



WELCOME

Roberto Valverde

14th October 2026, Zurich





Accelerating CephFS Backups with Recursive Change-Time Detection

Roberto Valverde, Paulo Saraiva – CERN

CEPH Days Zurich - 14.10.2026

Our Backup Context: CBACK, a **restic** wrapper

- **CBACK** is a backup orchestrator developed at CERN with the goal to offer daily snapshot-based backup for **CERNBox**, **CephFS**
- Wrapper around **Restic**
- In production as main backup system since **2020**
- **Some numbers:**
 - **+40,000 backup jobs** performed daily
 - **~45TB** added daily, **~4.5TB** of space freed daily
 - **11.50 PB** stored in S3
 - An average of **~60 restores** performed per year
- Open Source: **Try it!**



Our Backup Context: **restic**

- <https://github.com/restic/restic>
- Easy, fast and secure
- Repository, **Snapshot** based
- **S3** backend support
- Deduplication, Encryption, tooling for purging of old data
- Verification tools, fuse-mountable repositories



```
restic -r <repo> init    # Initialize repository
restic -r <repo> backup <path> # Backup a folder
restic -r <repo> snapshots # List snapshots
restic -r <repo> restore <snap_id> -t <target> # Restore
```

Backing-Up CephFS

- Backup workloads can be **expensive** at scale
- A Backup client behaves like a "bad user"
 - It regularly attempts to read a large portion of the filesystem
 - On distributed filesystems, this creates many **metadata-server requests**.
- Backup clients may compete with normal clients for **cache** and metadata resources -> **Anything could be improved here Ceph-Side?**
- Most backup runs do not require scanning the entire filesystem
 - Often, **only a small subset of the data has actually changed**

Backing-Up CephFS

- **Only inspect what may have changed**
 - Traditional approach: traverse the entire tree
 - Desired approach: identify unchanged subtrees before traversing them
 - POSIX timestamps provide some change information
 - **mtime**: modification time
 - **ctime**: metadata change time
 - Problem: changes inside child directories are not propagated to the parent directory

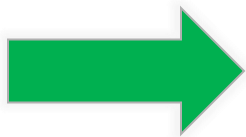
Recursive Change Tracking with CephFS

- **CephFS provides the required information**
 - CephFS exposes a recursive change timestamp through:
 - **ceph.dir.rctime**
 - The attribute is available on directories
 - It represents the timestamp of the latest change in the directory tree

```
getfattr -n ceph.dir.rctime <path>
```

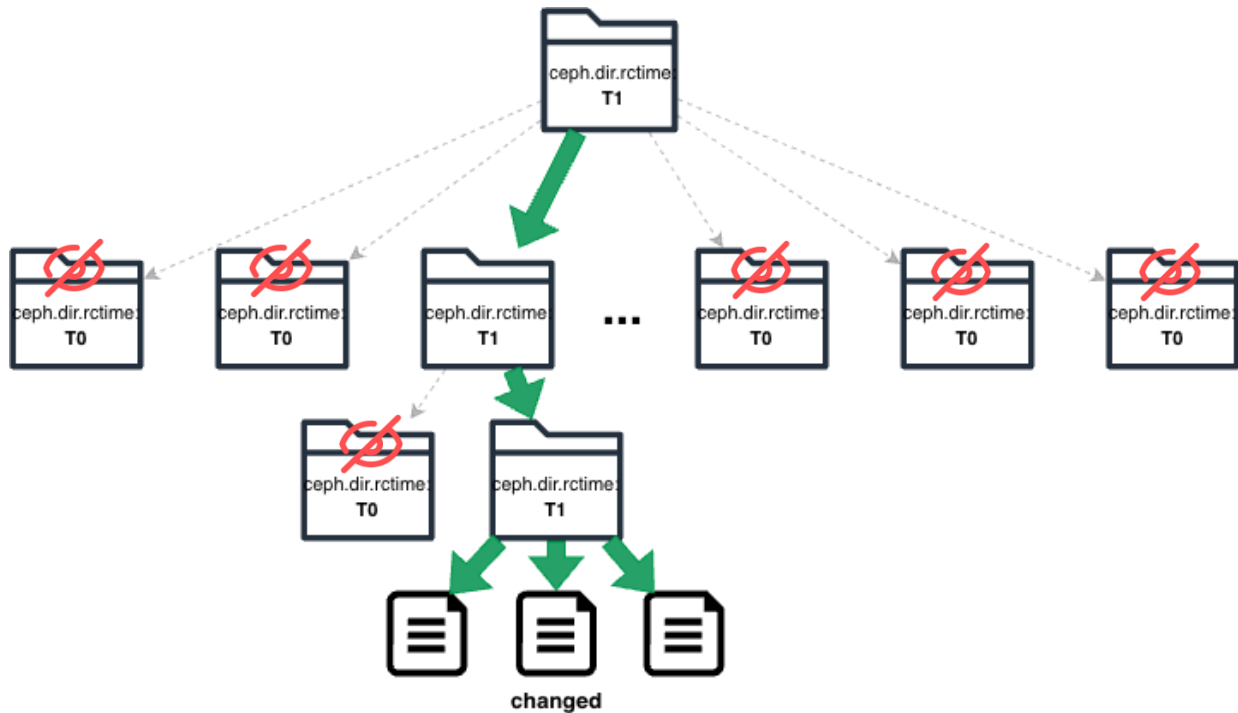
From Skipping Backups to Skipping Subtrees

- **First optimization:**
 - If the top-level directory is unchanged:
 - Skip the entire backup
 - Useful for daily backups when no changes occur (weekends)
- **Remaining Limitation:**
 - One change anywhere in the tree triggers a full traversal



Modified **restic** to compare **ceph.dir.rctime** attribute between backups and to not enter a directory if hasn't changed

RESTIC: Skip Unmodified Trees



RESTIC: Skip Unmodified Trees: How-To

- **How to Use:**

```
restic backup --compare-xattr ceph.dir.rctime
```

- **Extended attributes specified via:**

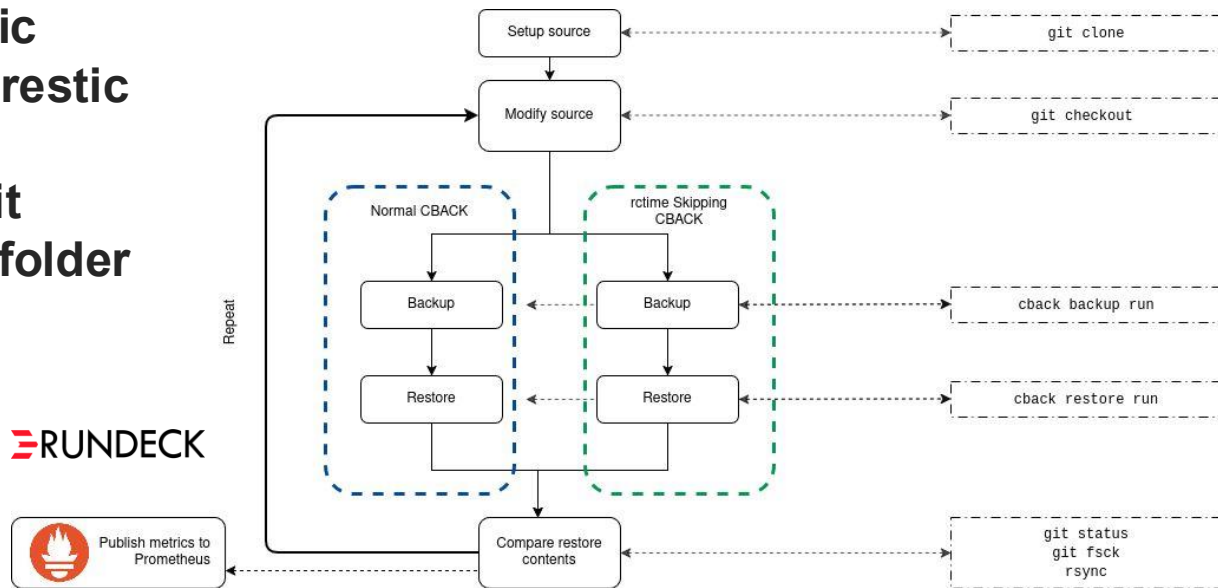
- Via `OS_XATTR="ceph.dir.rctime;ceph.dir.rbytes"`

- **It should work with other filesystems that exposes that information**

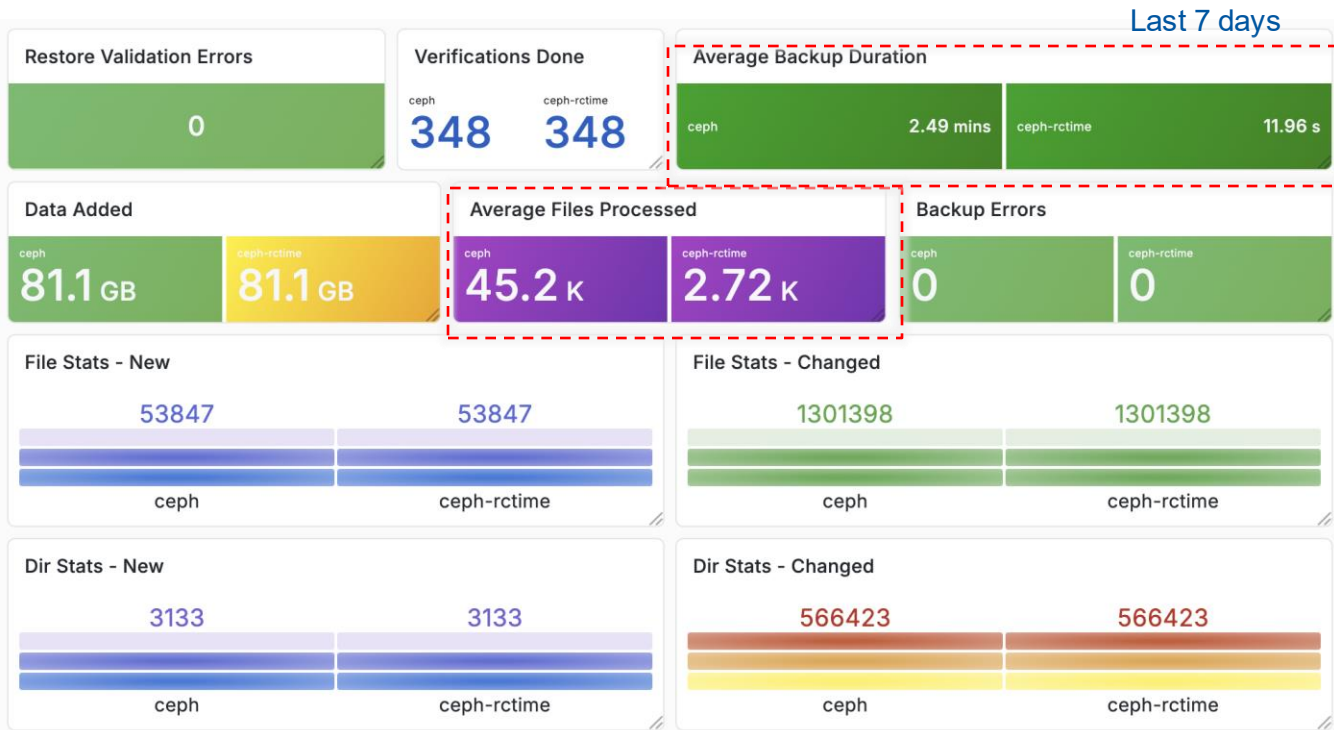
- `OS_XATTR="eos.ttime"`

RESTIC: Skip Unmodified Trees: Validation

- Deployed an automatic validation of the new restic binary
- Using Linux Kernel git repository as source folder



RESTIC: Skip Unmodified Trees: Validation

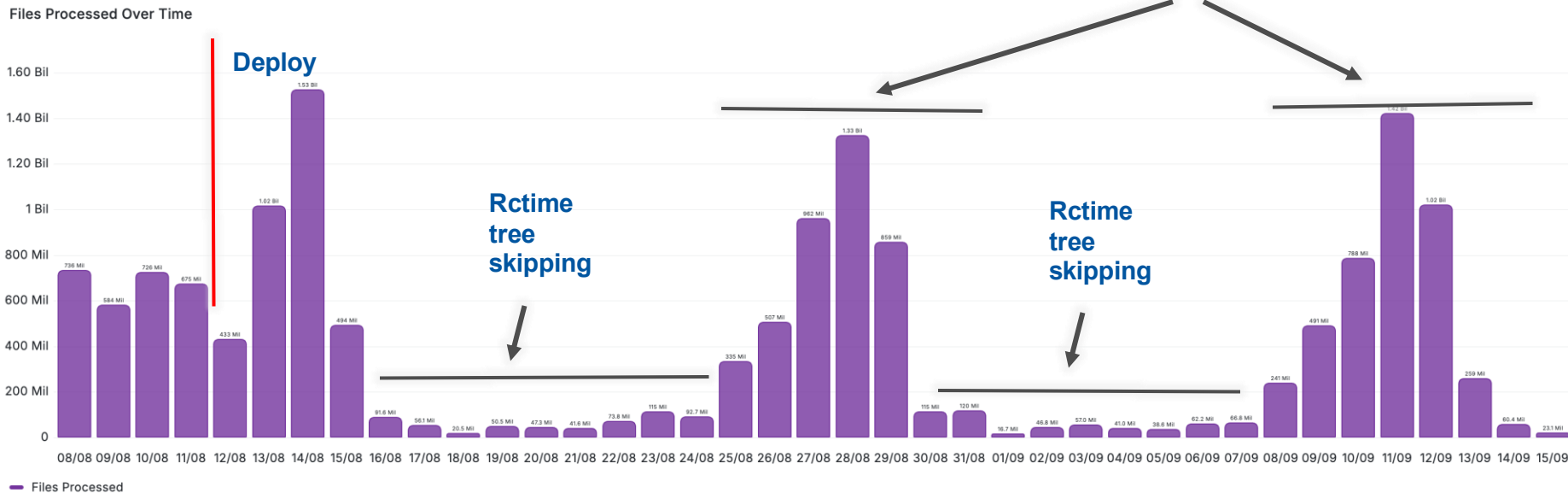


14X Faster
Backup
Duration

94% Less
Files
Processed
("Stated")

Results and Production Impact – File "Stats"

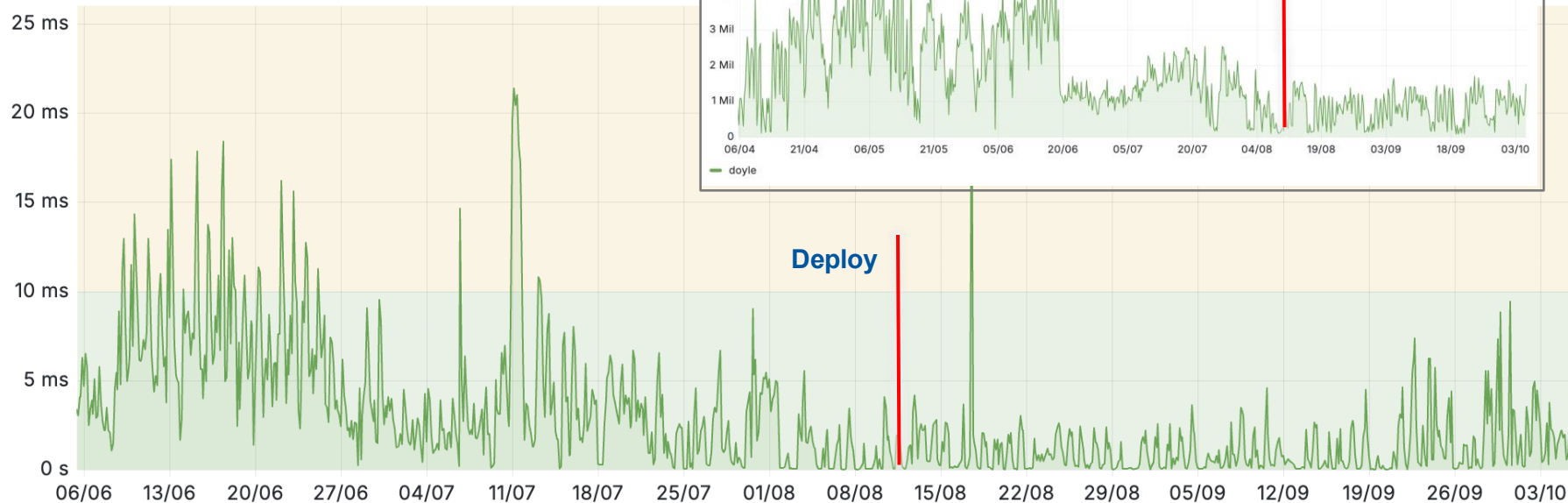
"full" = no rctime skipping = vanilla restic



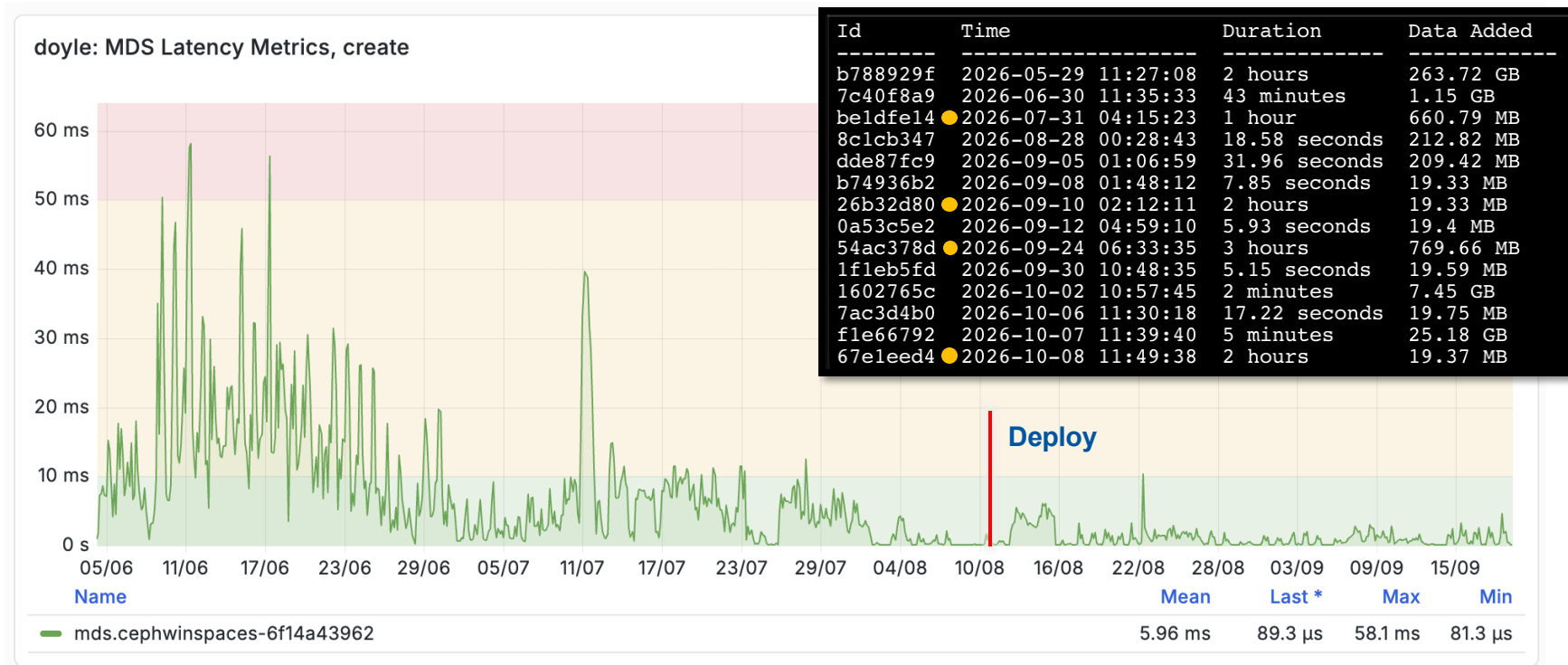
Files Processed = Files Stat()ed

Results and Production Impact – MDS Latency

doyle: MDS Latency Metrics, getattr



Results and Production Impact – MDS Latency



Caveats

- **Getting extended attributes adds latency**
 - Implemented parallelization while getting extended attributes on **restic**
- Helps with average use cases, **not pathological "daily overwrite"**
- **Recursive extended attributes are lazily updated and not totally reliable**
 - Is there any app "lying"? -> Updating content but not mtime
 - Motivation to perform a "full" pass from time to time
- **Restic statistics get messed-up**
 - Scanner statistics are not useful anymore as not all trees are traversed
-> Fixed it on CBACK side

Backing up CephFS: Other methods

- Is there a better way?
 - Backing up **Snapshots**
 - Same problematic as shown
 - Backing up with **Snapshots + SnapDiff**
 - We haven't tested this
 - **inotify** / similar approach for CephFS?
 - Any experience so far?

Conclusion

- **Running in production since +2 months**
 - So far so good
- **Next Steps**
 - Upstream the change to restic
 - A pull request is already available [#22063](#)
- **Give it a try**
 - The CERN fork can be used for testing
 - Current branch: [v0.19.1.rctime](#) [[binary amd64](#)]
 - Interested to see your experience!

restic operational tips on "backup" slides



Thank you! - Q/A

We Are Hiring!

<https://careers.cern/jobs/it-sd-gss-2026-96-ld/>





Backup Slides

RESTIC: Experience, Best Practices and Contributions

- Restic considers host+path as a different backup
 - Not ideal while backing-up a shared filesystem from different nodes
 - Used option ``restic backup --parent`` to link to the previous snapshot
- The previous also applied for retention policy
 - "`restic forget --group by path`" allows to apply retention to path, independently of the host that performed the backup.
- **Unstable inodes** -> Full backup trigger everytime
 - Since restic v0.9.5 an ``restic backup --ignore-inode`` flag is available.
- Un-efficient backup on network file systems
 - Fixed in v0.11.0 -> Kudos Hugo Gonzalez
 - 80% backup increase speed on big repositories (~850K files)

RESTIC: Experience, Best Practices and Contributions

- **Restic cache** can be big (up to 1%) of total repository size
 - Running with no cache is not an option -> Slow, a lot of pressure in backup backend (S3)
 - But it's quick to re-build -> Consider having disposable cache (clean after)
- **Restic scanner** thread (used for statistics) impacts backup speed on mounted filesystems
 - Can be disabled since version `0.15.0`: `restic backup --no-scan``
- **Restic restore --verify** requires all data to be restored on disk
 - A lot of local storage required, difficult to scale the system
 - Migrated to use `restic check --read-data|--read-data-subset=x.y%/K/M/G/T``
- **Restic v2 repository** adds data compression
 - All repositories migrated in 2025 with a total save of 2PB of data (~20% compression rate)

CBACK: Architecture and Design

