Running an EOS instance with tape on the back

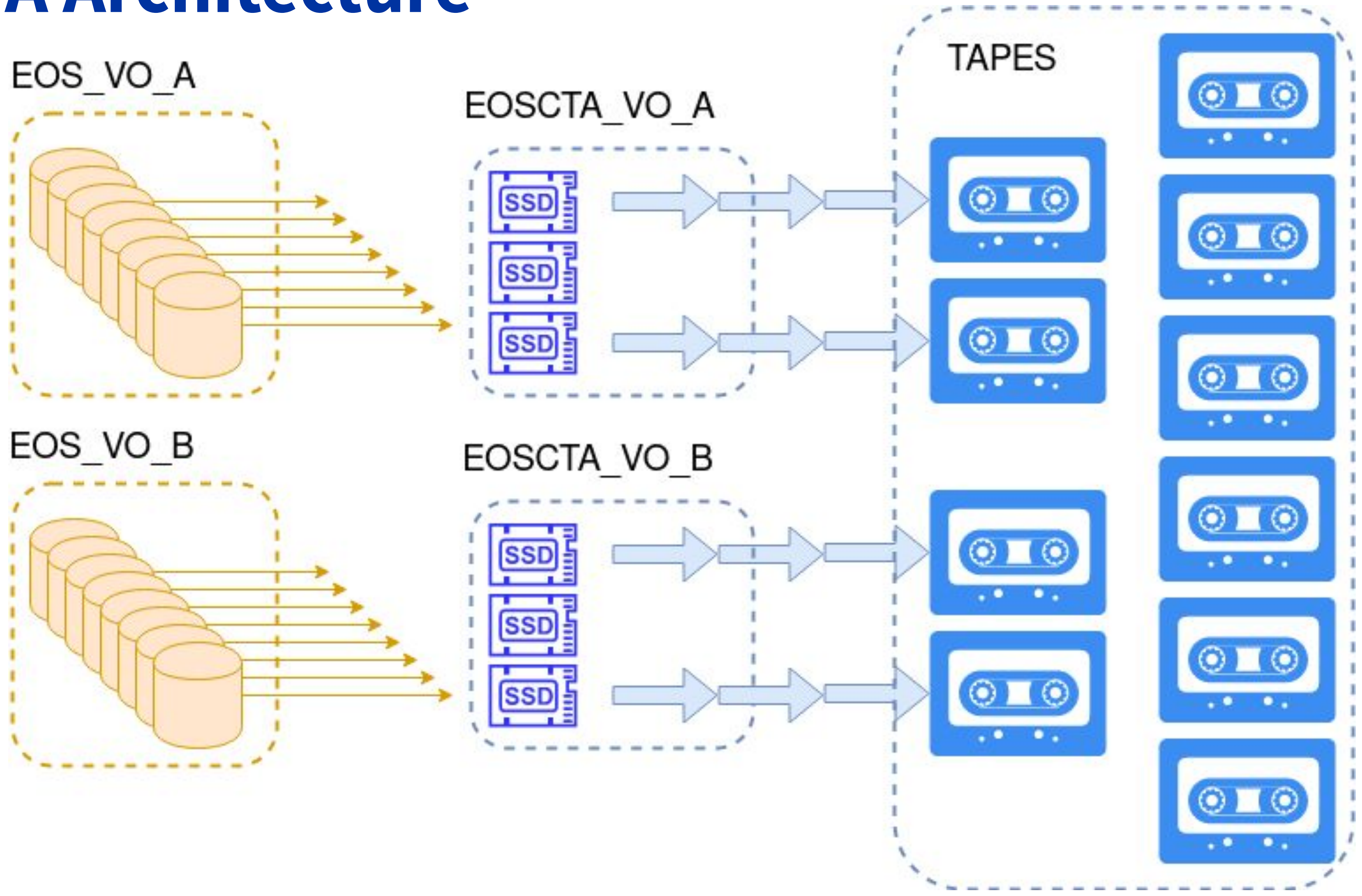
Julien Leduc

8/3/22 - EOS workshop

EOS+CTA Architecture

- **EOS+CTA is a pure tape system.**
- **Disk cache duty consolidated in main EOS instance.**
- **Operating tape drive at full speed full time efficiently requires a SSD based buffer: EOSCTA**

EOS+CTA Architecture



EOS instance VS EOSCTA instance characteristics

EOS DISK instance	EOSCTA instance
Transfer data to and from disks	Transfer data to and from tapes
Capacity oriented	Bandwidth oriented
Bandwidth as a by-product	0B capacity (tip: ∞ available on tapes)
Keep data safe for long	Only store transient data
Resilient to storage building block failure (HDD / FST server / full rack)	Efficient and early failure notification for retries

EOS+CTA Practical configuration

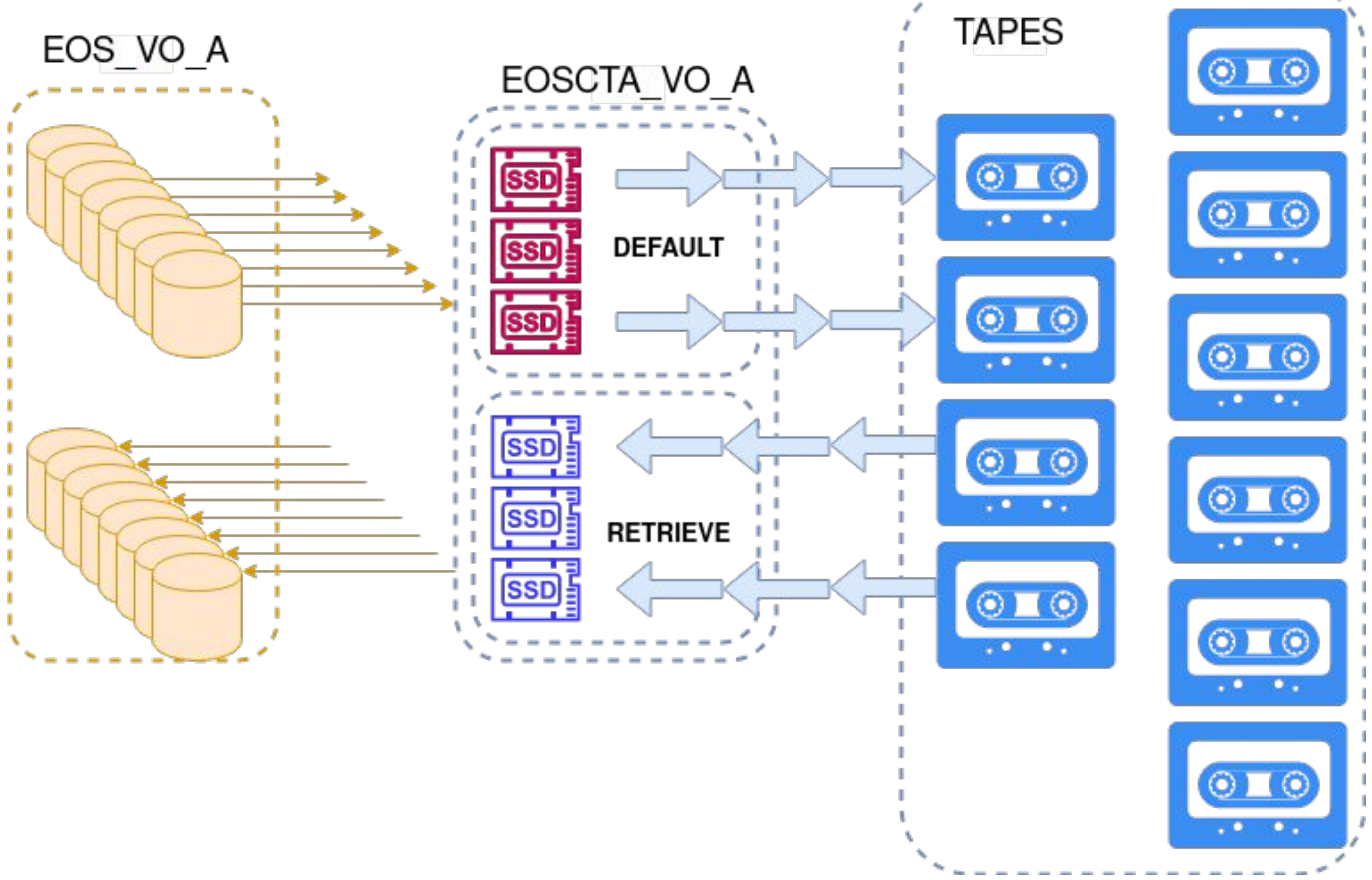
- Give EOSCTA instance *tape flavour*
  - `mgmofs.tapeenabled  true` in mgm config
- file replica transition are triggered by EOS WFE
  - `space.wfe=on` and `space.wfe.ntx=500`
- SSD spaces are read and written at the same time must survive full duplex network card speeds
  - 1 replica layout on SSD spaces
  - CERN eoscta production machines: 16 data SSDs for 3GB/s full duplex network bandwidth (at least 400MB/s per SSD)
    - no spare storage bandwidth for data scanning: disable it
      - `space.scanrate=0` and nail it with `space.scaninterval`
    - no spare network bandwidth for more than 1 replica
- instruct FTS to evict files successfully transferred out of eoscta instance (*xrootd only*)
  - `all.sitename cern_tape_archive_eoscta_VO_A`

Anatomy of an EOSCTA instance

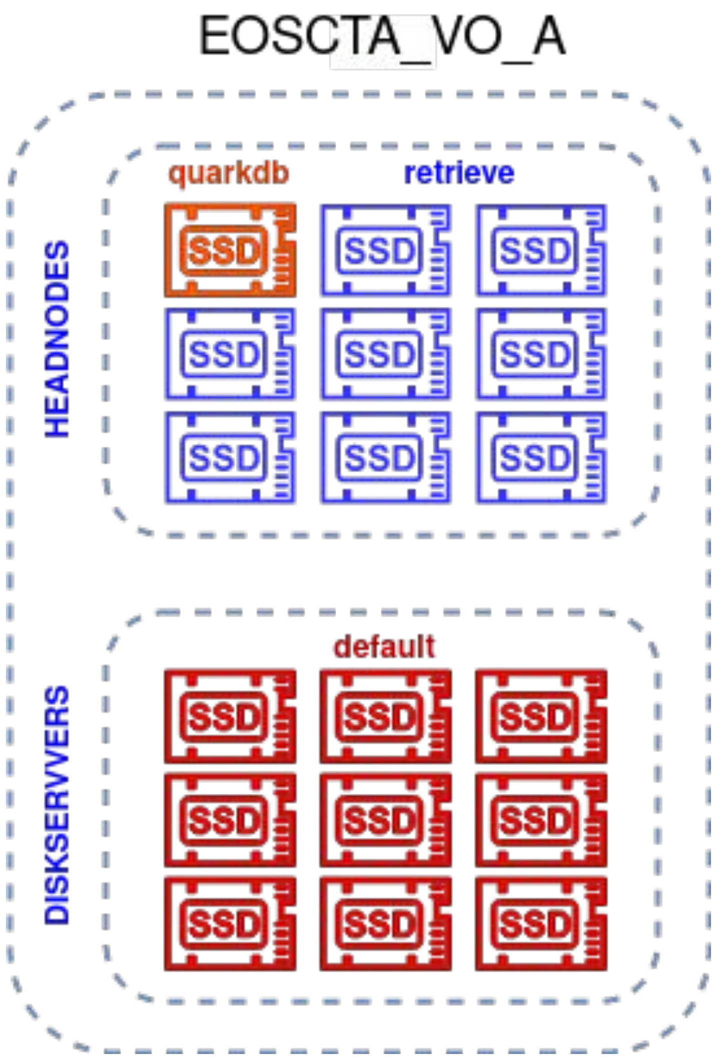
Space name	DATA source	DATA destination
default	Write data from EOS disk instance/DAQ	Read data for transfers to tape
retrieve	Write data from tape drives	Read data for transfers to EOS disk instance
spinners (optional)	Write data from retrieve space	Direct read-only access for reprocessing

You can operate an EOSCTA instance with less spaces but this convention will make your life easier

EOS+CTA Architecture



EOSCTA detailed



Rule: minimise operation pain

- loosing an EOS headnode is painful
- loosing EOSCTA archive space is painful

  NO PAIN² IF POSSIBLE

Hardware model for headnodes and diskservers is identical

EOSCTA HEADNODES:

- 1 SSD dedicated to quarkdb
- other SSDs for FST retrieve space

EOSCTA DISKSERVERS:

- all SSDs for FST default space

EOS+CTA Space Properties

- All files in *default* space are on their way to tape:
  - \`d1:t0\` and disk *default* replica deleted when successfully written to tape
- All files in *retrieve* space are on their way to EOS disk coming from tape:
  - \`d1:t1\` and disk *retrieve* replica *evicted* when successfully transferred out
  - disk replica deleted after 24 hours by the FST Garbage Collector: cta-fts-gcd

SSD spaces are (mostly) empty when everything is fine no file should stay more than 8 hours in these spaces

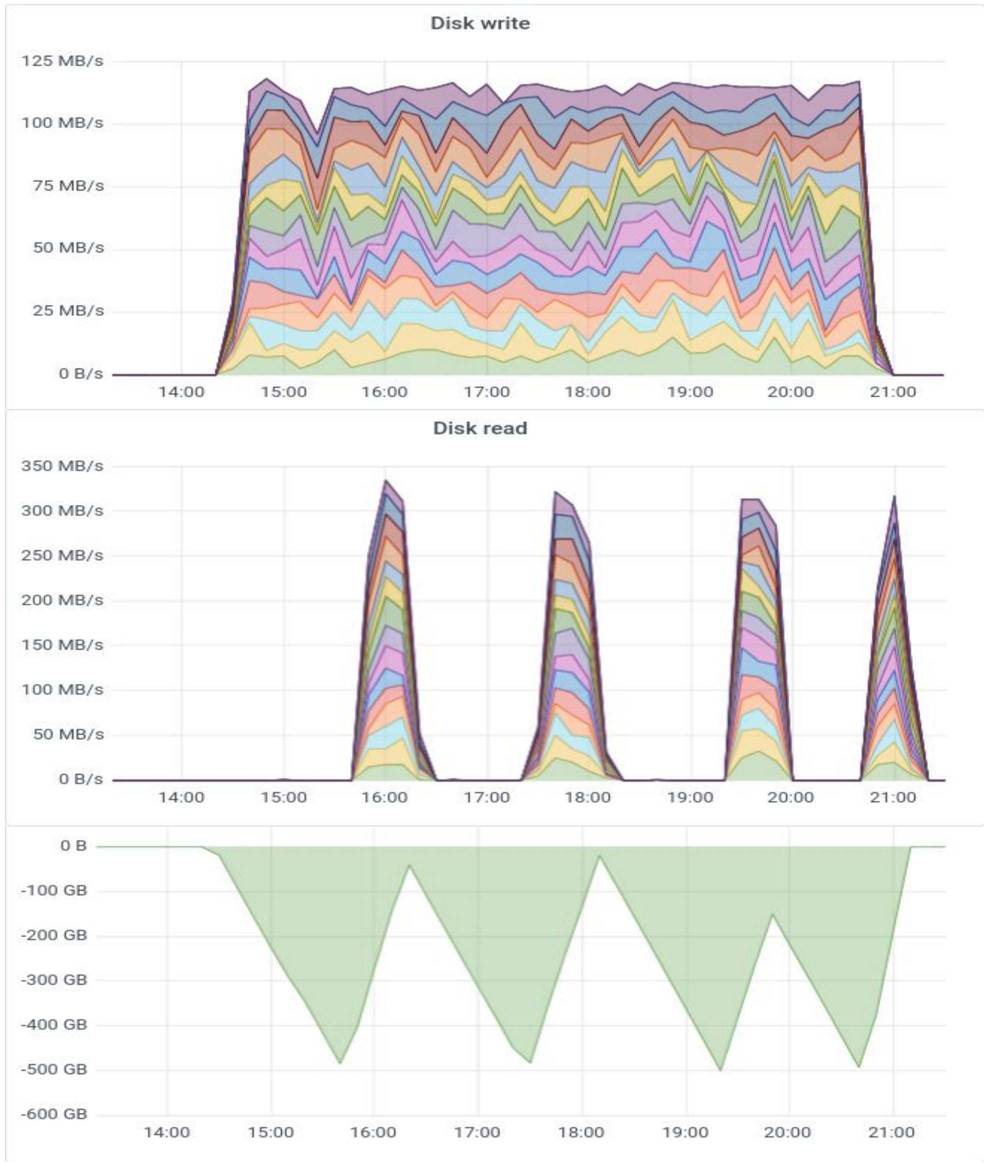


Fig 1: Default space during slow write to EOSCTA public instance

EOS+CTA backpressure mechanisms

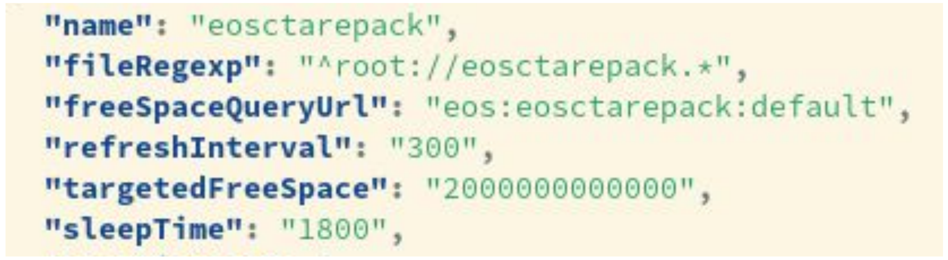
SSD spaces can fill up if space reader is too slow at evicting data

- *default*: when write bandwidth to tape is too low
  - not enough free tape drives, one or more tape libraries are down
- *retrieve*: destination EOS instance is abnormally slower
  - heavy experiment use, heavy disk operations (draining, rebalancing…)
  - similar heavy usage on *spinners* space

Backpressure mechanisms allows to temporarily slow down writers to the space

- `Destination is full` (*default*)
  - *xrdcp* to destination space fails as there is no space to allocate to write file
  - user retry write later
- Sleep tape retrieve queues (*retrieve*)
  - Dismount tapes, suspend VO retrieve queues for xx minutes: *cta-admin disksystem*
  - Free up tape drives for other tape mounts

EOS+CTA retrieve backpressure

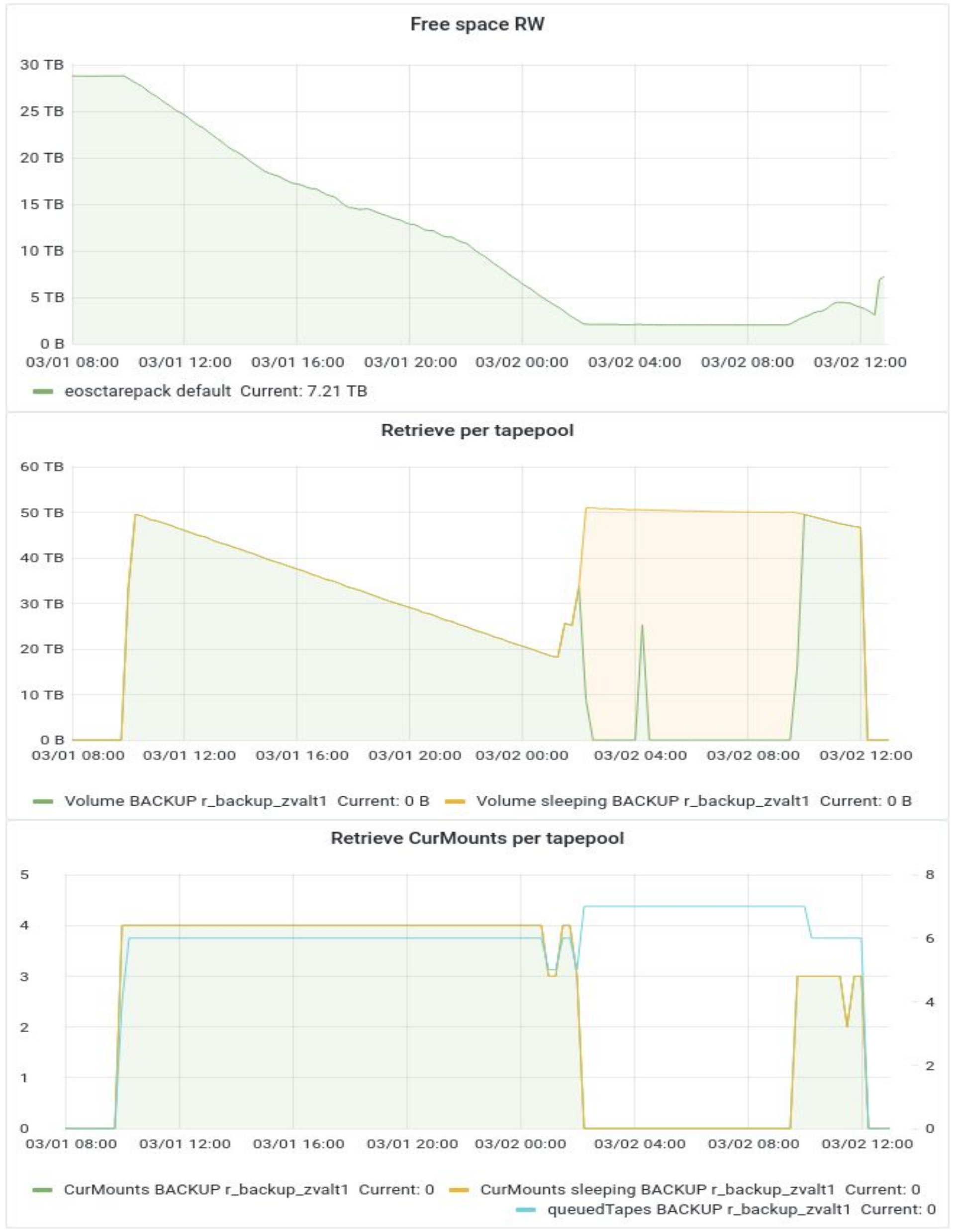


```
"name": "eosctarepack",
"fileRegexp": "^root://eosctarepack.*",
"freeSpaceQueryUrl": "eos:eosctarepack:default",
"refreshInterval": "300",
"targetedFreeSpace": "2000000000000",
"sleepTime": "1800",
```

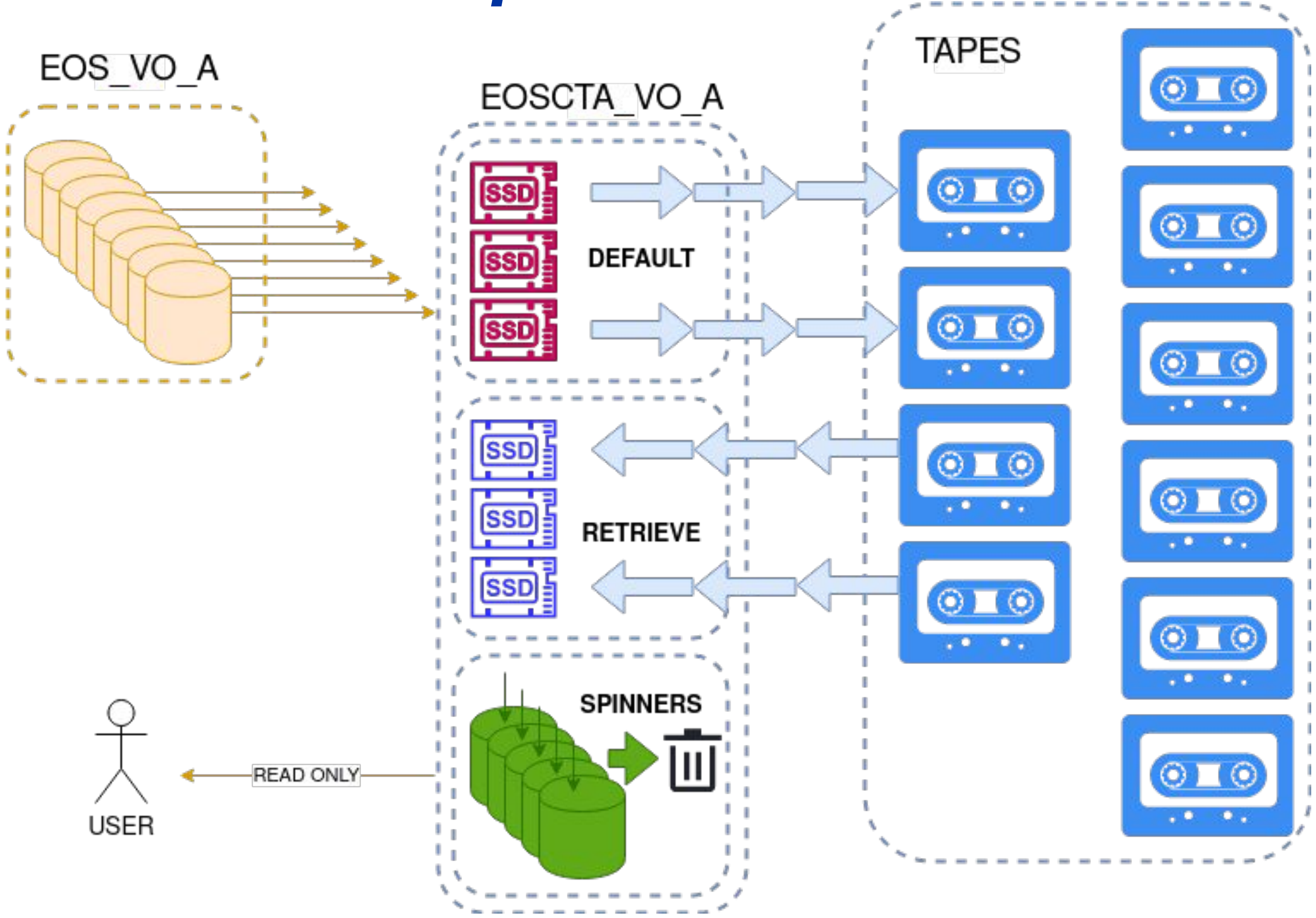
Fig 2: Disk system configuration for eosctarepack instance

Retrieving to an almost full space

- 4 tape drives reading tapes
- when 2TB of free RW space left
  - queued volume sleeps
  - mounted tapes filling space are dismounted
- when more free RW space is available
  - queued volume awakens
  - tapes are mounted



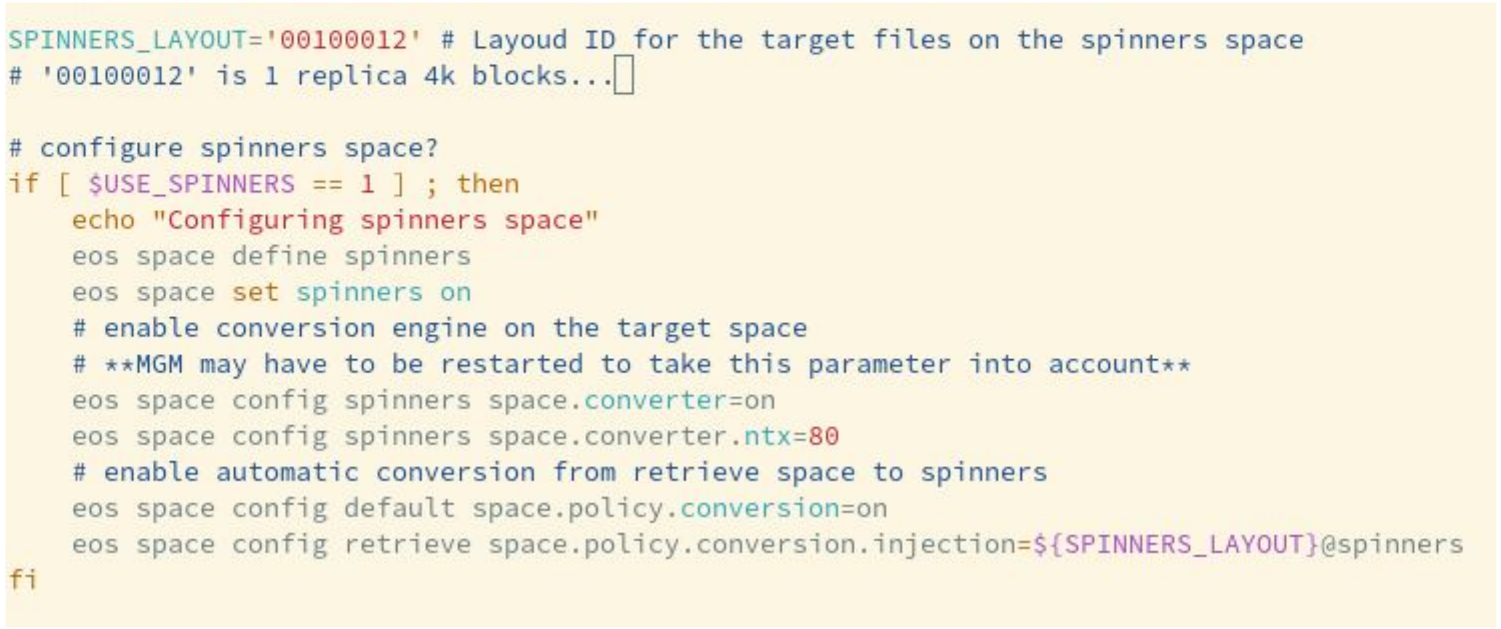
EOS+CTA Architecture *spinners* Addon



EOS+CTA Architecture *spinners* Addon

- *spinners* space is a natural extension of an EOSCTA instance
    - operates full with LRU garbage collection
    - *retrieve* space replicas are transferred to spinners space using EOS *converter*
        - same properties as *retrieve*: `d1:t1` files only
    - same standard backpressure rules apply

EOS MGM garbage collector makes room for new files



```
SPINNERS_LAYOUT='00100012' # Layoud ID for the target files on the spinners space
# '00100012' is 1 replica 4k blocks...

# configure spinners space?
if [ $USE_SPINNERS == 1 ] ; then
    echo "Configuring spinners space"
    eos space define spinners
    eos space set spinners on
    # enable conversion engine on the target space
    # **MGM may have to be restarted to take this parameter into account**
    eos space config spinners space.converter=on
    eos space config spinners space.converter.ntx=80
    # enable automatic conversion from retrieve space to spinners
    eos space config default space.policy.conversion=on
    eos space config retrieve space.policy.conversion.injection=${SPINNERS_LAYOUT}@spinners
fi
```

Conclusion

- *Dangerous* tape specific replica management features disabled by default in EOS
    - only on when `mgmofs.tapeenabled  true`
- Leverage and extend existing EOS concepts for tape needs
    - extended attributes, WFE, spaces, replicas, converter
- Add a few new ones
    - dX:tX, eos stagerrm, garbage collection

Try to offer a small toolbox that can be combined to match all tape use cases and keep complexity under control home.cern