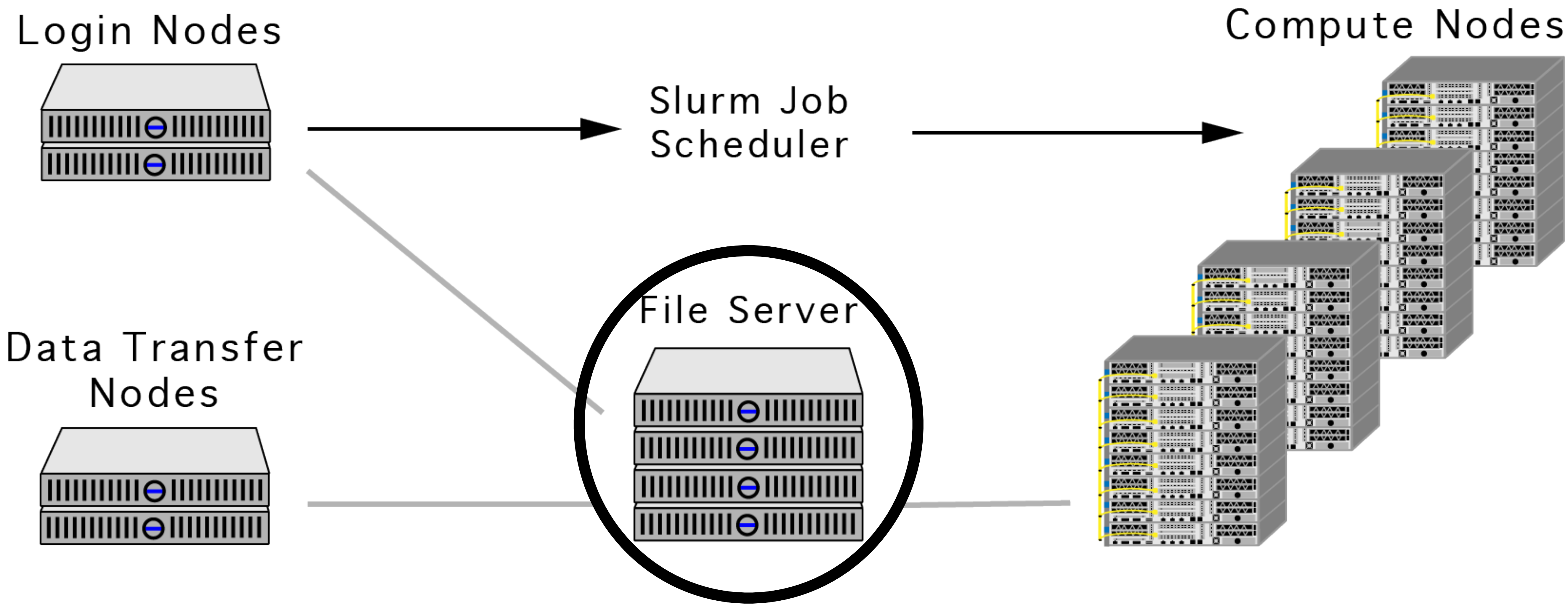
- **Purpose** - Shared node for editing code, managing files, compiling code*, and submitting jobs
- **Never run actual computation here**

This node is where you ssh into

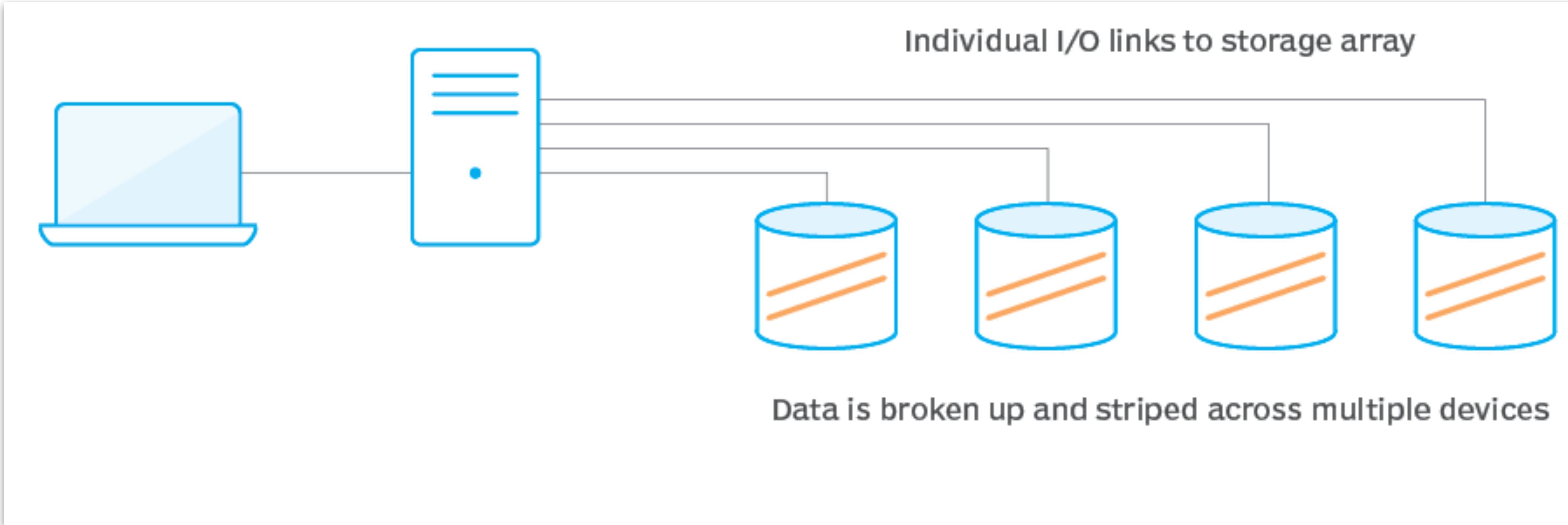
How are they structured?



How are they structured?



File server



- This is where your home directory*, data, and code live
- These systems are made to handle many people accessing the same data

Big hard drive???

Laptop - 1-4 TB

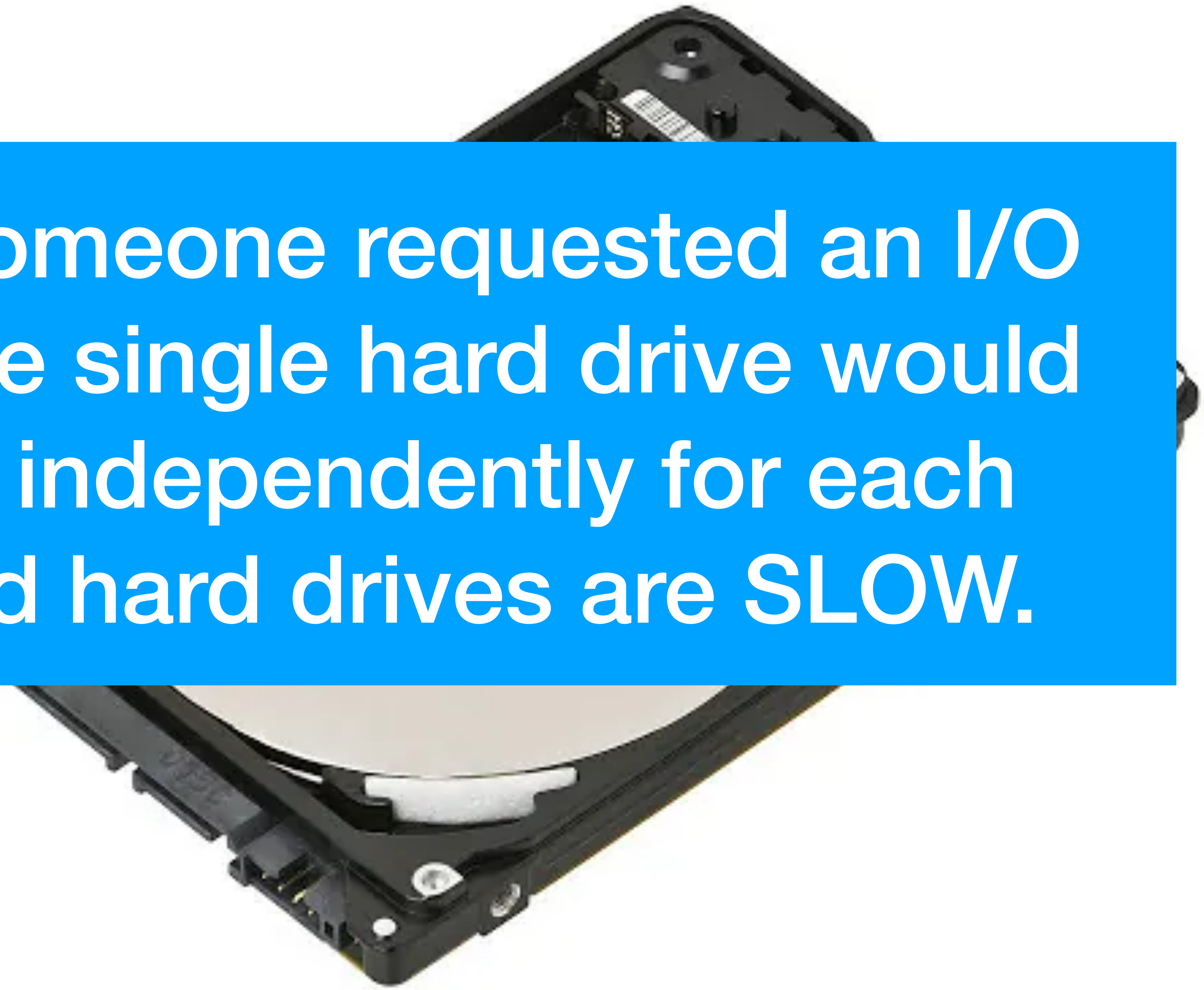
Big hard drive???

Laptop - 1-4 TB

So then why not just use a BIGGER hard drive?

HPC? - 400 TB!

Big hard drive???

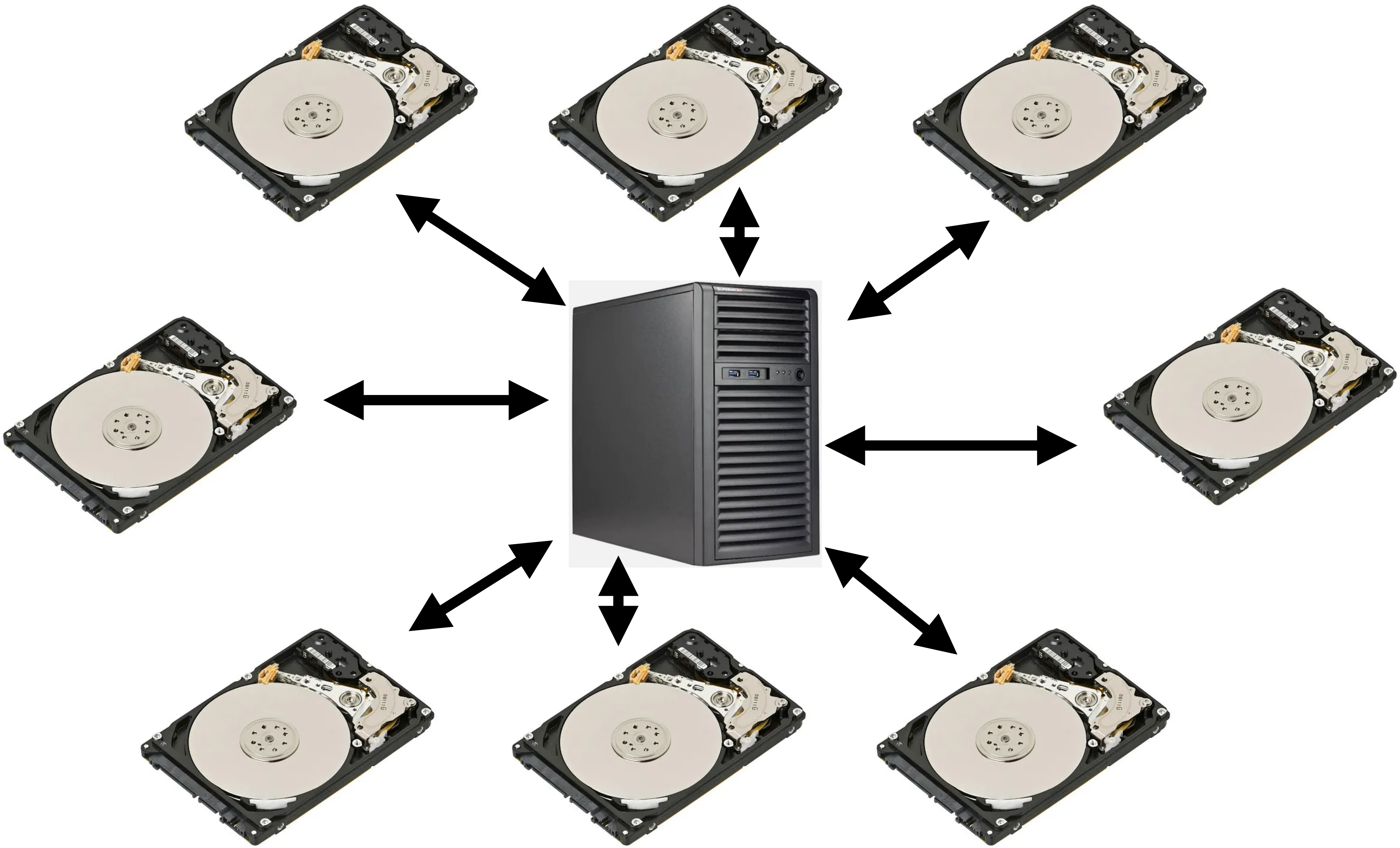


Every time someone requested an I/O operation, the single hard drive would have to act independently for each request. And hard drives are SLOW.

not just use a BIGGER hard drive?

HPC? - 400 TB!

Solution: Lots of hard drives and server to direct the orchestra



Strategy: Striping

- In a parallel file system, individual files are broken into small chunks and spread throughout the hard drives.
- You aren't limited by one hard drive; you can effectively speed up I/O by 10, 50, 100+ times

Metadata vs data

- **Metadata servers (MDS)** - These know the “structure” of the filesystem. When you query a directory, you are querying the metadata server.

- **Object storage servers (OSS)** - These are the actual workhorses. The MDS tells the compute node where your data is and then the compute node queries the OSS.

Avoid: Small file syndrome

- Give preference to fewer and larger files (not too large) over many very small files. Too many can hand the metadata server (I’ve been there many times…)
- **This is a key rationale for NetCDF, HDF5, Zarr.**

Permanent vs scratch

- HPC usually has some permanent storage that you can use while the rest is temporary (scratch).
- Learning to know when to use permanent storage (e.g., /home) vs scratch is key.
- Scratch does not usually have backups and is erased fairly quickly.

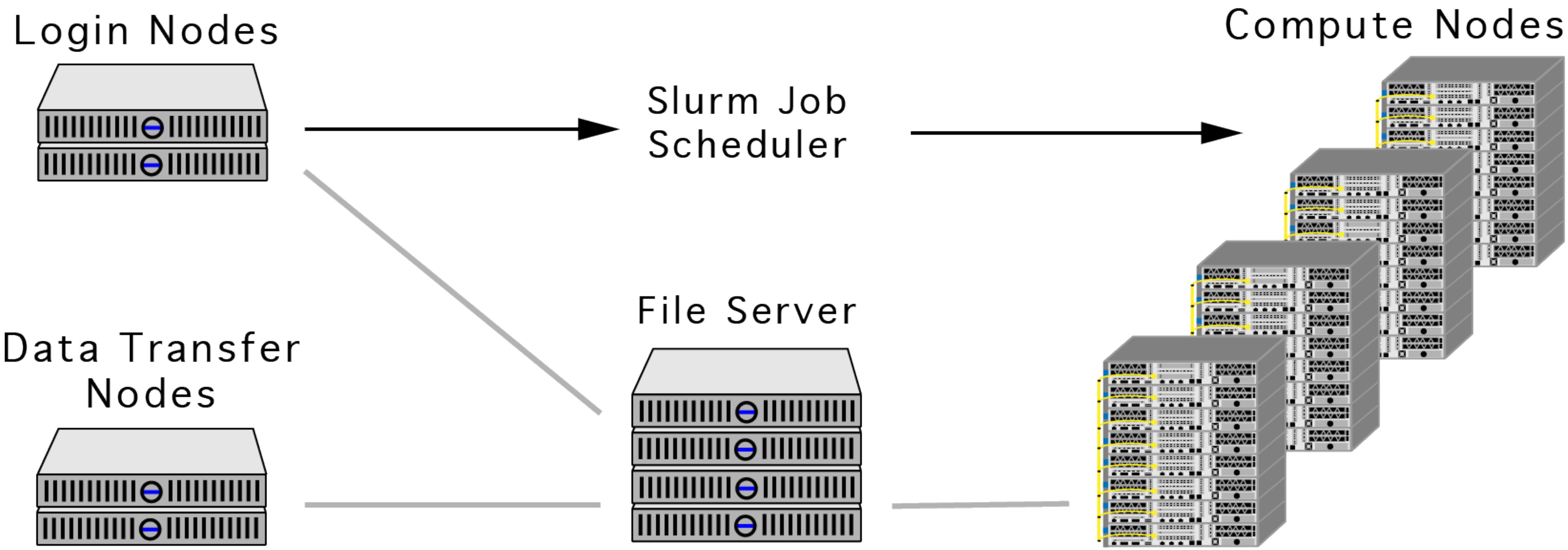
Parallel file system software

- **Lustre** - The most widely used in supercomputers.
- **BeeGFS** - Easier to install and use than Luster (good for small/medium clusters)
- **IBM Storage Scale (GPFS)** - Very mature and great for large systems.
- …

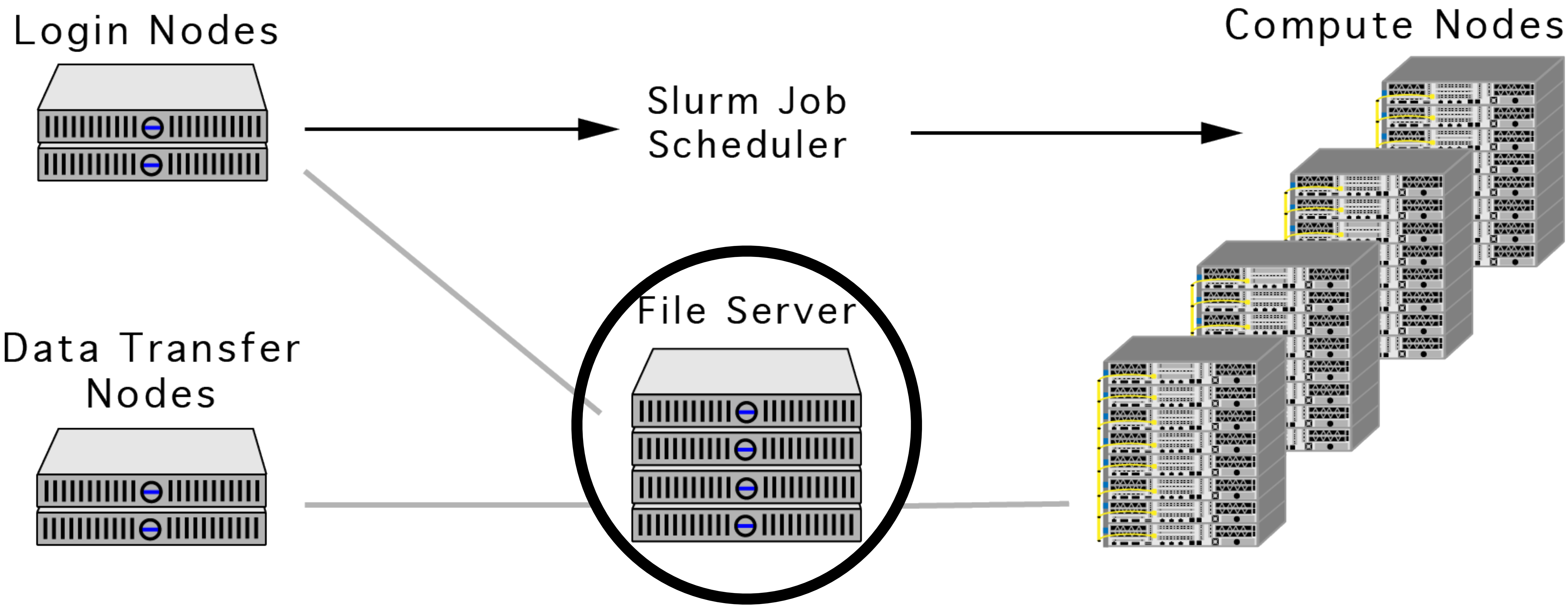
Examples of large file systems

System	Architecture	Capacity	Peak Throughput	Primary Use
Colossus	Google (Internal)	**10+ EB**	50+ TB/s	Global Web Services / AI
Orion	Lustre	**700 PB**	10 TB/s	Exascale Physics / Climate
DAOS (Aurora)	Object/Flash	**230 PB**	31 TB/s	AI & High-Speed Simulation
Alpine	GPFS	**250 PB**	2.5 TB/s	General Scientific HPC

How are they structured?

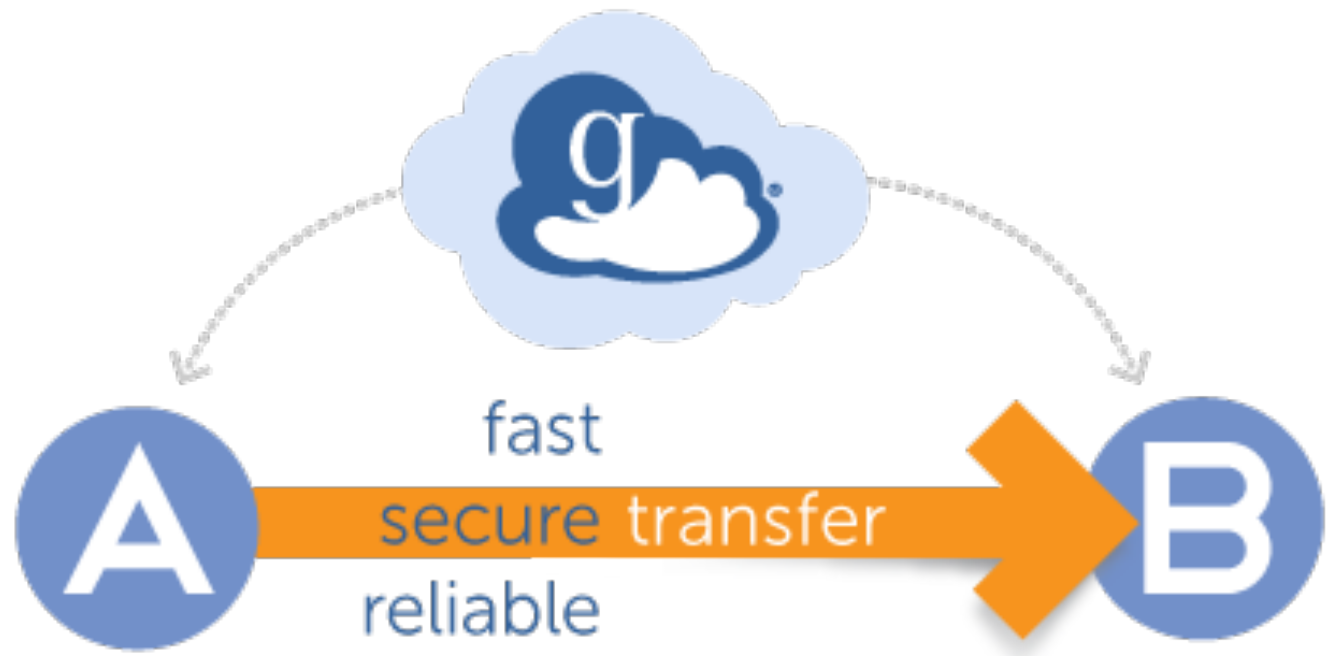


How are they structured?



Data transfer nodes (DTNs)

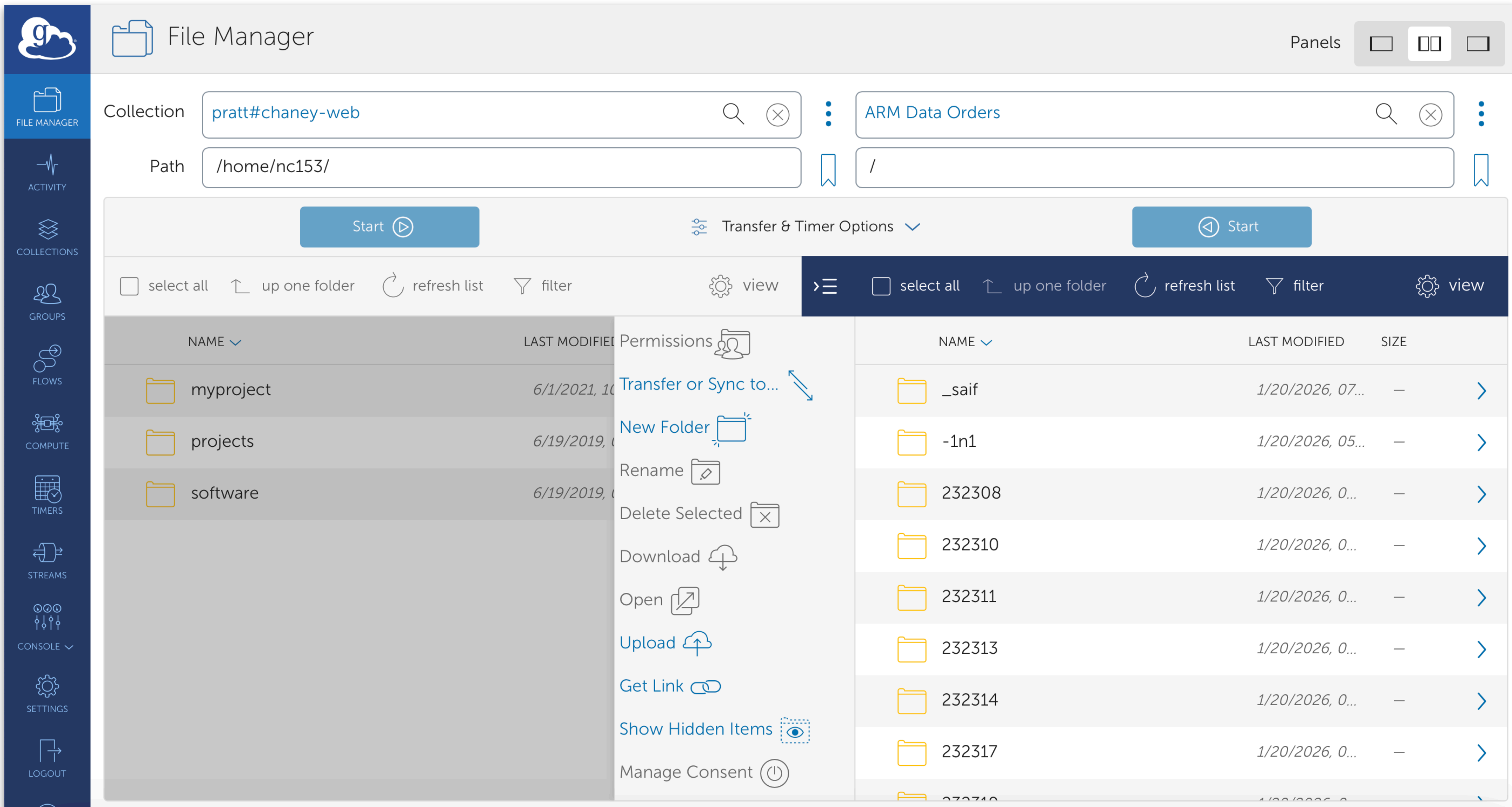
- Purpose is to move large amounts of data to/from external source
- DTNs are optimized for high-bandwidth, long-distance data movement.



Globus

- Nowadays it is the most common interface used on DTNs
- Connect any two endpoints (e.g., two HPC systems or your laptop and HPC)
- Why not SCP or RSYNC? - If the transfer stops, it can restart itself without losing much
- GridFTP - It breaks a file into multiple streams that are sent in parallel to/from the endpoints

Globus online interface



There is also a command line interface that you can use

To be continued..