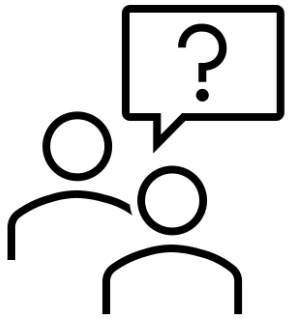
Deploying a new real-time XRootD-v5 based monitoring framework for GridPP

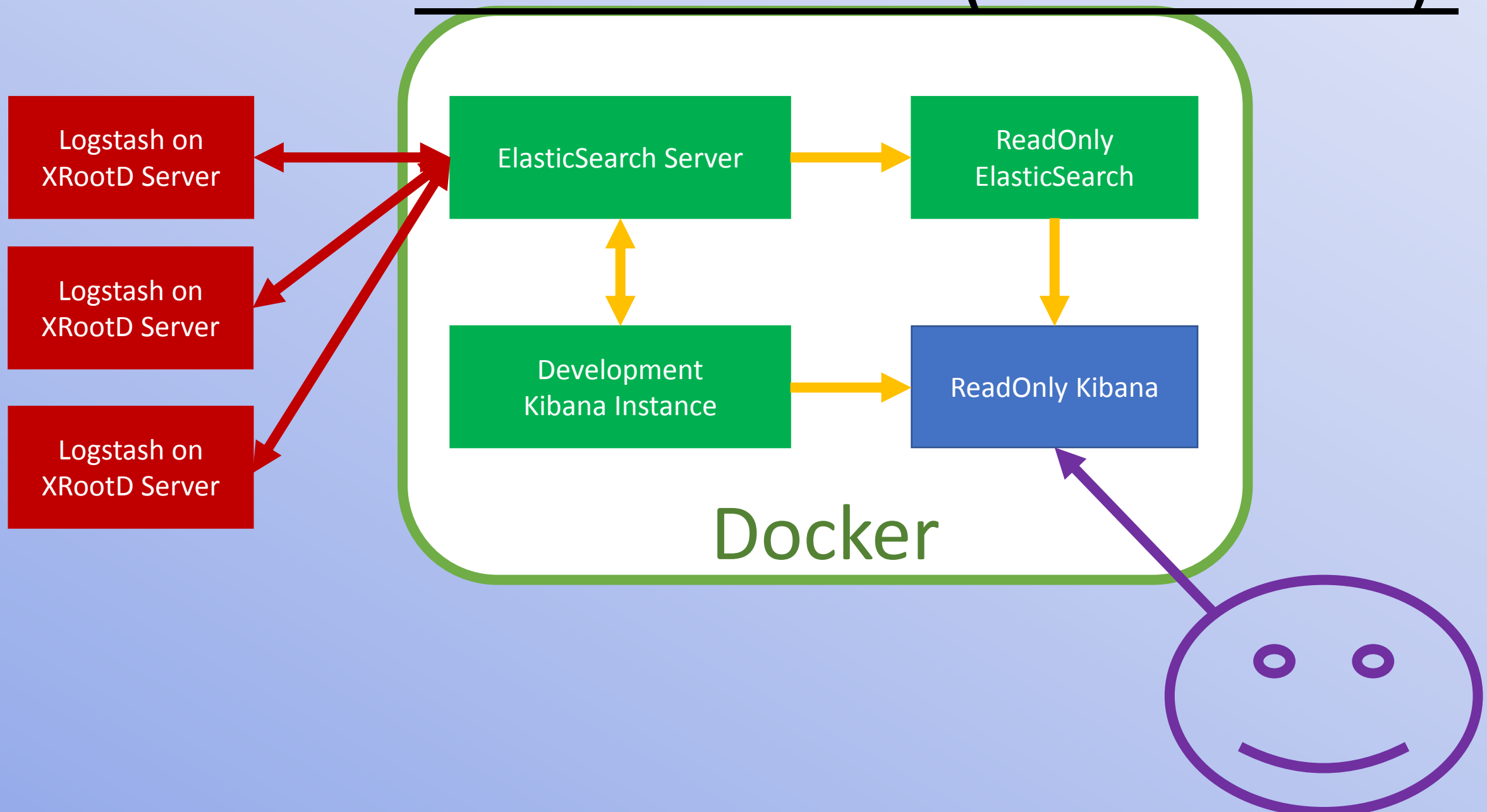
Robert Currie on behalf of Edinburgh GridPP group

What is this monitoring system designed to track?

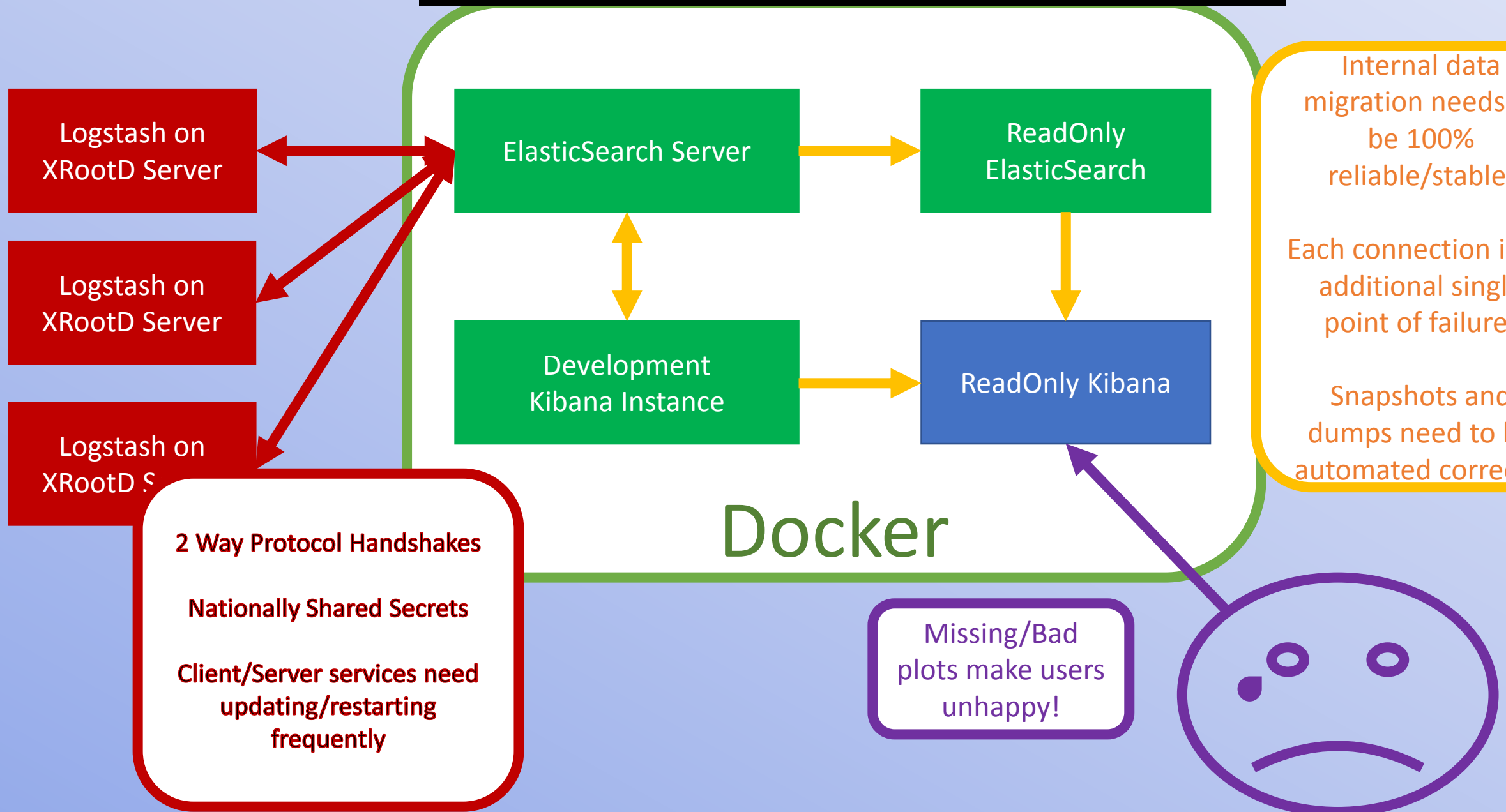


- Availability/Reliability of remote XRootD service
  i.e. is the service active? is it in use? is it working?
- Contents of XCache storage
  i.e. How many files are in the cache? Size of cache?
- Statistics about the XCache Networking performance
  i.e. How much data is flowing in/out? How many connections?
- How effective is this as a cache?
  i.e.
  Does the cache reduce the total bandwidth in/out?
  Are the number of constant connections to the backend reduced?
- Ultimately;
  Does site/job-efficiency/reliability improve?

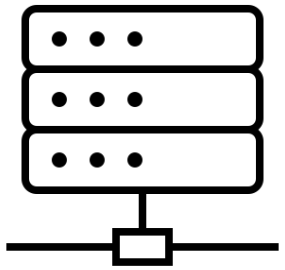
The 2018 Plan (Slate based):



The 2019/2020 Reality:



How is this data collected?



Old system based on Slate (2018):

- Custom tooling based on SlateCI which manually walks the posix filesystem and reports (via distributed Logstash)
- Custom tooling which reports system stats (via Logstash)
- Pseudo-real-time granularity with buffered data flow

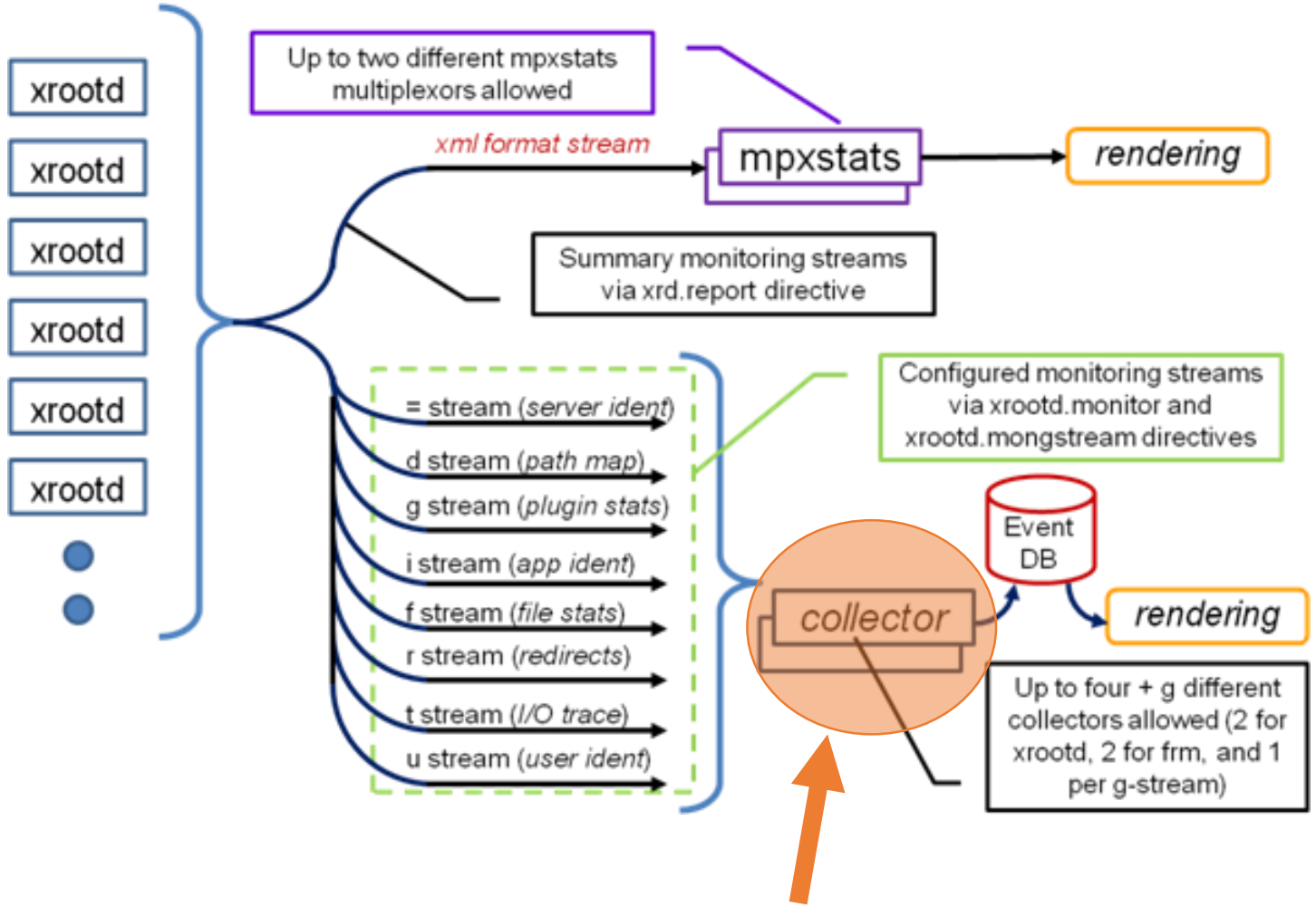
New system based on OSG XRootD collector:

- XRootD internals tracks file access and cache metrics closer to real-time.
  Datagrams broadcast from the service itself via UDP.
- Realtime granularity *and* data flow

XRootD 5.x service monitoring infrastructure

Directly From XRootD docs:
(no attempt to take credit)

https://xrootd.slac.stanford.edu/doc/dev51/xrd_monitoring.htm

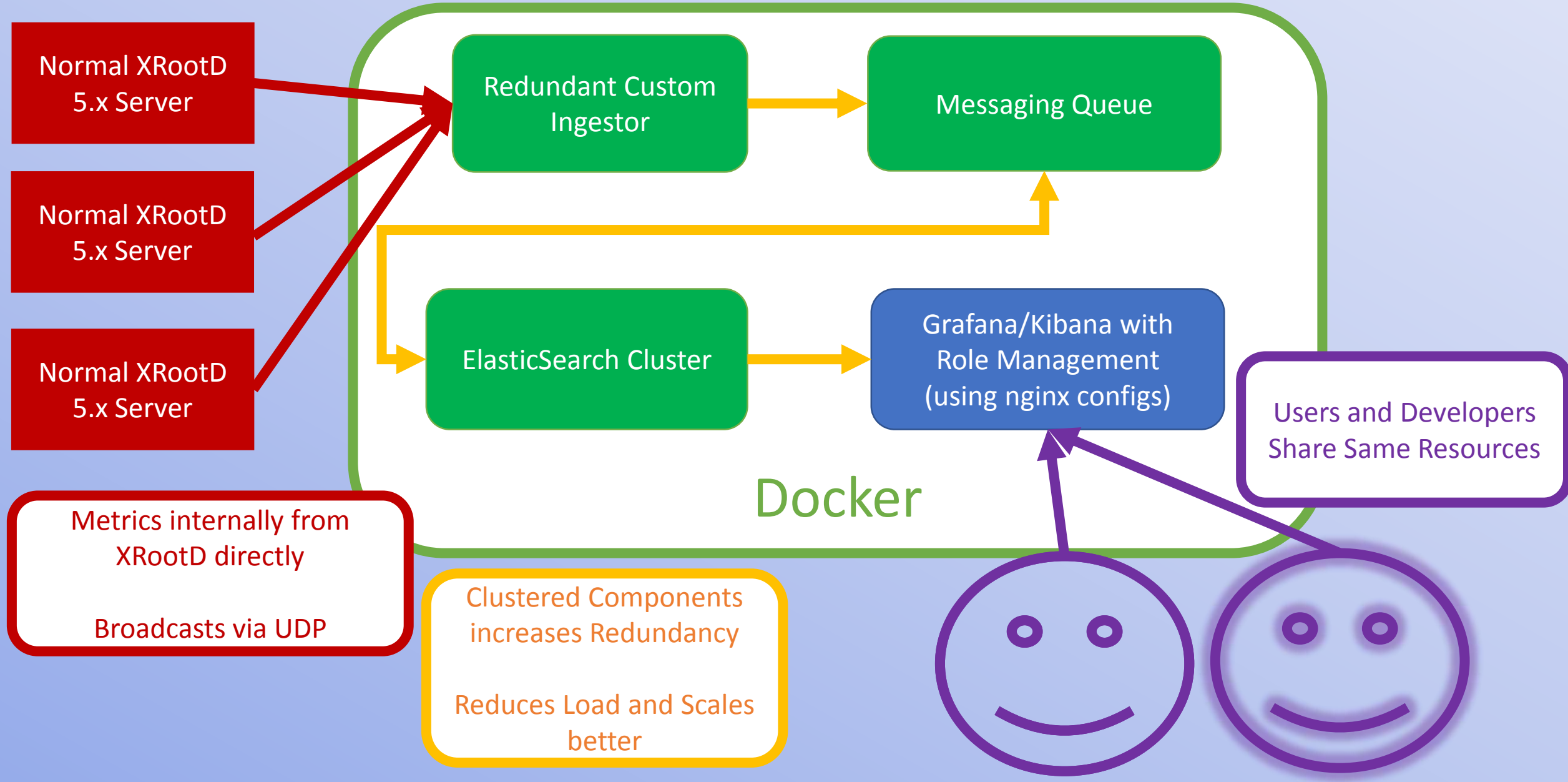


Need a collector capable of parsing
Both normal and g-stream (caching) metrics

What does this new system need to address?

- No requirement for sustained connection between sites.
  i.e. Sites can fire and forget their metrics
- Much less effort for site to support/maintain.
  Change config, restart, forget!
- More accurately monitor if a remote service is active
- Less stress on the XRootD proxy server/service
- More accurate metrics:
  i.e. tracking what XRootD actually knows
  vs, reverse engineering this
- New monitoring pages at fixed locations for users/site-admins

The 2021 Plan (OSG based):



Which services are we able to monitor?

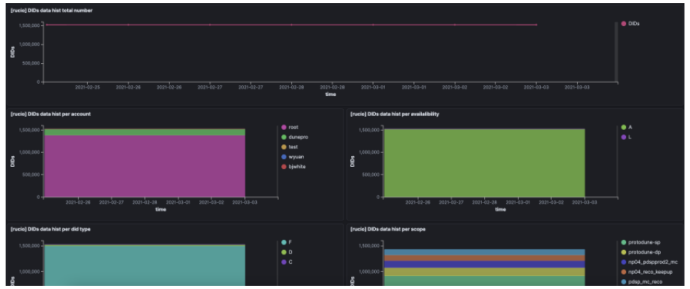
- We are monitoring a “Transparent” XCache at the Edinburgh Tier2 computing site.
  (Intercept transfers and add caching for running jobs)
- Added monitoring to the Birmingham XRootD-4.x site and XCache services
- Added monitoring to the Edinburgh-DPM site SE
  (can only track (x)root access, but better than no data to work with)
- Able to show if the XCache service at Edinburgh reduces backend work for site storage
  (aim here is to improve site A/R for ATLAS)

https://monitoring.edi.scotgrid.ac.uk

Edinburgh-GridPP Monitoring

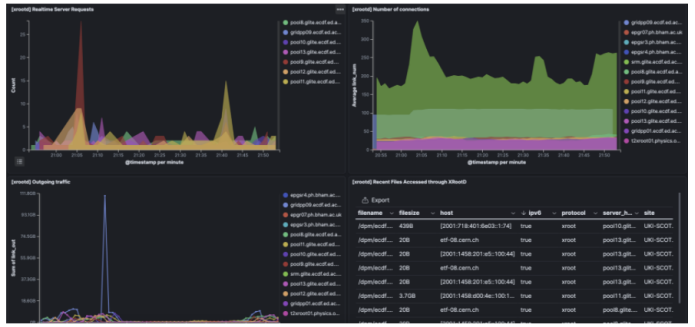
***new* for 2021!**

DUNE 7Day RUCIO Monitoring

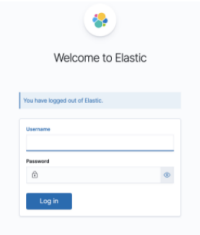


DUNE UK Monitoring

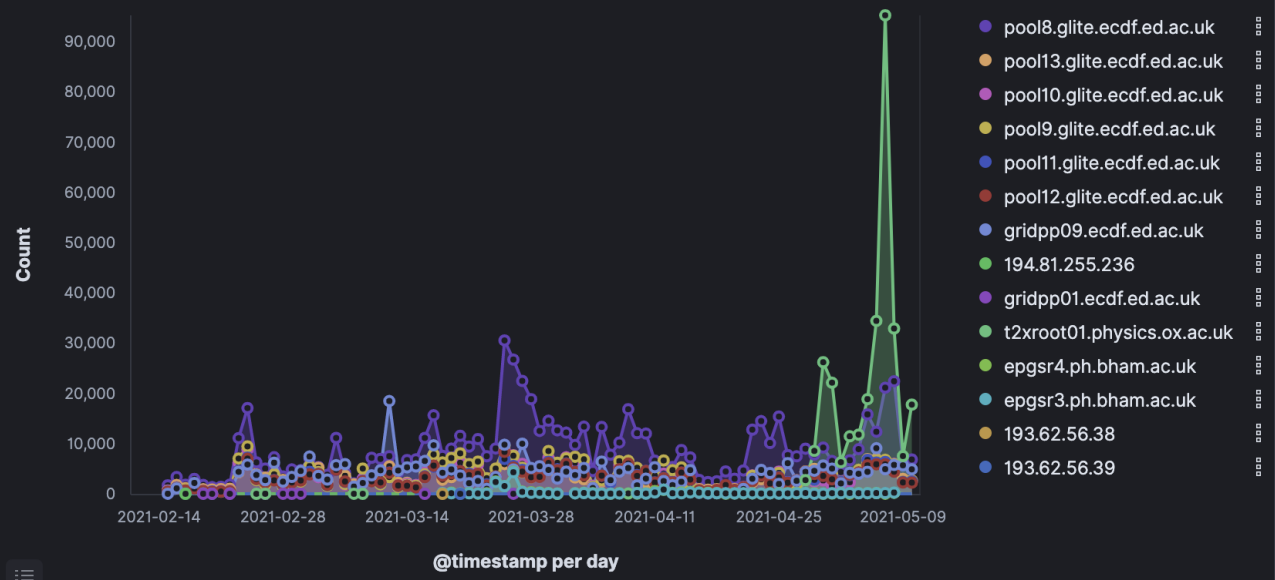
GridPP XRootD Monitoring



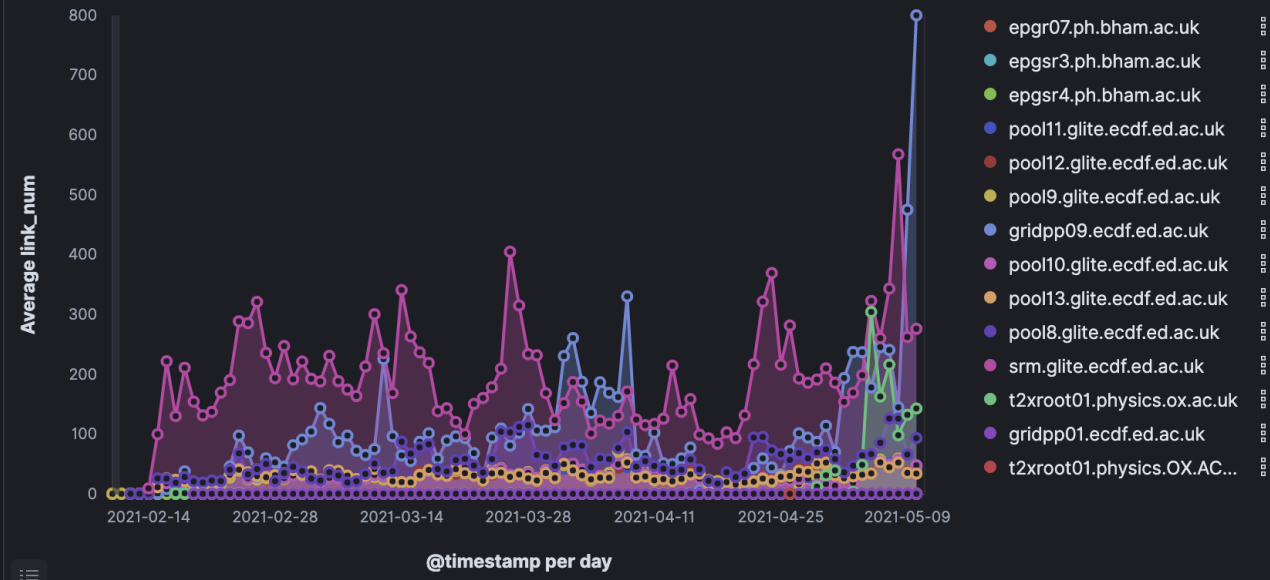
Dev Kibana Instance



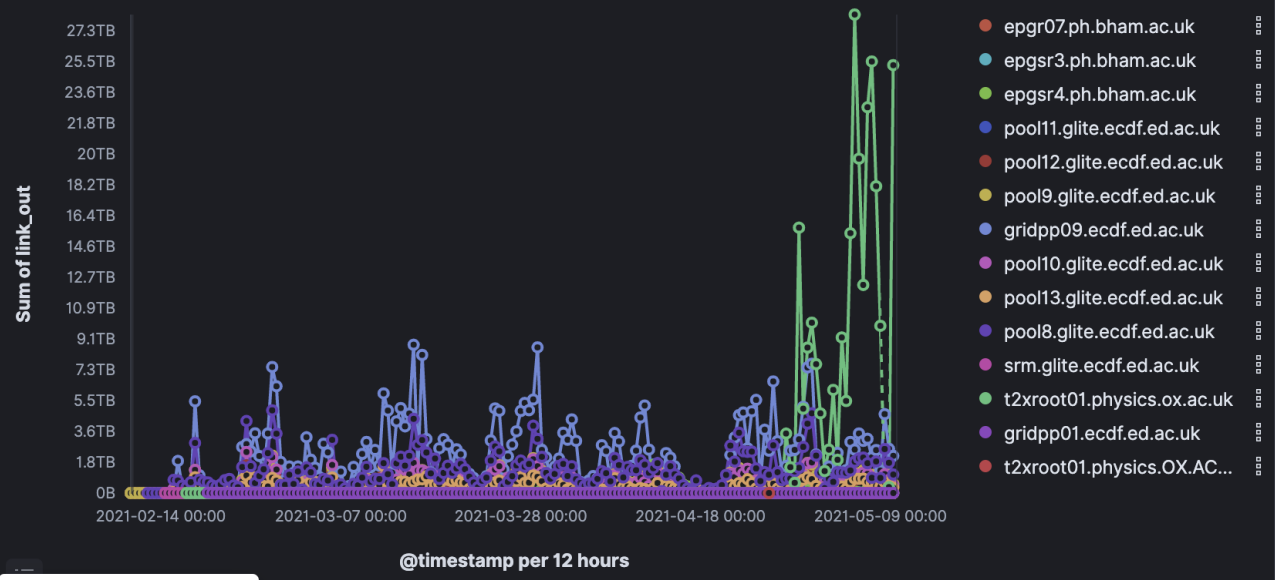
[xrootd] Realtime Server Requests



[xrootd] Number of connections



[xrootd] Outgoing traffic



[xrootd] Recent Files Accessed through XRootD

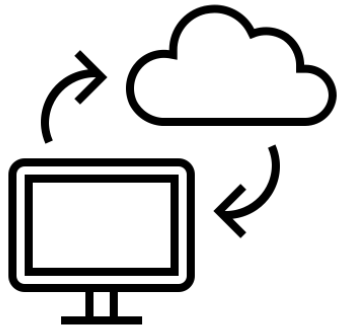
Export

filename	filesize	host	ipv6	protocol	server_h...	site
/root:/xroot...	134.4MB	t2wn170.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	222.7MB	t2wn109.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	138MB	t2wn170.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	134.6MB	t2wn170.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	219.4MB	t2wn109.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	135.2MB	t2wn170.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	64.9MB	t2wn167.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...
/root:/xroot...	219.9MB	t2wn109.physics.ox.ac.uk	false	xroot	t2xroot01....	UKI-SOUT...

Initial Results (XCache-Hardware)

Transparent XCache Server at Edinburgh:

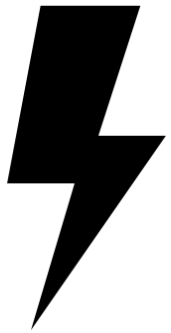
1 server buffering traffic between ~1kCPU cores and ~1PB of site storage.



Resource	Was this adequate	Comment
Storage: 16TB (10 x 2TB disks, ZFS)	Yes, for testing, **should be bigger**	Use oscillated around 10TB with no problems. However, there is **significant churn** on a server this small.
RAM: 32GB RAM	Yes, **more is always better**	With ZFS caching only 24GB normally in use
CPU: 2x E5620 (8core,16thr,aes/sse4)	No, **working to make this yes**	Lots of **100% CPU usage** (XRootD), potentially checksum related, **likely main bottleneck**
Network: 1 x 10Gbps	Maybe	Bottlenecks appeared to be elsewhere
CentOS7: Kernel-LT (5.4.xx) with latest updates	**Yes**, with modern kernel	Higher kernel version allowed for better performance. (No reason not to use Ubuntu)
XRootD 5.1+: Tracking 5.x series releases	**Yes**	Appears to just work as expected.

Edinburgh XCache Performance

(*first 3 months 2021*)



- Total broadcast from cache to WN: 445TB
- Total ingested to cache from “remote” storage: 388TB

15% bandwidth reduction

- Cached data requests are dealt with by XCache server directly.

33% less requests over WAN to SE

- 20% of cached files have access count >1
  Some files have access count in excess of 1k! (user data)

- Traffic filtering/smoothing via **tc** and **XRootD** has been experimented with.
  Impact on job/site efficiency is yet unclear.

Conclusions

- We have built upon the OSG collector to collect additional caching metrics from distributed XCache instances.
- Redesigned our Edinburgh ELK+ monitoring stack to be more resilient/scalable/secure.
- Have taken some preliminary measurements of metrics such as cache-efficiency/effectiveness using new monitoring stack.
- Demonstrated that deploying an XCache at a site can offer improved performance.
- Examine potential performance/scalability improvements in both XCache and monitoring system.

BACKUPS

What is required to collect this monitoring data?

Old system based on Slate:

- Custom code needs to run on Cache system
  (install via pip or docker but required installing something)
- Shared secrets to broadcast data back to nodes at Edinburgh
- Collecting metrics is expensive

New system based on OSG XRootD collector:

- Just need to change the XRootD config at the remote site:

xrootd.conf:

```
all.sitename RemoteSite
…
if exec xrootd
    xrd.report monitoring.edi.scotgrid.ac.uk:9931 every 1m all
    xrootd.monitor all auth fstat 60s lfn ops ssq xfr 1024 rnums 1 ident 1m dest ccm files fstat info pfc redir user monitoring.edi.scotgrid.ac.uk:9930
fi
```

What is in the collected metrics?

Data collected comes from the XRootD Monitoring system.

https://xrootd.slac.stanford.edu/doc/dev50/xrd_monitoring.htm

https://xrootd.slac.stanford.edu/doc/dev51/xrd_monitoring.htm

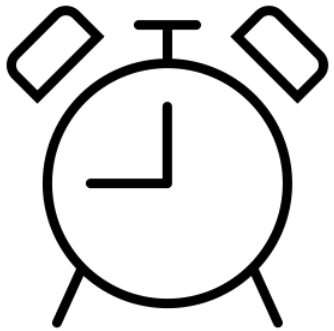
xrootd.monitor: Datagrams containing details of file access.
e.g. bytes-read, protocol-used, user, file-size

xrd.report: Summary data of the xrootd service.
e.g. memory-used, nproc and cache-summary*

All data reported using XRootD streams which are reasonably well documented.

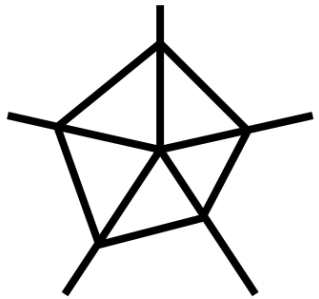
*new in XRootD 5.x

Future Planned Work



- Working to integrate monitoring of more sites as part of GridPP XCache project.
- Understand how/if better file purging/caching decisions can improve cache performance. Currently using default (FIFO?) system in XRootD.
- Working to build additional dashboards to give real-time monitoring to sites looking to use an XCache.
- Quantify how well an XCache could perform compared to real-world results.

Improved distributed metric handling



- Collecting metrics has been greatly simplified
  - XCache metrics collected from UDP traffic
  - Other ELK metrics still via http(s) to ElasticSearch directly
- ELK stack already in use by other contributors

  - :9200 via http for legacy compatibility
    (nginx upgrades this to https)
  - :9201 via https using nginx proxy
    this allows different public/internal certs as needed
- Kibana roles combined with additional nginx proxies provide fully public dashboards with no shared credentials:

https://monitoring.edi.scotgrid.ac.uk