# Filesystems
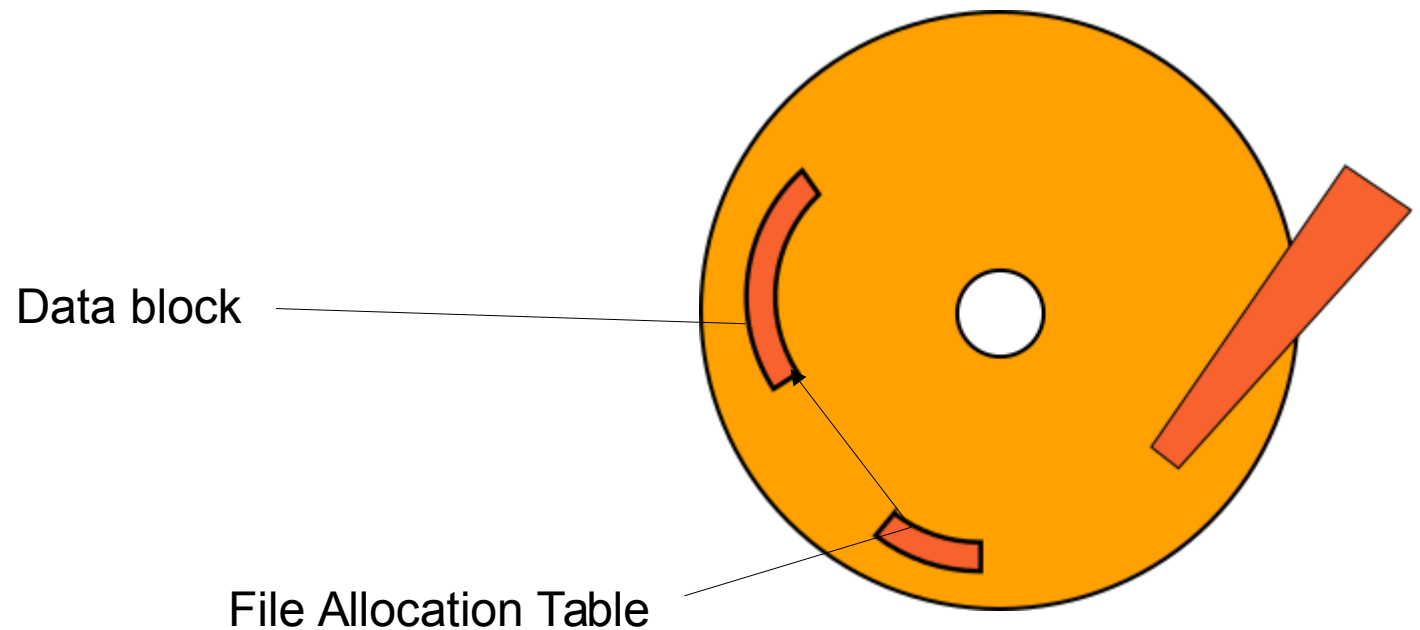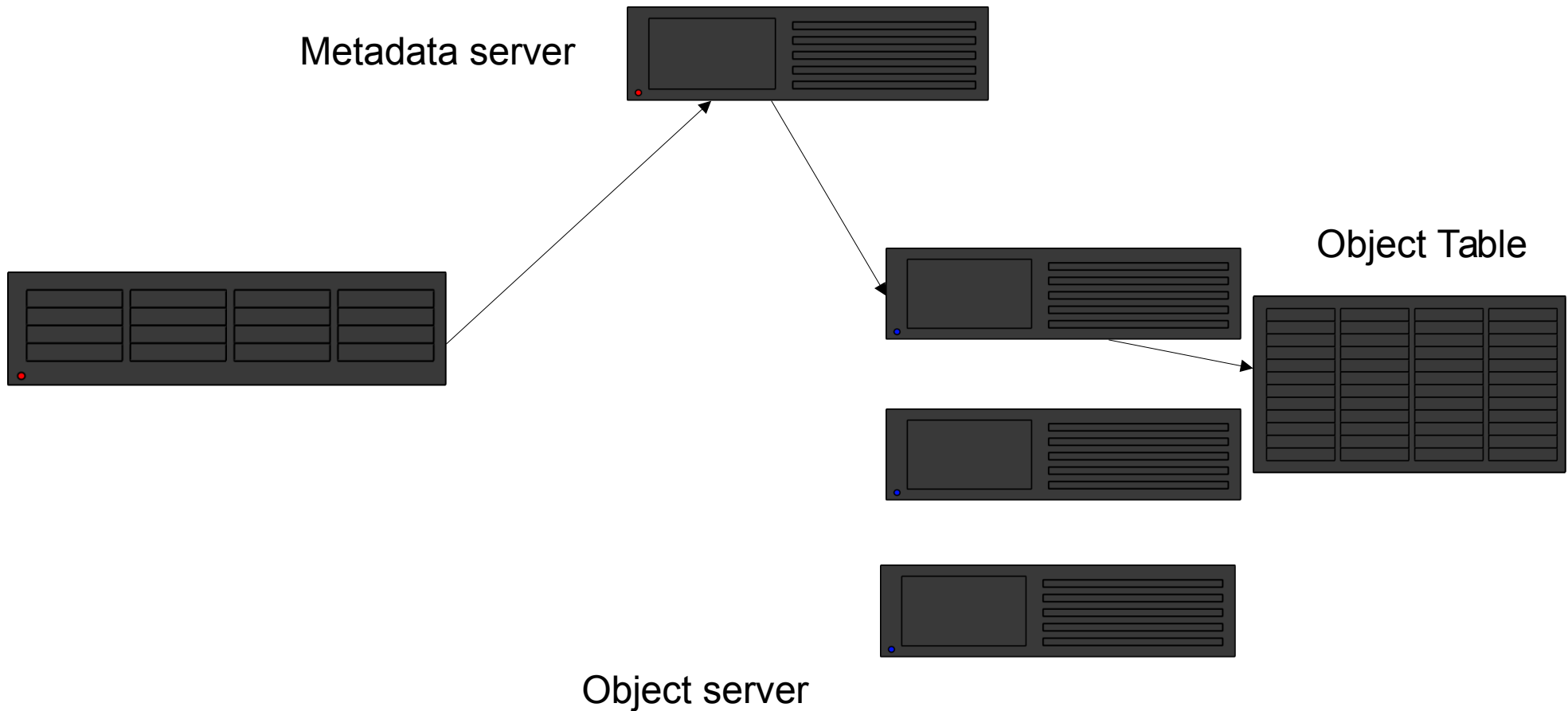
- Filesystems consist of two parts:
    - Metadata – where is my data
    - Content – the data itself



Data block

File Allocation Table

# Lustre

- Same basic concept
- Built to scale



Metadata server

Object Table

Object server

# NFS

- For comparison:

- 

- 



NFS server

- Metadata and object data in same place
  - Reading excessively prevents access
- No ability to spread load
  - No ability to server multiple clients efficiently

- 

- Reason for 'No datasets on /home' recommendation

# Lustre

- 6xOSS
- 6x6xOST
- 
- 1x MDS (+redundant)
  - Major bottleneck for distributed filesystem

# Lustre

- ## 100G in 1x file:
  - 1x access MDS
  - 1x access OST
    - Bottleneck is OST disk read speed (~500MB/s)

-

- ## 100G in 1000x files
  - 1000x access MDS
  - 1000x access OST
    - Bottleneck is MDS access rate (~300 iops/s)
      - Drastically affects other users!

# Small File Workarounds

- Try to avoid using small (<1Mb) files individually
  -
    - If you can't:
- If it's small (<32G), use shared memory
- If it's bigger (<350G), use /tmp

# Local Storage

Lustre

NFS

/dev/shm
FUSE

/ (boot image)
/tmp (350-500G)
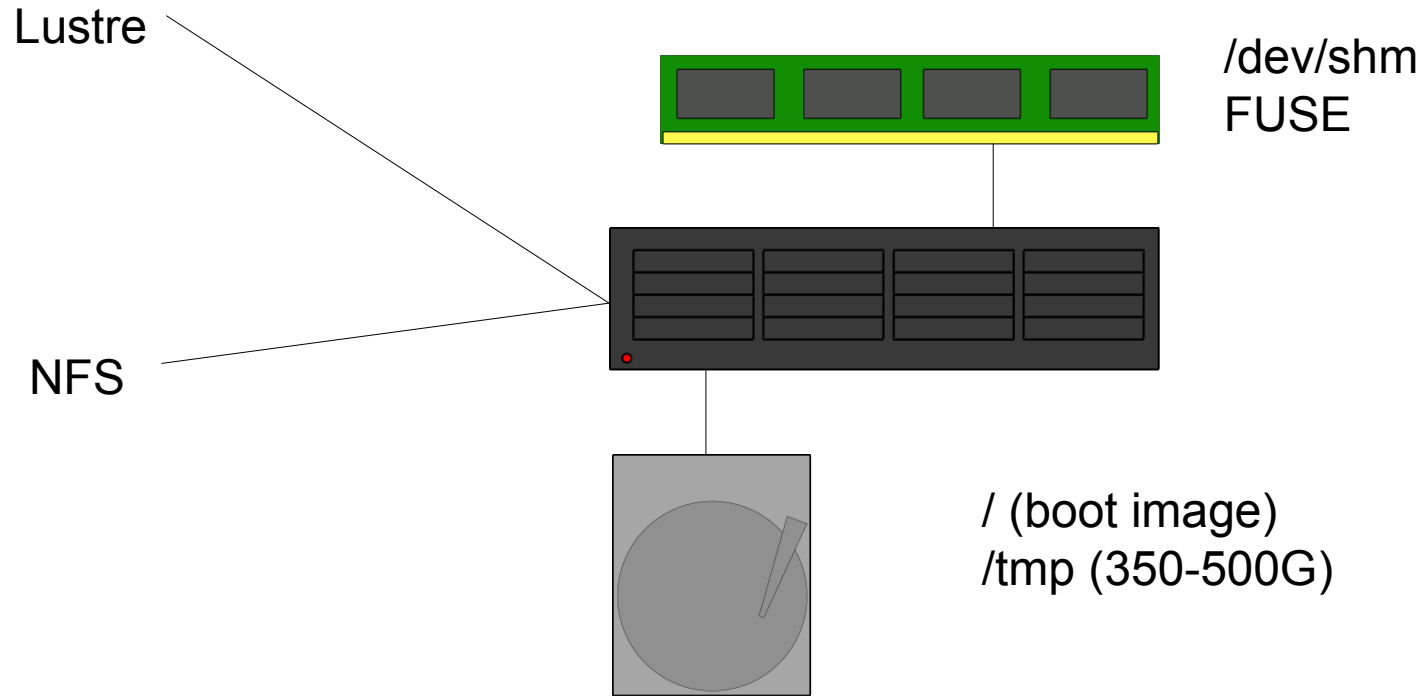
# Local Storage

- CAVEAT:
- If you use local storage
- 
-         PLEASE CLEAN IT UP
- 
- I can't know what your job specifically has written, especially if there's more jobs of your own running there
  - Thus there's no automatic way to remove local files
  - This INCLUDES /dev/shm!

# Local Storage

- /dev/shm means Shared Memory
  - Traditionally for transferring data between processes quickly
  - Can be abused for quick filesystem storage
  - 50% max ram size (32G/512G max capacity)
  -
  - Counts against memory usage for job
  - Typical IO ~1Gb/s

# Local Storage

- /tmp locally present on compute node
  - Small size disk – high RPM + high iops
- Nodes installed onto this disk (~20G)
  - Rest available for tmp
  -

- But – you still have to copy data to and from this location
  - If consists of small files – still problems!
    - tar + untar is your friend

# Large File Workarounds

- One file – one location – one disk
    - Bottleneck

Metadata server

Object Table

Object server
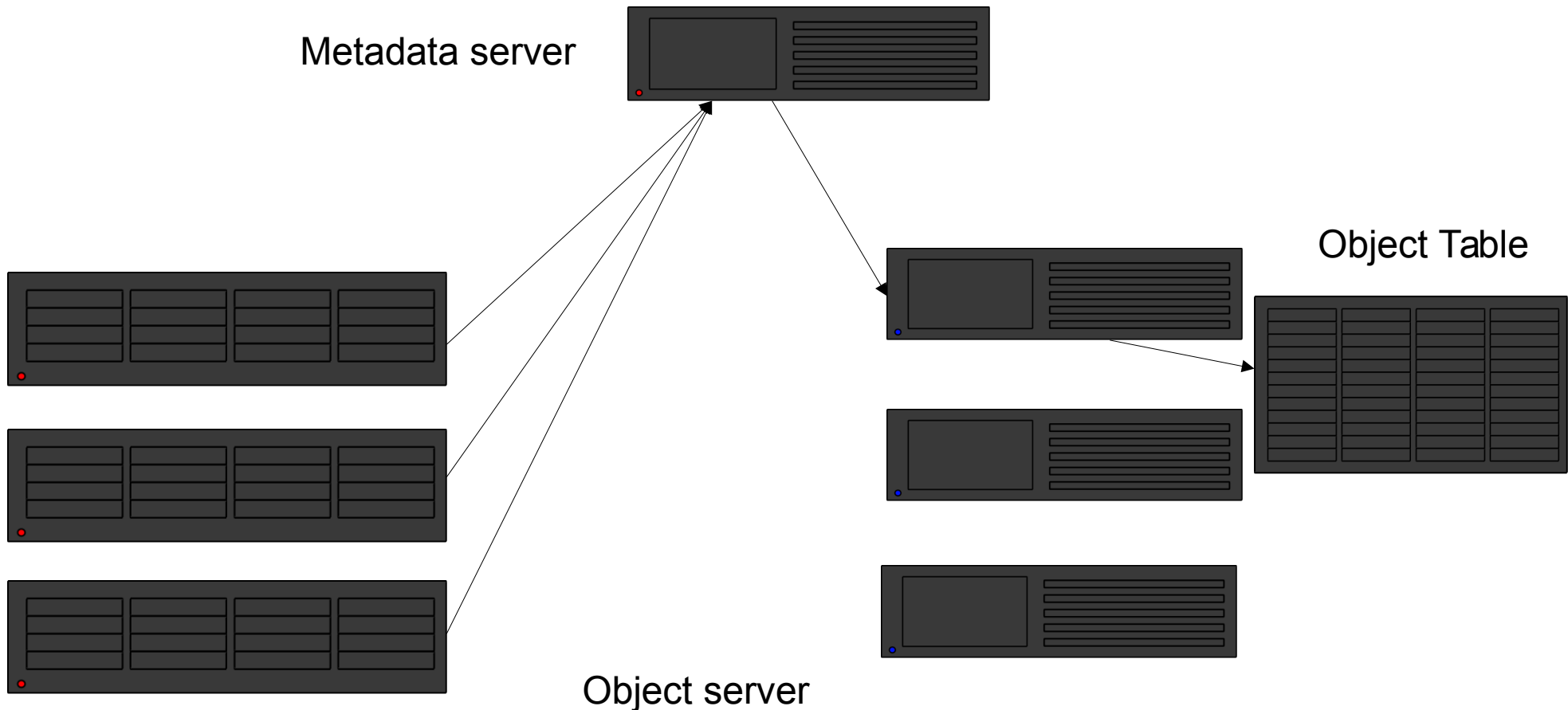
# Lustre striping

- Using lfs setstripe
- Set stripe size and count
- Spreads file over multiple OSTs
- 
- MUST BE pool = normalosts

```
dawes001@L0134766: ~
File  Edit  View  Search  Terminal  Help
-bash-4.2$ lfs setstripe -p normalosts -c -1 -S $((1024*1024)) striped_file
-bash-4.2$ dd if=/dev/urandom of=striped_file count=100 bs=1M
100+0 records in
100+0 records out
104857600 bytes (105 MB) copied, 9.99168 s, 10.5 MB/s
-bash-4.2$ lfs getstripe striped_file
striped_file
lmm_stripe_count:    36
lmm_stripe_size:     1048576
lmm_pattern:         1
lmm_layout_gen:      0
lmm_stripe_offset:   12
lmm_pool:            normalosts
     obdidx           objid            objid           group
         12        50086365       0x2fc41dd               0
         32        48926968       0x2ea90f8               0
         34        51866669       0x3176c2d               0
         19        48653588       0x2e66514               0
          6        47527451       0x2d5361b               0
         26        49695813       0x2f64c45               0
         35        48306347       0x2e118ab               0
         15        47376379       0x2d2e7fb               0
          2        47776151       0x2d90197               0
         11        47625562       0x2d6b55a               0
         14        46751053       0x2c95d4d               0
          4        52353914       0x31edb7a               0
          8        51172500       0x30cd494               0
         29        39334253       0x258316d               0
          9        50231833       0x2fe7a19               0
         30        50669726       0x305289e               0
          7        46047589       0x2bea165               0
         18        46581747       0x2c6c7f3               0
         21        49561994       0x2f4418a               0
          0        48112706       0x2de2442               0
         31        49924850       0x2f9caf2               0
         33        38360434       0x2495572               0
         13        51352170       0x30f926a               0
         20        49089063       0x2ed0a27               0
         28        41945275       0x28008bb               0
         23        47492004       0x2d4aba4               0
         16        45745846       0x2ba06b6               0
         27        42089200       0x2823af0               0
         22        45202256       0x2b1bb50               0
         25        49002279       0x2ebb727               0
         17        43806772       0x29c7034               0
          3        49495471       0x2f33daf               0
         10        48776495       0x2e8452f               0
         24        47735116       0x2d8614c               0
          5        49202209       0x2eec421               0
          1        46308998       0x2c29e86               0
-bash-4.2$
```

# Large File Workarounds

- No longer bottlenecked on multiple section reads

Metadata server

Object Table

Object server
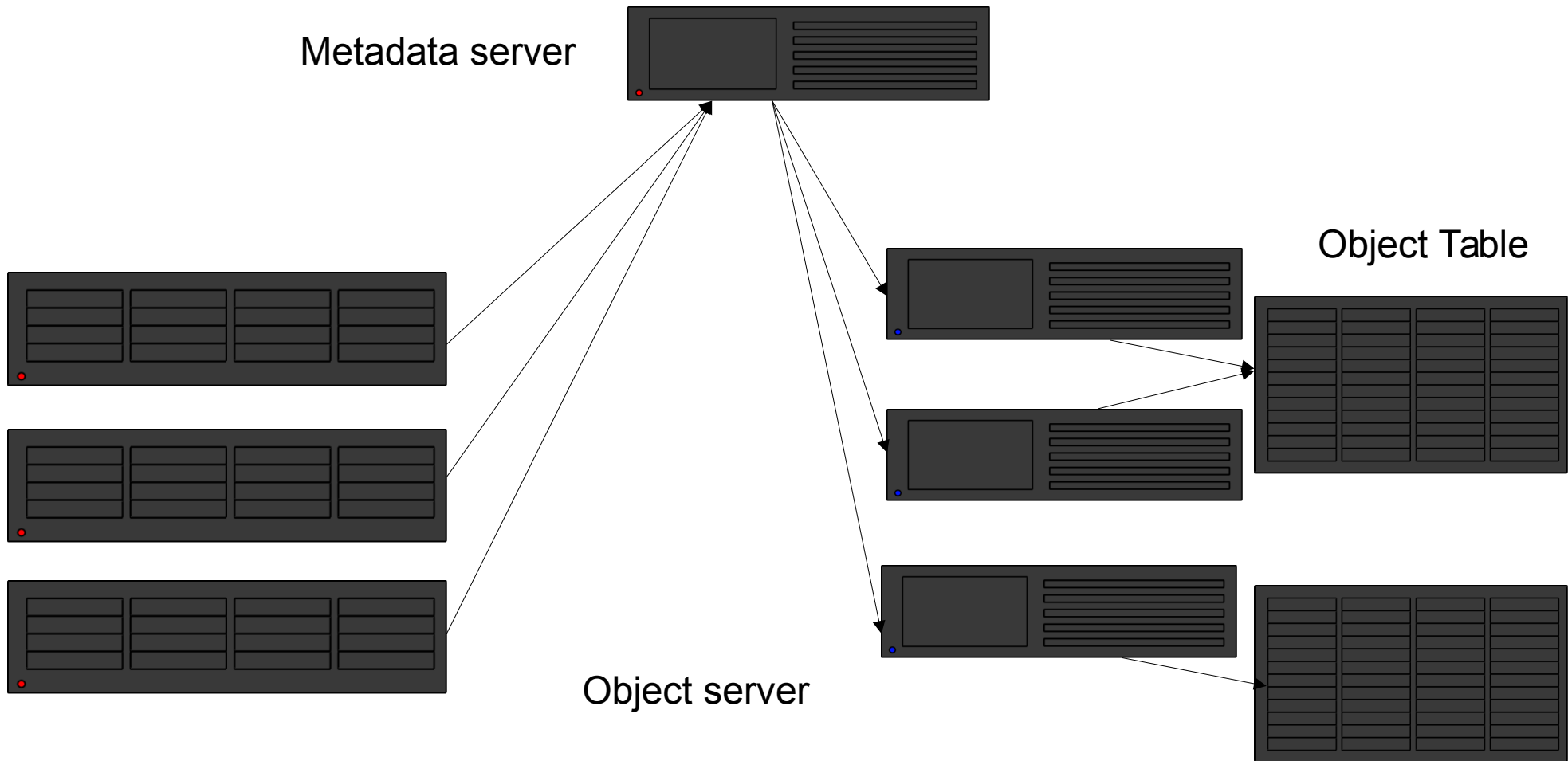
# Other Filesystems

- /archive on nfs01 – data on ISILON
  - WUR only
- 

- FUSE:
  - sshfs – mounts to remote fileservers
  - archivemount – technical curiosity only
    - Performance v. poor

# Scontrol

# scontrol details

- Priority
  - Scheduling priority given to job based on information in sprio
- JobState=PENDING
- Reason=PartitionTimeLimit
  - Descriptive reason why job isn't starting

# scontrol details

- SubmitTime/EligibleTime/StartTime/EndTime
  - (Start – Eligible) Rough queue length based on what Slurm expects jobs to take
  - Reason why job lengths are important
- NumNodes=1 NumCPUs=2 NumTasks=2 CPUs/Task=1
- TRES (Trackable Resources)
  - Check for what resources you've specified

# scontrol update?

- Basically no – you can't change the requirements of a job after it's running
    - Except for TimeLimit – you may always **reduce** this
- 
- But you can reduce the MinCPUNode/MinMemNode fields whilst job is pending

# sbatch Options

- Unusual options you might not know…
- 
- 
-         --dependency

# sbatch Dependencies

- after:job_id[:jobid...]
  - This job can begin execution after the specified jobs have **begun** execution.
- afterany:job_id[:jobid...]
  - This job can begin execution after the specified jobs have **ended**.
- afternotok:job_id[:jobid...]
  - This job can begin execution after the specified jobs have **terminated in some failed state** (non-zero exit code, node failure, timed out, etc).
- afterok:job_id[:jobid...]
  - This job can begin execution after the specified jobs have **successfully executed** (ran to completion

# sbatch Dependencies

- This allows you to submit multiple jobs in a chain
    - Not all the same size too, e.g.
    - small linear job to download/unpack (e.g. on normalmem)
    - Large assembly job (e.g. on fat)
    - Small packing job (e.g. on normalmem)

# sbatch Dependencies

- aftercorr:job_id[:jobid…]
  - A task of this job array can begin execution after the corresponding task ID in the specified job has completed successfully (ran to completion with an exit code of zero).

- This allows you to build chained array jobs

# sbatch Dependencies

- expand:job_id
  - Resources allocated to this job should be used to expand the specified job. The job to expand must share the same QOS (Quality of Service) and partition. Gang scheduling of resources in the partition is also not supported.
- singleton
  - This job can begin execution after any previously launched jobs sharing the same job name and user have terminated.

# sbatch Dependencies

- Singleton can be used to limit job rate
  - Name all in one 'pool' of jobs the same job-name
  - Only one will be executed at a time
- Don't get excited about expand!
  - Can only add additional nodes to jobs
  - scontrol update jobid NumNodes=ALL

# sbatch Options

- Unusual options you might not know…
- 
- 
-         --deadline

# Deadlines

- You can opt to have a job fail if it will never get to finish before a certain time
- Can also be a good safety switch for massive job submission

# sbatch Options

- Unusual options you might not know…
- 
- 
-                    --tmp

# Temporary Space

- You're going to use /tmp for something
- You need X Mb of space
  -
  - --tmp=X
  -

- Will not execute job on node with less than X available space
- Reduces heartache from other lazy users

# sbatch Options

- Unusual options you might not know…
- 
- 
-         --export

# Environment Settings

- You are submitting jobs from a script and want to pass in some environment variable:

- 

- sbatch –export="MYVAR=3"

- 

- You want to explicitly prevent your environment from tainting this job:

- 

- sbatch --export=NONE

# sbatch Options

- Unusual options you might not know…
- 
- 
-           --open-mode

# Append/Truncate

- #SBATCH –open-mode=append
- 
- Will append to existing output/error files rather than overwriting them
- Great for extending jobs / repeating jobs

# sbatch Options

- Unusual options you might not know…
- 
- 
-         --gres

# Generic Resources

- Not so generic
- Mainly used for additional hardware plugins – Graphical Processing Units (GPUs) and Many Integrated Cores (MICs, e.g. Knights Landing)
  - 
  - This is how you (could) specify GPU's if/when requested:
  - 
- #SBATCH –gres=gpu:1

# sbatch Options

- Unusual options you might not know…
- 
- 
-            --signal

# Signalling

- Slurm will send out signals to processes at a controlled time period before termination
    -
    - --signal=INT@120
    - Sends out a SIGINT (Interrupt) 120 seconds before job period expires
- Also can be done from scancel:
    -
    - scancel –signal USR1
    - Useful for sending signals in to get jobs to do things

# sbatch Options

- Unusual options you might not know…
- 
- 
-         --constraint

# Features

- Nodes are not uniform:
  - Normal nodes:
    - Intel CPUs
    - 4000M/CPU
  - Fat nodes:
    - AMD CPUs
    - 16000M/CPU
- May well be others besides in the future

# scontrol Features

- scontrol show nodes

# Features

- Can be combined:
- 
  - "opteron&video"
    - AND
  - "fast|faster"
    - OR
  - [rack1|rack2|rack3|rack4]
    - EVERY part of this job must be in one rack

# Reservations

- Some point in the future you need resources immediately
  -
  - e.g. a course
  - A seminar
  - Time critical computation
  -
- You can submit a job in advance, but you have to specify the result of that
  - How to proceed?

# scontrol Reservations



```
dawes001@L0134766: ~

File  Edit  View  Search  Terminal  Help
-bash-4.2$ scontrol show reservations
ReservationName=Major Downtime Nov 2017 StartTime=2017-11-23T08:00:00 EndTime=2017-11-23T20:00:00 Duration=12:00:00
   Nodes=fat[001-002],node[001-042,049-054] NodeCnt=50 CoreCnt=896 Features=(null) PartitionName=(null) Flags=MAINT,IGNORE_JOBS,SPEC_NODES
   TRES=cpu=896
   Users=root Accounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/a

ReservationName=GPUTEST StartTime=2017-08-30T15:27:19 EndTime=2017-11-01T00:00:00 Duration=62-09:32:41
   Nodes=gpu001 NodeCnt=1 CoreCnt=16 Features=nvidia PartitionName=(null) Flags=IGNORE_JOBS
   TRES=cpu=16
   Users=katzi001,vande018,verho068,moral005,warri004,knape001,dawes001,lith010 Accounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/
a

ReservationName=CANU_roden015 StartTime=2017-09-28T13:17:31 EndTime=2017-10-27T00:00:00 Duration=28-10:42:29
   Nodes=node[004-006,008,016] NodeCnt=5 CoreCnt=80 Features=(null) PartitionName=(null) Flags=IGNORE_JOBS,SPEC_NODES
   TRES=cpu=80
   Users=roden015 Accounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/a

ReservationName=HG_FREE StartTime=2017-10-10T10:48:36 EndTime=2018-01-01T00:00:00 Duration=82-14:11:24
   Nodes=node001 NodeCnt=1 CoreCnt=16 Features=normalmem PartitionName=(null) Flags=OVERLAP
   TRES=cpu=16
   Users=dings01,huisma01,vereij01,peeter01,ytourn01,willem01,bronsv01,zwiers01,bink01,visser01,blonk01,vila01,weteri01,ehlers01,faure01,rome01,fablet01 A
ccounts=(null) Licenses=(null) State=ACTIVE BurstBuffer=(null) Watts=n/a

ReservationName=HG_5 StartTime=2017-11-20T08:00:00 EndTime=2017-11-22T23:59:59 Duration=2-15:59:59
   Nodes=node002 NodeCnt=1 CoreCnt=16 Features=normalmem PartitionName=(null) Flags=
   TRES=cpu=16
   Users=dings01,huisma01,vereij01,peeter01,ytourn01,willem01,bronsv01,zwiers01,bink01,visser01,blonk01,vila01,weteri01,ehlers01,faure01,rome01,fablet01 A
ccounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/a

ReservationName=HG_6 StartTime=2017-12-11T08:00:00 EndTime=2017-12-13T23:59:59 Duration=2-15:59:59
   Nodes=node002 NodeCnt=1 CoreCnt=16 Features=normalmem PartitionName=(null) Flags=
   TRES=cpu=16
   Users=dings01,huisma01,vereij01,peeter01,ytourn01,willem01,bronsv01,zwiers01,bink01,visser01,blonk01,vila01,weteri01,ehlers01,faure01,rome01,fablet01 A
ccounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/a

ReservationName=HPC_ADVANCED_COURSE StartTime=2017-11-09T08:00:00 EndTime=2017-11-09T13:00:00 Duration=05:00:00
   Nodes=node[002-004] NodeCnt=3 CoreCnt=48 Features=normalmem PartitionName=GUESTS_Low Flags=
   TRES=cpu=48
   Users=-root Accounts=(null) Licenses=(null) State=INACTIVE BurstBuffer=(null) Watts=n/a

-bash-4.2$
```

# Reservations

- Need to be added by admin
- Can only be assigned to users, not groups
  - Can be hacked to follow groups – contingent on admin awareness
- Can only allocate entire nodes
  - Can allocate CPU's, but no memory – basically useless
- General policy – max 3 nodes