Code health in EOS

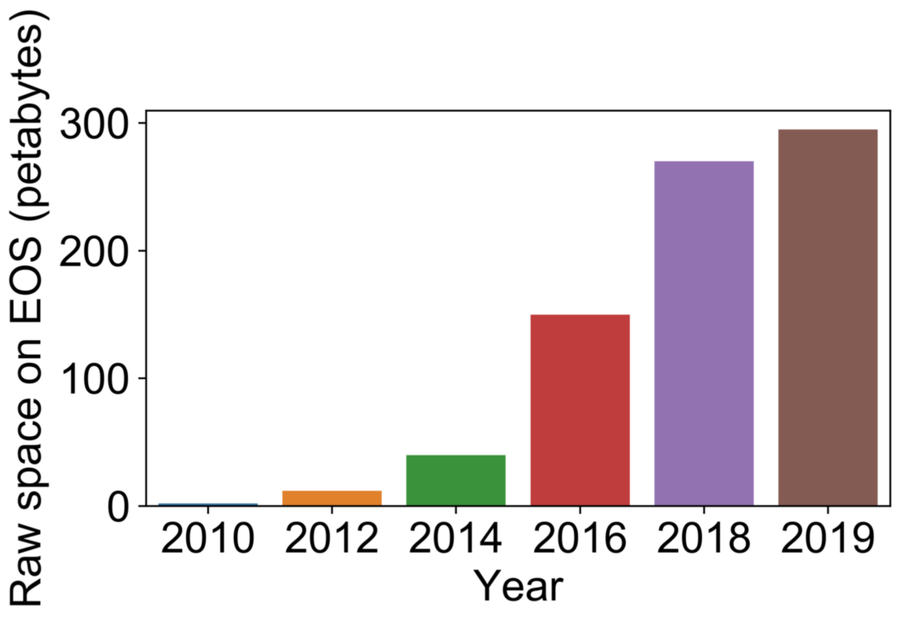
Improving test infrastructure and overall service quality

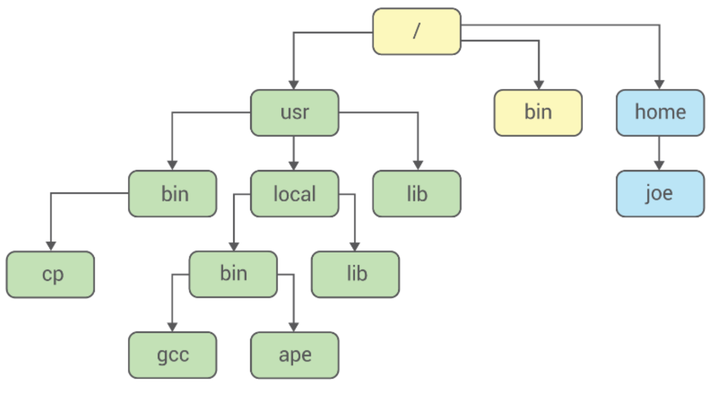
Elvin Alin Sindrilaru, Georgios Bitzes, Fabio Luchetti, Mihai Patrascoiu

Introduction

EOS, the distributed storage system developed and deployed at CERN, recently reached the milestone of **300 PB** in total deployed raw disk space, storing approximately **5 billion files** across more than **50.000** physical hard drives.

This has brought the unwelcome side-effect of stretching the EOS software stack to its design constraints, resulting in frequent user-facing issues and occasional downtime of **critical** services.





Design limitations

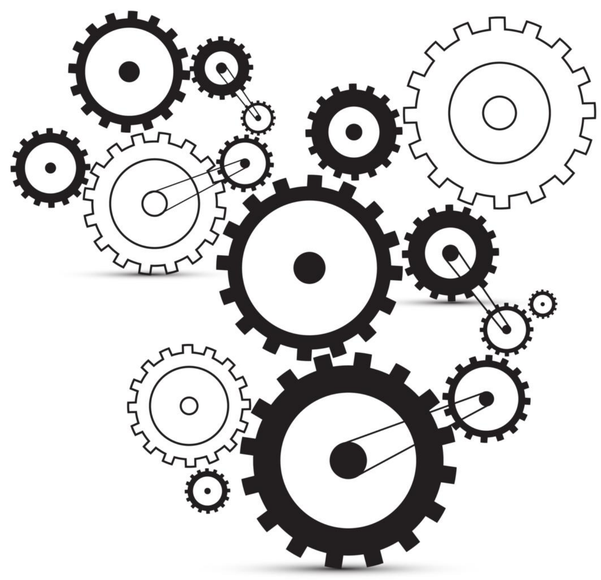
The most significant issue affecting EOS **uptime** has been its in-memory namespace implementation. The metadata server had to scan through the entire namespace contents during boot, taking up to **1 hour** for large instances.

This greatly amplified the disruption caused by occasional server crashes and upgrades, as a simple process restart would take very long. The **new namespace** based on QuarkDB has now been deployed across all EOS instances, making metadata server restart virtually **instant**.

Accumulated technical complexity

Over the past several years of its operation, EOS has received many feature requests both from end-users as well as operators, causing an accumulation of code complexity and some difficulty in ensuring a **reliable service** in face of heavier and heavier production traffic.

To counter these effects, a refactoring effort is being undertaken with the goal of significantly simplifying the C++ codebase and increasing test coverage.



On-going testing strategy

Comprehensive testing at the development stage will enable both a **lower incidence** of user-facing issues, as well as a more **robust development process**. The cornerstones of our testing strategy are **Gitlab CI** and **Kubernetes**, allowing to spawn virtual clusters and orchestrate demanding test scenarios on a nightly and weekly schedule.

- **Unit tests** to validate individual functions and classes.
- **Functional tests** to validate entire subsystems, such as namespace or inter-cluster messaging.
- **AddressSanitizer tests** for detecting hard-to-identify memory corruption bugs.
- **Stress tests** for validating cluster stability under heavy load.
- **Performance tests** to measure end-to-end performance and overall system scalability.
- **Coverity scanning** for static code analysis.

https://eos.web.cern.ch
https://gitlab.cern.ch/dss/eos
https://gitlab.cern.ch/eos/quarkdb
https://gitlab.cern.ch/eos/eos-on-k8s