TECHNICAL WHITEPAPER : APEIRORA ARCHITECTURE BLUEPRINT

Running Rook at Petabyte Scale Across Multiple Regions

Deploying a Digitally Sovereign Storage Backbone for Europe

Executive Summary & The 120-Petabyte Challenge

When operating modern cloud infrastructures at global scale, manual administrative operations quickly cease to scale. The target for this joint engineering initiative was to build a system capable of managing 120 Petabytes of storage across 30 global regions.

Historically, SAP Cloud Infrastructure relied on a combination of proprietary storage appliances and legacy systems like OpenStack Swift. However, as the teams architected the next-generation cloud stack—developed internally as part of the ApeiroRA project—we faced a fundamental, non-negotiable constraint: Digital Sovereignty. The entire infrastructure stack had to be completely free of hyperscaler vendor lock-in, running on dedicated hardware within our own data centers.

This created a concrete engineering challenge for the joint SAP and CLYSO teams: build a storage layer that is API-first by design, fully open-source, and capable of automated self-management at a global scale. We selected Ceph as the storage engine and Rook as the cloud-native automation layer that makes it manageable.

01. Strategic Technology Choice: Why Rook?

Managing Ceph at this scale without a dedicated Kubernetes operator would mean building and maintaining highly complex, custom internal tooling. The real alternatives would have required evaluating entirely different storage architectures or relying on non-Kubernetes-native tools like ceph-adm (the default Ceph orchestrator) to handle daemon placement, upgrades, and recovery. However, this would have fractured our operational model.

Rook gives our engineers all of this as a declarative, Kubernetes-native interface. This means that existing GitOps and CI/CD workflows extend naturally directly down to the storage tier. Instead of writing and maintaining region-specific manual runbooks, we write standardized Helm values.

Historically, SAP Cloud Infrastructure relied on a combination of proprietary storage appliances and legacy systems like OpenStack Swift. However, as the teams architected the next-generation cloud stack—developed internally as part of the Apeiro project—we faced a fundamental, non-negotiable constraint: Digital Sovereignty. The entire infrastructure stack had to be completely free of hyperscaler vendor lock-in, running on dedicated hardware within our own data centers.

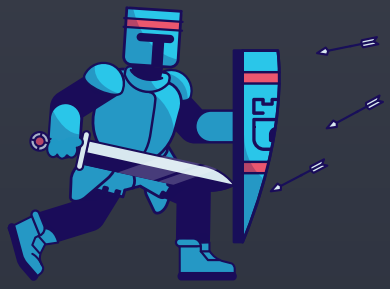
This created a concrete engineering challenge for the joint SAP and CLYSO teams: build a storage layer that is API-first by design, fully open-source, and capable of automated self-management at a global scale. We selected Ceph as the storage engine and Rook as the cloud-native automation layer that makes it manageable.

01. Strategic Technology Choice: Why Rook?

Managing Ceph at this scale without a dedicated Kubernetes operator would mean building and maintaining highly complex, custom internal tooling. The real alternatives would have required evaluating entirely different storage architectures or relying on non-Kubernetes-native tools like ceph-adm (the default Ceph orchestrator) to handle daemon placement, upgrades, and recovery. However, this would have fractured our operational model.

Rook gives our engineers all of this as a declarative, Kubernetes-native interface. This means that existing GitOps and CI/CD workflows extend naturally directly down to the storage tier. Instead of writing and maintaining region-specific manual runbooks, we write standardized Helm values.

02. Architectural Framework: The Separation of Metal and Software



Our platform, CobaltCore, is built on top of Gardener and the Metal-API, both of which are integral components of the ApeiroRA reference architecture. Within this design, storage is not treated as a static resource, but rather as a programmable Kubernetes object.

For performance and stability reasons, we run our storage infrastructure on dedicated nodes that are strictly isolated from actual application workloads. Given our high hardware density (16 NVMe drives per node), co-locating workloads would introduce unpredictable and unacceptable I/O interference, so our storage nodes do one thing: serve data.

2.1 THE PHYSICAL METAL LAYER

The Metal-API independently manages the entire physical lifecycle of our bare-metal servers, including hardware inventory, provisioning, firmware management, and operating system deployment. Once the physical foundation is ready, Gardener takes over to provision and manage the production Kubernetes clusters on top of these hardware nodes. This clean separation completely decouples physical state management from the software layer.

2.2 THE DECLARATIVE STORAGE LAYER (ROOK)

Rook operates natively within the cloud-native stack. Its lifecycle begins once the Kubernetes cluster has been fully provisioned by Gardener and the Metal-API. Installed directly into the Kubernetes cluster, Rook acts as the operator that installs, configures, and dynamically operates Ceph from the inside out.

We use a strict GitOps workflow to ensure architectural consistency across the entire fleet:

Base Blueprint

A central Helm chart defines global best practices and standardized Ceph configurations.

Region Overlay

Region-specific configurations (such as CephBlockPools and RGW placement rules) are injected via localized values.yaml files.

Automation

Rook autonomously handles daemon bootstrapping, configuring CRUSH failure domains, and provisioning Rados Gateway (RGW) endpoints.

2.3 STANDARD STORAGE NODE SPECIFICATION

Component	Specification / Technical Details
Server Model	Dell PowerEdge R7615
Processor (CPU)	AMD EPYC 9554P (64 cores)
Memory (RAM)	384 GB
Storage Population	16x 14 TB NVMe drives
Network Connectivity	100 GbE (fully redundant design)

03. Validation: Establishing the Performance Envelope

Before committing to our final production capacity planning, we needed to establish the exact performance envelope of our RGW tier (Object Storage). We executed a breakpoint test on a representative Ceph Squid cluster (consisting of 28 nodes and 362 OSDs) to empirically determine the stable operating range, saturation threshold, and absolute hard ceiling.

3.1 TEST SETUP & METHODOLOGY

Workload Profile	2 million objects (4 KB each), simulated by 20 parallel k6 test clients targeting a dedicated "premium" NVMe bucket.
Methodology	Ramping up the load linearly over a 30-minute window until p90 latency exceeded the defined threshold of 500 ms (Latency Exit Condition).

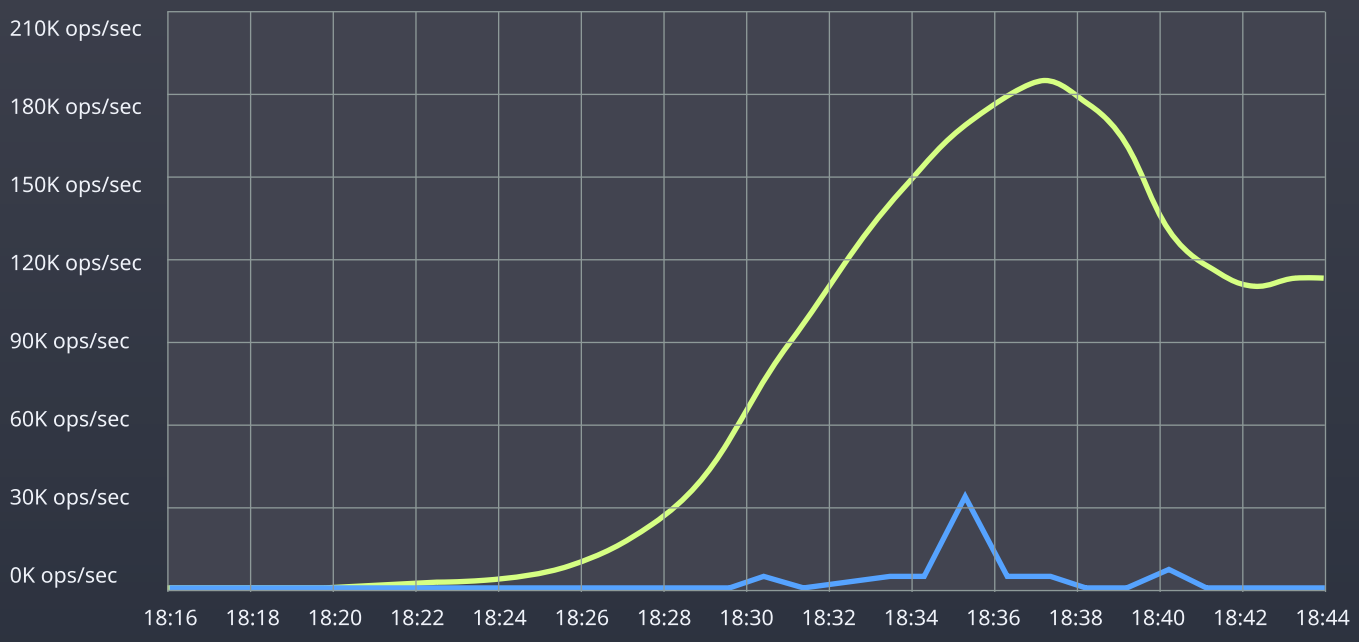
3.2 TEST RESULTS & SATURATION ANALYSIS

The cluster entered its stable saturation threshold at approximately 90,000 GET requests/sec.

Beyond this point, latency percentiles began to diverge noticeably, and internal request queues started accumulating.

The absolute maximum performance peaked at 171,000 GET requests/sec (measured directly at the RGW tier) before our runners hit the latency exit condition.

> Critical Engineering Observation: Isolated HTTP 503 errors appeared as early as ~33,000 GET requests/sec on single RGW instances. This was caused by uneven load distribution across the tier rather than global cluster-wide saturation.



NAME	AVG	LAST	MIN	MAX
failed	1.99K ops/sec	290 ops/sec	0 ops/sec	31.1K ops/sec
reqs	67.1K ops/sec	103K ops/sec	2.18 ops/sec	171K ops/sec

3.3 ANALYSIS OF LATENCY DIVERGENCE (CLIENT- VS. SERVER-SIDE VIEW)

Comparing client-side metrics against internal RGW metrics provides crucial telemetry insights for our engineering teams: At peak load, the RGW backend reported a p99 latency of only ~210 ms, while our k6 clients simultaneously observed a latency of over 1.5 seconds.

This significant delta is caused by Connection Queueing. Requests sit in the queue before they are picked up by the RGW Beast frontend threads. Internal RGW metrics only measure the processing time after a request is accepted, omitting the time spent waiting in the queue.

Additionally, RADOS operation latency increased under heavy load, keeping RGW threads occupied longer and accelerating queue accumulation. At the breaking point, the queues filled completely, and RGWs began returning cascading 503 errors across all instances.

Because our primary workload is heavily read-centric (read-heavy), the saturation threshold of 90,000 GET requests/sec provides our teams with a highly reliable and conservative operating ceiling for regional capacity planning.

04. Operational Reality: Making "Day 2" Operations Boring

The true test of any production storage system is what happens when things break or need upgrading. Our joint engineering goal at this scale is to make operations entirely predictable and boring.

4.1 ZERO-DOWNTIME UPGRADES

Rook has successfully reduced storage maintenance from a coordinated, high-risk event to a routine background task. Since the first cluster went live in May 2024, we have maintained a continuous upgrade cadence with zero customer-facing downtime and zero data loss:

GardenLinux

Monthly rolling updates deployed across all active regions.

Kubernetes

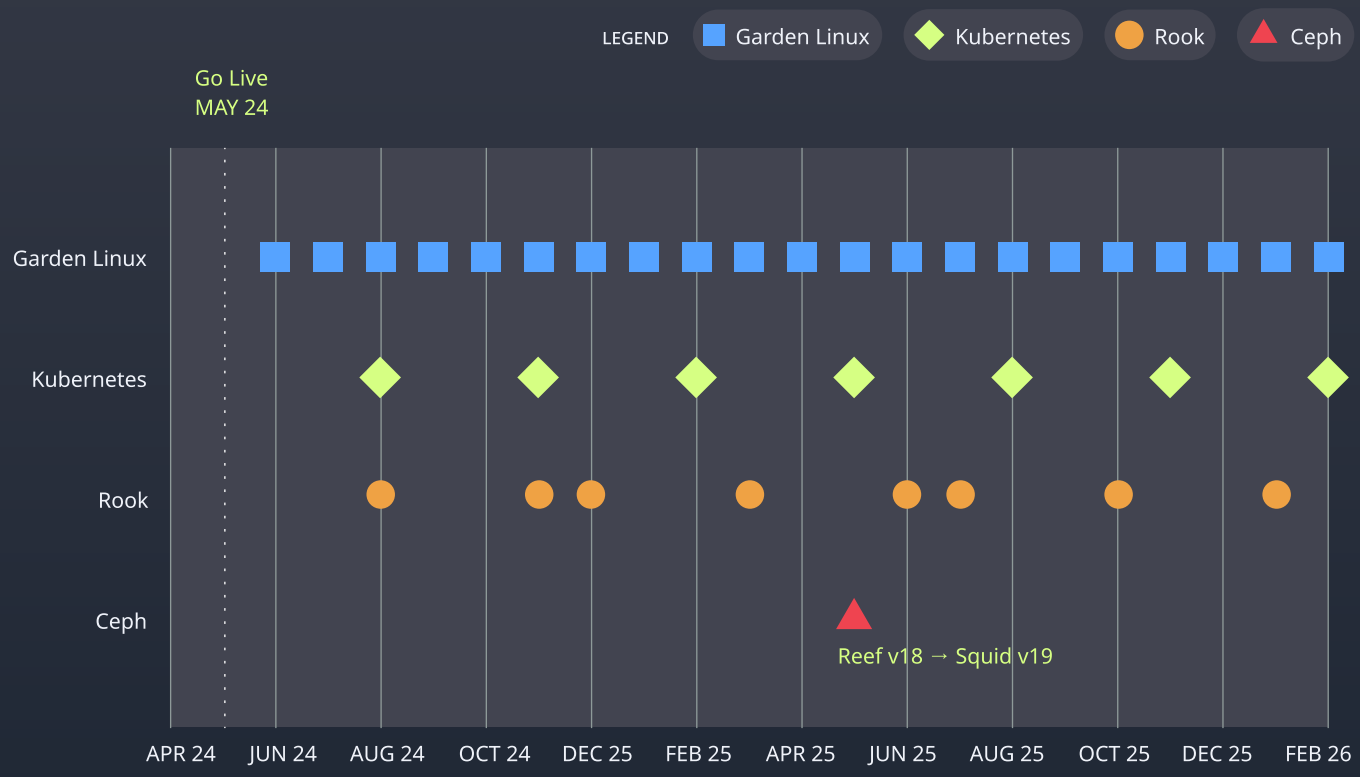
Quarterly minor version upgrades.

Rook

Quarterly upgrades (successfully executed from v1.14 through v1.18), with out-of-cycle updates applied when a needed features ships.

Ceph

Major version migration from Reef (v18) to Squid (v19). A rolling upgrade of our largest single cluster (~816 OSDs) finishes in approximately 2 days, running completely in the background.



Upgrade Cadence

4.2 RESILIENCE TO DRIVE FAILURES

With roughly 2,800 active OSDs currently in our fleet, physical drive failures are statistical routine events. When a drive fails, the Ceph infrastructure (RADOS) automatically handles data recovery and rebalancing across the remaining healthy OSDs, no operator action is required to protect the data.

On the orchestration side, Rook detects the failed OSD pod and manages its software lifecycle. While the full drive replacement cycle still involves manual operational steps on our side, data durability is never at risk due to Ceph's inherent self-healing capabilities.

05. Current Status and Strategic Roadmap

As of early 2026, our production storage fleet spans 10 live regions, with an 11th region newly provisioned:

251 Nodes
EMERGENCY SERVICE

~2,800 OSDs
ACTIVE OSDS IN FLEET

~37 PiB raw
TOTAL PROVISIONED CAPACITY

Our regional deployments scale flexibly from small footprints (13 nodes / 96 OSDs) up to large clusters (59 nodes / 816 OSDs). Every variation is managed uniformly via the exact same Rook-based GitOps workflow.

NEXT MILESTONES

Our next phase is onboarding high-performance Block Storage (RBD) into this declarative model to fully deprecate and retire our remaining proprietary SAN hardware. The final strategic target remains unchanged: 30 regions with a total capacity of 120 PB.

Conclusion

Our joint collaboration on this infrastructure is a core part of ApeiroRA, an open initiative developing a sovereign reference blueprint for cloud-edge infrastructure. Both the SAP and CLYSO engineering teams are active contributors to the Rook project, and we continue to collaborate closely with the maintainers as we scale toward our goals. All architectural components utilize enterprise-friendly open-source licenses under neutral community governance.

Shape the Future of Sovereign Cloud Infrastructure with Us

Are you running large-scale cloud-native storage or facing similar multi-region scale challenges? We welcome you to collaborate with us.

Explore the Tech: Visit our documentation portal to dive deeper into the ApeiroRA reference architecture blueprints.
Join the Community: Connect with our engineering teams and the wider community in the Rook Slack to exchange ideas, contribute components, or share operational runbooks.

Scan this QR code to read the original operational blog post and explore the detailed performance charts and community discussions.

CLYSO

Let's architect your sovereign storage solution together.

+49 89 215252730 sales@clyso.com www.clyso.com

We are proud members of

NeoNephos · The Linux Foundation · The Linux Foundation Europe · kcp · Cloud Native Computing Foundation · ceph