CVMFS Mix at CERN

F.Furano
CERNVM workshop 2022

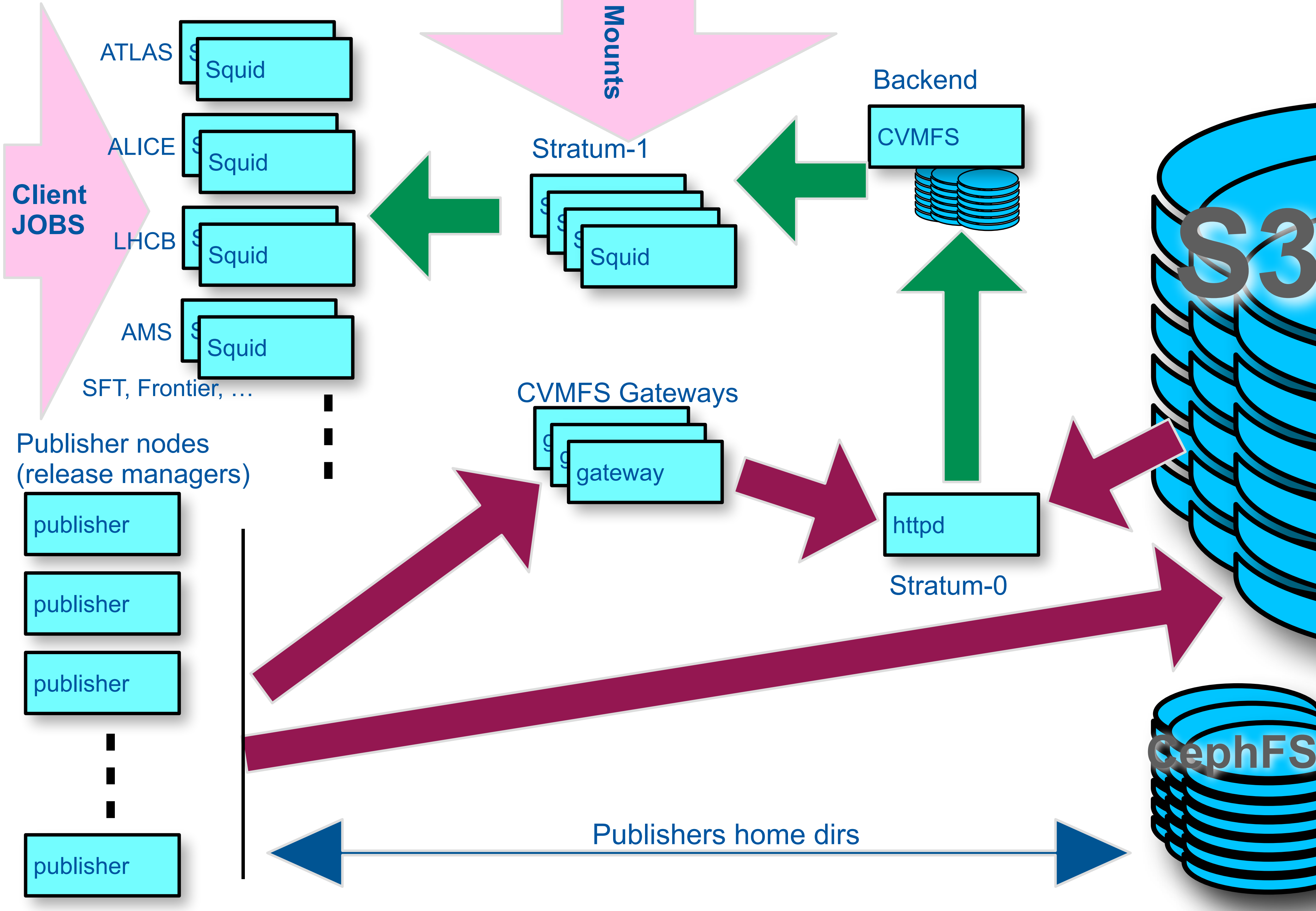
CVMFS at CERN

- About 4.5 billion files
- >200TB of data

- 63 release managers
- 4 stratum-1
- 24 caching proxy machines
  - 15KHz req rate (peak 30KHz)
  - >400 MB/s aggregate throughput (peak 1GB/s)

CVMFS at CERN

- Various waves of upgrades and config evolution
- Converged to a unique deployment model
- Main points:
  - Bulk data on S3 (Ceph)
  - Release managers homes on CephFS
  - Focus on robustness and redundancy


Mounts
Client JOBS
ATLAS
Squid
ALICE
Squid
LHCB
Squid
AMS
Squid
SFT, Frontier, …
Stratum-1
Squid
Backend
CVMFS
S3
CVMFS Gateways
gateway
httpd
Stratum-0
Publisher nodes (release managers)
publisher
publisher
publisher
publisher
CephFS
Publishers home dirs

Monitoring probes

- Beside the “usual” performance graphs
- Challenge: not more than 30 seconds for a human to tell if the service is OK
  - Basic status of each machines (e.g. overload)
  - Basic status of cvmfs, presence of the sw (e.g. cvmfs_server mount -a)
  - Status of the synchronization among the various stratums at CERN

- Also many alarms (e.g. synchronization glitches)
- Also the "usual" CERN IT alarms (e.g. HW failures)

New dashboards

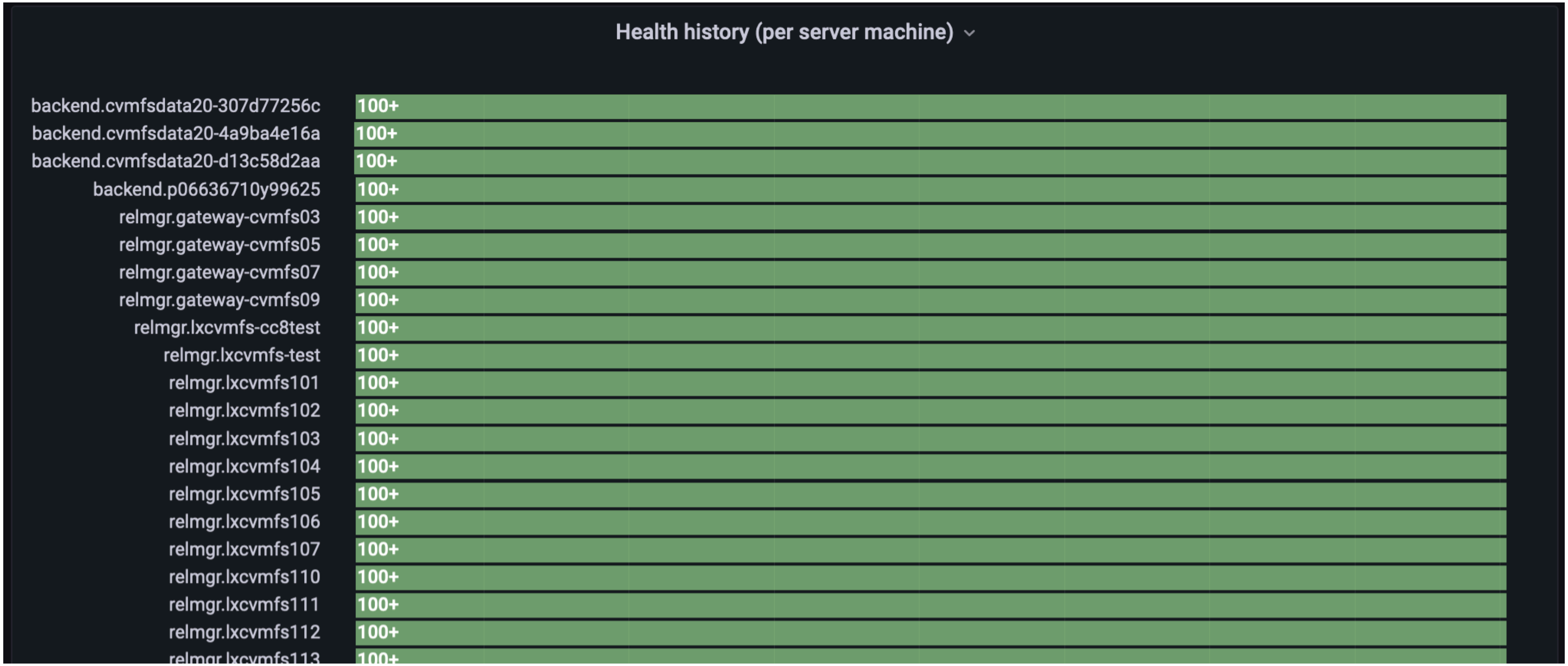
- Mandatory: coherent data in a coherent syntax
- Mandatory: must self-populate! No manual actions to add machines/repos etc.

- 2 scripts sending compatible data
  - per host (running on the host)
  - per repo (running in a probe machine)

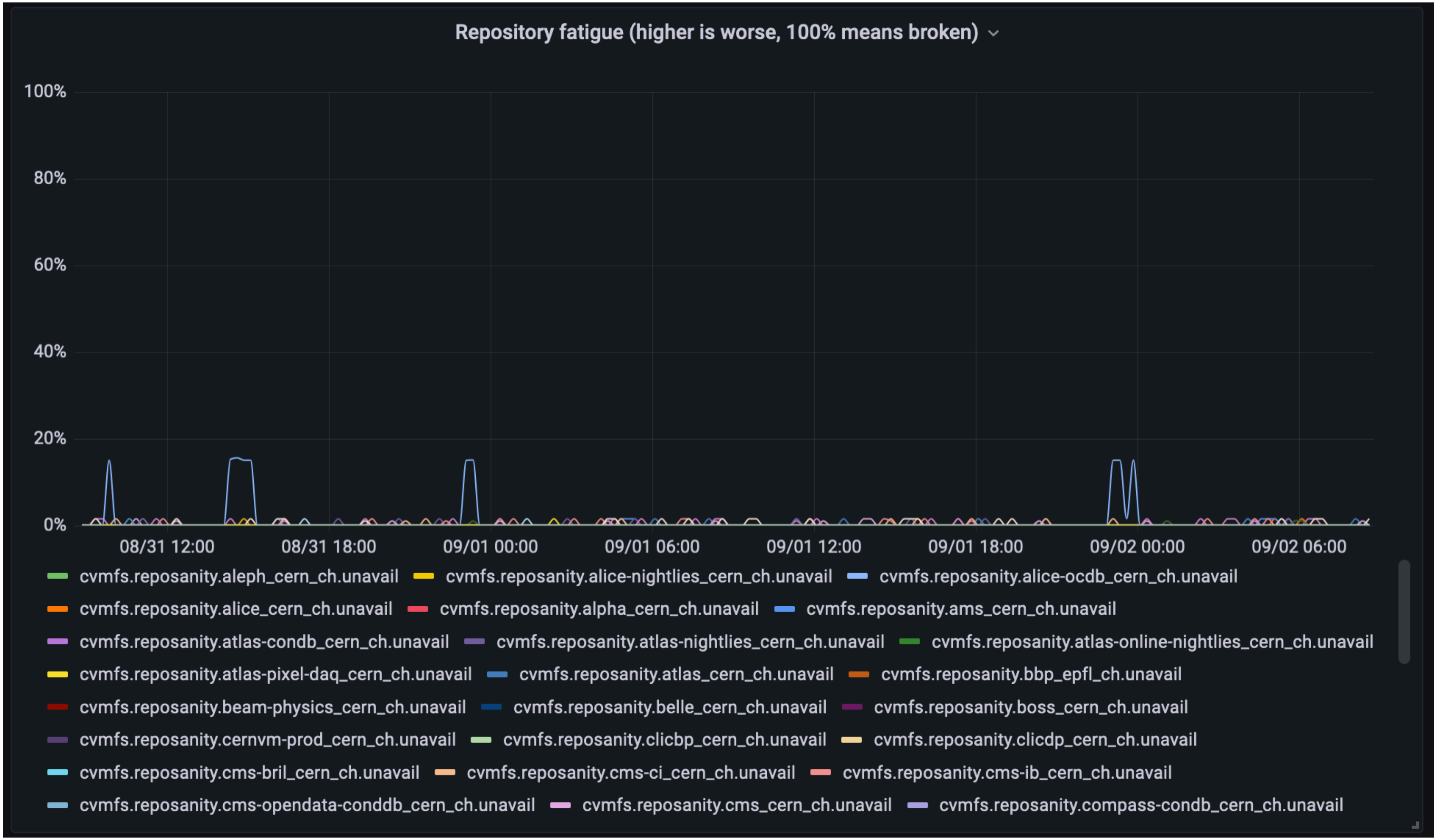
- These scripts send numbers to Grafana
  - pre-computed on the fly, with HW<->repo relationship
  - e.g. repo stress index is a relatively sophisticated computation

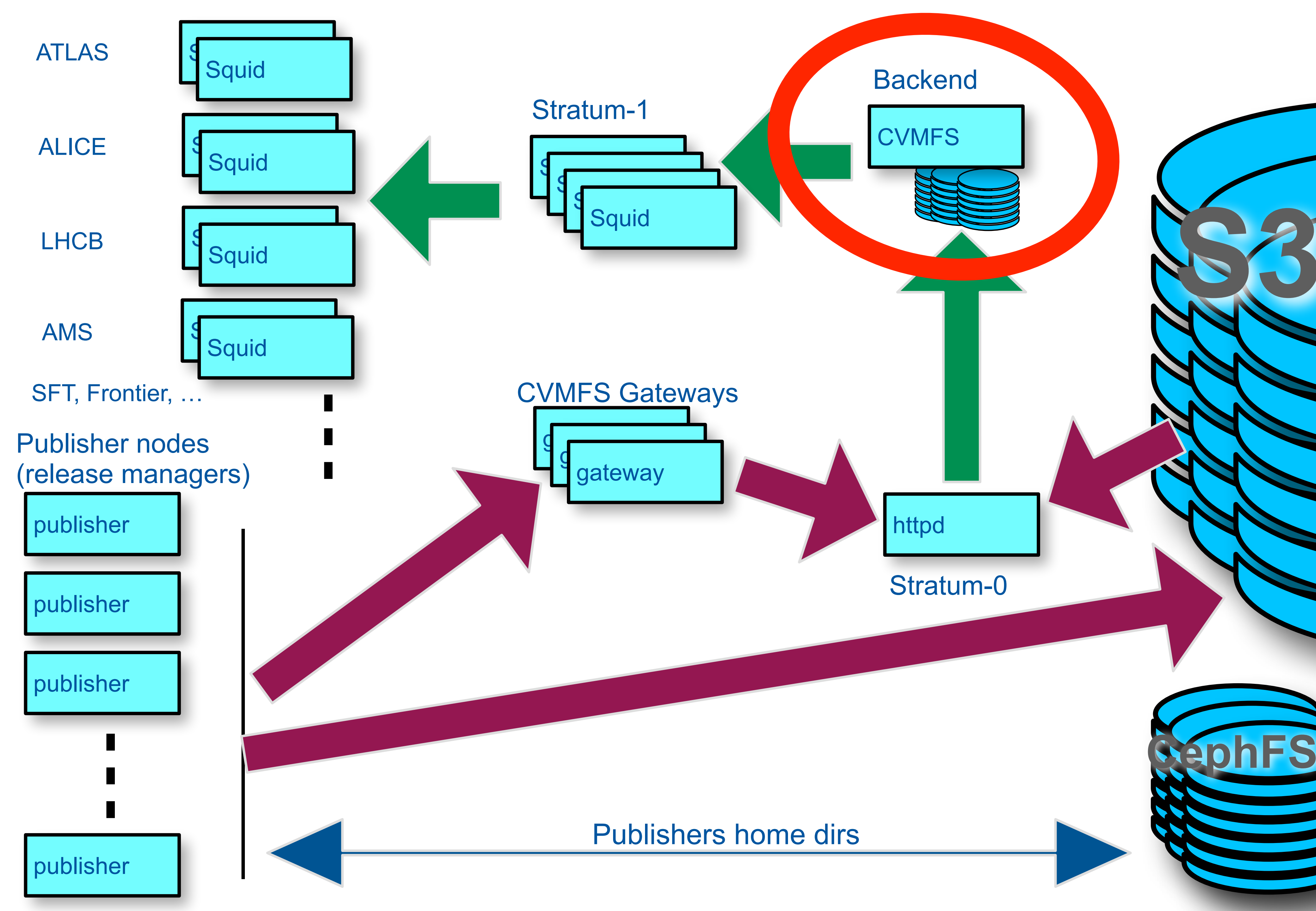
- The intrinsic coherency makes it easier to use Grafana
  - Much simpler queries
  - Still many…

By host example


Health history (per server machine)
backend.cvmfsdata20-307d77256c 100+
backend.cvmfsdata20-4a9ba4e16a 100+
backend.cvmfsdata20-d13c58d2aa 100+
backend.p06636710y99625 100+
relmgr.gateway-cvmfs03 100+
relmgr.gateway-cvmfs05 100+
relmgr.gateway-cvmfs07 100+
relmgr.gateway-cvmfs09 100+
relmgr.lxcvmfs-cc8test 100+
relmgr.lxcvmfs-test 100+
relmgr.lxcvmfs101 100+
relmgr.lxcvmfs102 100+
relmgr.lxcvmfs103 100+
relmgr.lxcvmfs104 100+
relmgr.lxcvmfs105 100+
relmgr.lxcvmfs106 100+
relmgr.lxcvmfs107 100+
relmgr.lxcvmfs110 100+
relmgr.lxcvmfs111 100+
relmgr.lxcvmfs112 100+
relmgr.lxcvmfs113 100+

By repo example




ATLAS
Squid
ALICE
Squid
LHCB
Squid
AMS
Squid
SFT, Frontier, …
Stratum-1
Squid
Backend
CVMFS
S3
Publisher nodes
(release managers)
publisher
publisher
publisher
publisher
CVMFS Gateways
gateway
httpd
Stratum-0
CephFS
Publishers home dirs

CERN CVMFS backend

- beefy machine
  - 186 TB raid-6
  - 40 cores
- snapshots **all the repos** continuously from S0 (which is a gateway to S3)
- Serves the S1 caches at CERN

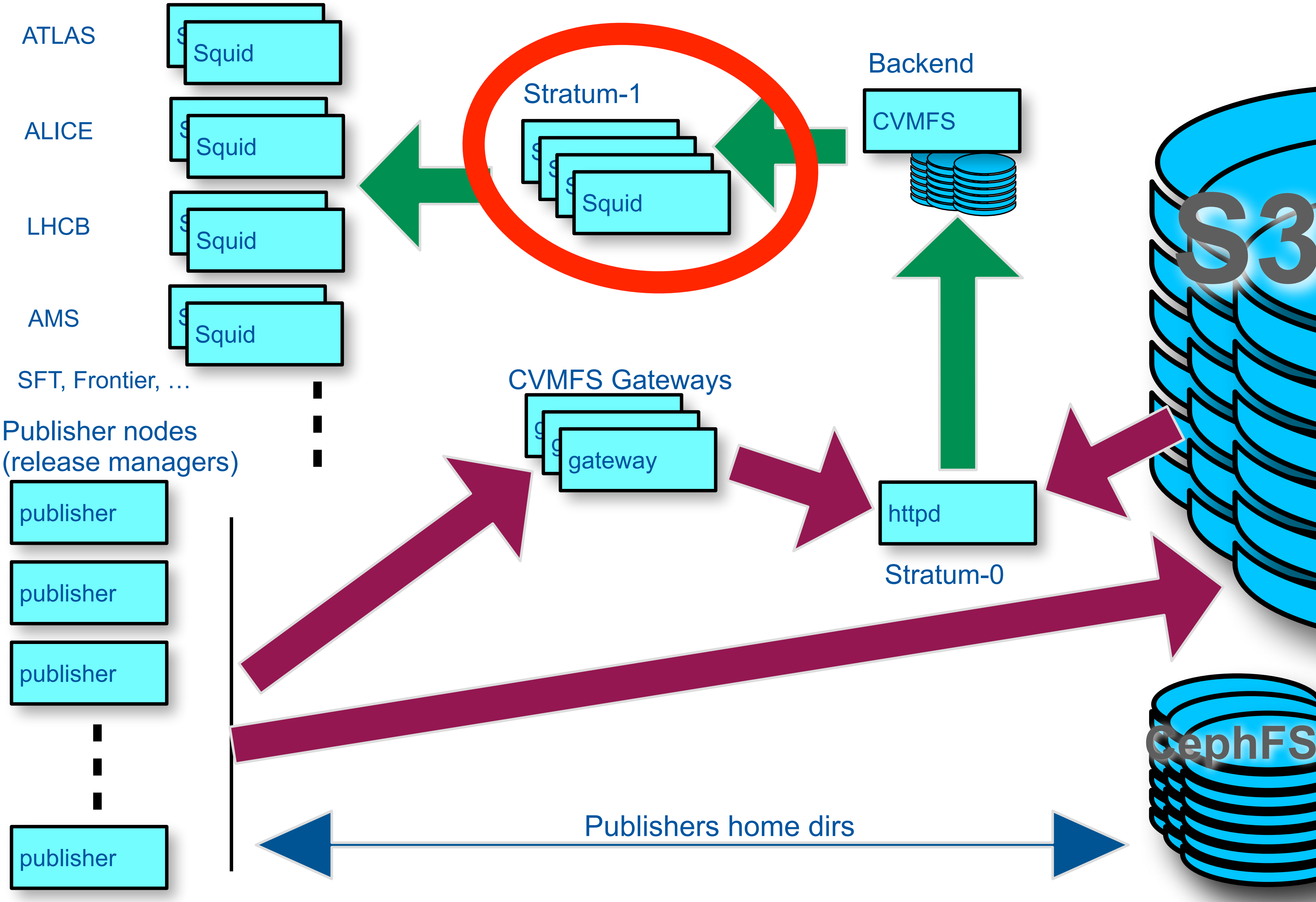
- Single point of failure, however with respectable uptimes (up to years)
  - hard to demonstrate that whatever other solution works better

CERN CVMFS HA backend

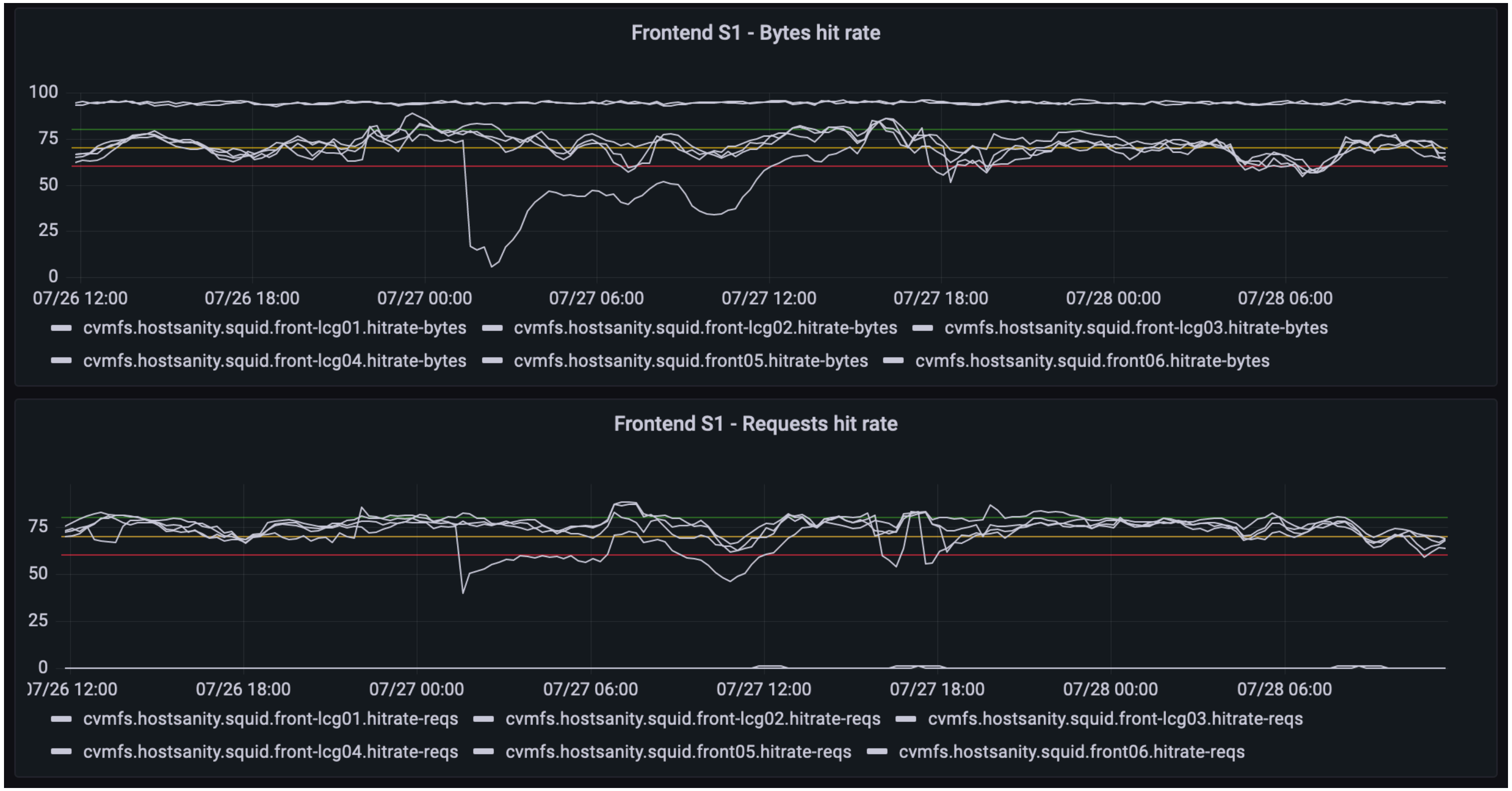
- In 2021 we started testing the CVMFS-ha scripts by D.Djikstra
- Needed some work to well polish their integration with the newer Linux-HA components
  - Wrote plugins for pacemaker
  - Needs special router config, with an IP address that can bounce between the two machines
    - This puts constraints on the deployment of the two machines, e.g. connected to the same router
  - A probe sends data about the internal alignment to our Grafana

CERN CVMFS HA backend

- Does it work better? Difficult to tell!
- Surely it's more complex and seems to work fine
  - It well resisted our tortures, killing machine, etc.
  - Will it resist time? Will we manage it right after one year of perfectly working silence?
- Managing it needs basic understanding of **`pcs`** and a few more recipes in our internal docs
- We decided to keep it as "hot spare" for the glorious single host backend


ATLAS
Squid
ALICE
Squid
LHCB
Squid
AMS
Squid
SFT, Frontier, …
Stratum-1
Squid
Backend
CVMFS
S3
CVMFS Gateways
gateway
httpd
Stratum-0
Publisher nodes
(release managers)
publisher
publisher
publisher
publisher
CephFS
Publishers home dirs

Focus: S1 hit rate



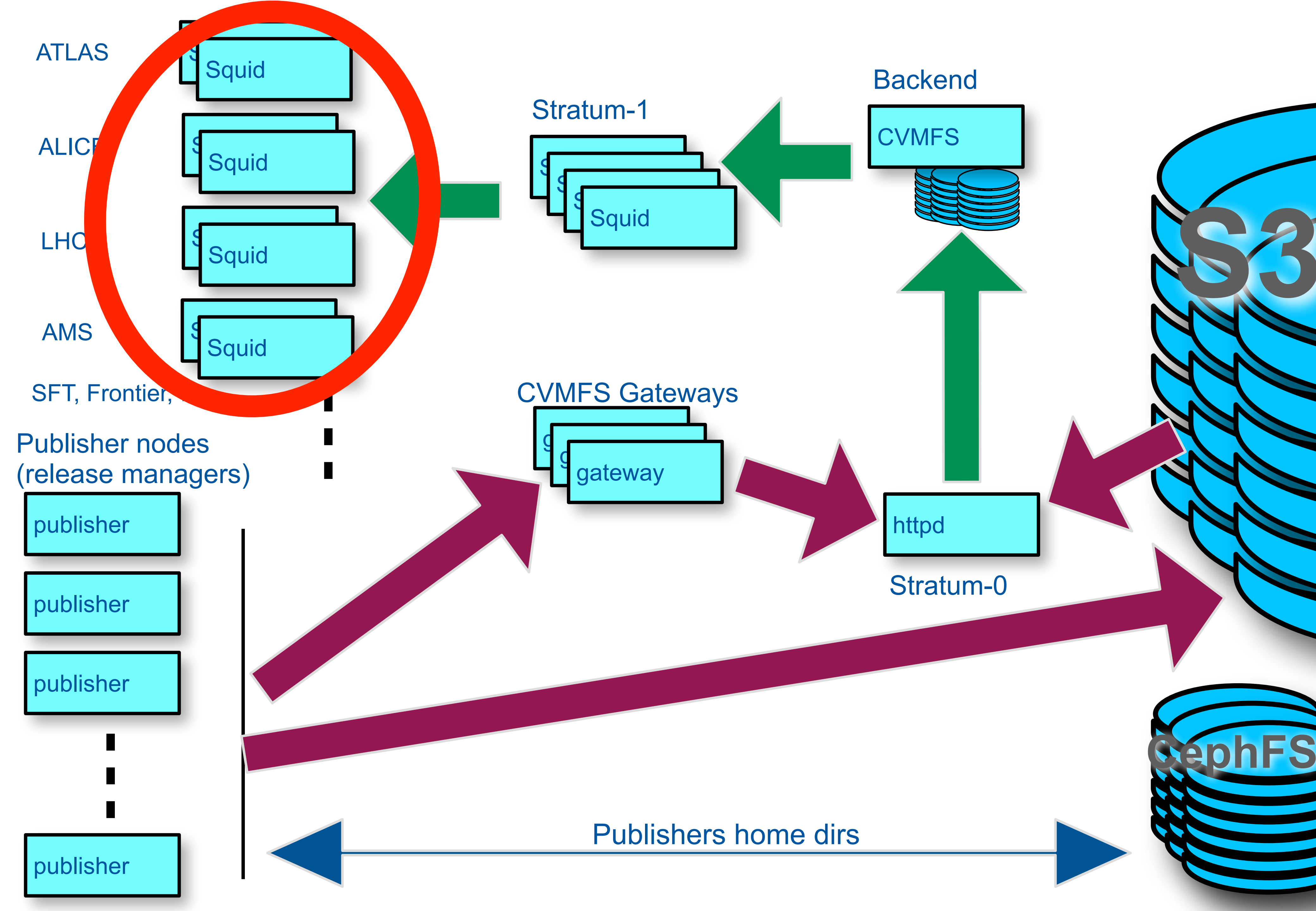
- Hit rate on the 2nd level squid caches
- Still OK-ish, system is up, however uhm… looking for space for improvement

Focus: S1 hit rate

- 75% when it goes well
- How much scalable is this part? How will it perform if we multiply by 4 either
  - the working set size
  - the throughput requested
  - both
- This is open to further discussion

Focus: squid clusters

- The point is that just putting squids aside behind a DNS alias (e.g. cvmfs-stratum-one) makes a suboptimal cluster
- 8 squid processes (2 per machine, 4 machines)
  - each file is cached 8 times, and has to be fetched 8 times from the backend
- Squid in reality does have proper clustering based on internal tunnelling
- Not compatible with the data volumes we have, it would multiply the internal network consumption
- The best workaround for this so far has been partitioning the traffic…


ATLAS
ALICE
LHCb
AMS
SFT, Frontier, ...
Squid
Squid
Squid
Squid
Stratum-1
Squid
Backend
CVMFS
S3
CVMFS Gateways
gateway
httpd
Stratum-0
Publisher nodes
(release managers)
publisher
publisher
publisher
publisher
Publishers home dirs
CephFS

Ourproxy clusters

- One scaling way that was exploited
- Put an additional layer of caches on top of S1, serving CERN jobs
- Giving a “private” cache to individual big data consumers at CERN (3-4 machines each, 160GB)

- This reduces the load on S1, which has to serve external sites and mounts
- At the price of more HW
  - Every squid process runs as an individual cache … means pretty high redundance

Conclusion

- The deployment is remarkably stable
- Needs non negligible maintenance effort
  - e.g. to allow quasi-transparent interventions
- (frontier)Squids work fine, at the price of data (and hw) multiplication
  - And quite some traditionally delicate ops on the aliases for interventions
- Would welcome the HTTP caching tech to become more cluster-friendly