MINIO

High Performance Object Storage

WHITE PAPER

Executive Summary

MinIO is a high performance, distributed object storage system. By following the methods and design philosophy of hyperscale computing providers, MinIO delivers superior performance, resilience and scalability to a wide variety of workloads in the private cloud, public cloud, Kubernetes distributions and edge. This makes MinIO the standard for hybrid cloud architectures.

While MinIO is ideal for traditional object storage use cases like secondary storage, disaster recovery and archiving, it truly excels in overcoming the challenges of delivering massive primary storage across a range of use cases from Kubernetes-powered cloud applications to AI/ML/advanced analytics workloads.

Because MinIO is purpose-built to serve only objects, a single-layer architecture achieves all of the necessary functionality without compromise. The advantage of this design is an object server that is high-performance and lightweight: efficient enough to run in a container and powerful enough to become the core of a Kubernetes-managed object storage as a service platform.

MinIO is a pioneer in the development of cloud-native object storage, refining and perfecting many of the features, protocols and APIs that have come to define best in class. This is evidenced by the more than 530M Docker pulls, 26K+ GitHub stars and thousands of production deployments across every continent.

This paper details the philosophical approach and technical attributes of MinIO and why those attributes are important to any enterprise seeking to develop or migrate to an object storage centric, microservices architecture across the public and private cloud.

The Enterprise Challenge

How enterprises store, access, move and analyze data is undergoing massive change. Driven by the storage and compute efficiencies made possible by disaggregation, enterprises are finding that their investments in traditional storage solutions like Hadoop HDFS are now obsolete. The secret of elite hyperscalers, disaggregation, offers multiple benefits, but the two largest are economics and performance. As a result, enterprises are rearchitecting their data infrastructures to take advantage of this separation.

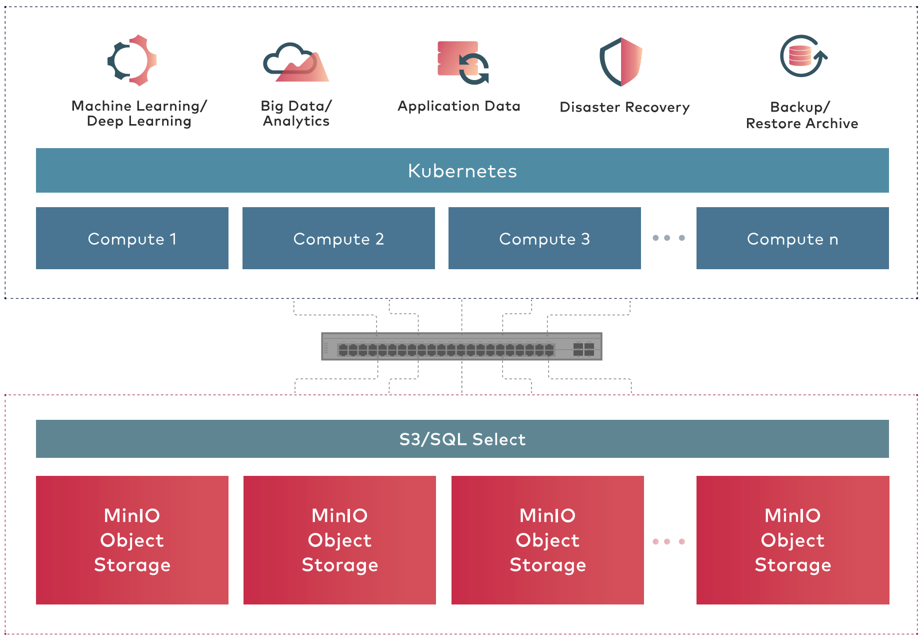


Figure 1: The modern, disaggregated architecture

The reasons are straightforward. File and block protocols are complex, have legacy architectures that impede innovation, are limited in their ability to scale or are compromised from a performance perspective. Examples of these limitations include the aforementioned aggregation of compute and storage but also include replication, security, encryption and data mobility.

The winner in this transformation is cloud-native object storage.

Storage as a Service or STaaS is the second-fastest growing cloud workload worldwide, representing a USD 19.9 billion annual market in 2020 and projected to grow to USD 101.9 billion in 2027. Data is growing exponentially every year and by 2025, experts predict that the world will create and replicate 163 zettabytes (ZB) of data. The vast majority of that will be unstructured or semi-structured.

Fueling that growth is a focus on big data applications, primarily analytics and artificial intelligence (AI) workloads on Internet of Things (IoT) and other event data. These workloads demand high rates of throughput, excellent data integrity and a cost-effective deployment model.

Simple, powerful and with unlimited scalability, modern object storage has moved out of backup and into the application and analytic workflow. A reduced set of storage APIs, accessed over HTTP RESTful services mean that these cloud-native solutions are lightweight enough to be packaged with the application stack to be run in containers orchestrated by Kubernetes.

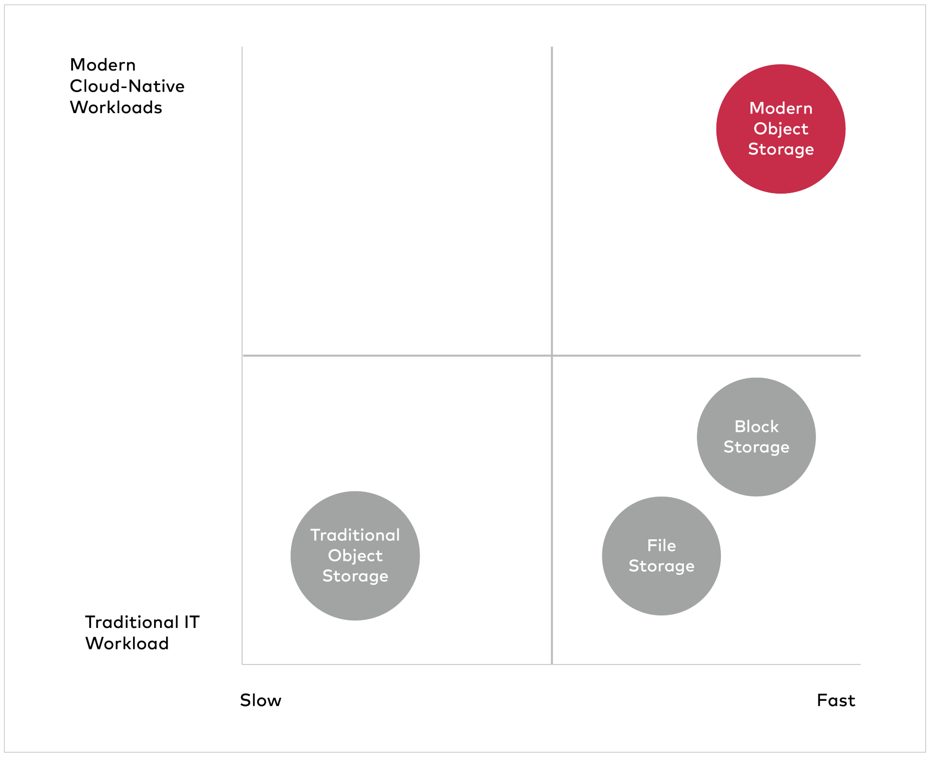


Figure 2: The advantages of modern object storage

The Philosophy Of The Cloud

MinIO combines the inherent advantages of object storage with a robust suite of features, a stunningly simple, intuitive interface and an expansive set of integrations.

MinIO is unique in that it was built from the ground up with cloud-native technologies to be simple, fast, durable and highly scalable. With the belief that a complex solution cannot be scalable, a minimalist design philosophy forms the foundation of the MinIO architecture.

The result is a system that excels across several key dimensions:

- **Performance**. With its focus on high performance, MinIO enables enterprises to support multiple use cases with the same platform. For example, MinIO's performance characteristics mean that you can run multiple Apache Spark, Presto/Trino and Apache Hive queries, or to quickly test, train and deploy AI algorithms, without suffering a storage bottleneck. MinIO provides a high-performance object-storage back end to streaming data analytics. MinIO object storage is used as the primary storage for cloud native applications that require higher throughput and lower latency than traditional object storage can provide.

- **Scalability**. A design philosophy that "simple things scale" means that scaling starts with a single pool (an independent set of compute, network and storage resources) which can be combined with other MinIO pools to expand the capacity per deployment. In multi-tenant configurations, each tenant is a cluster of server pools that are fully isolated from each other in their own namespaces. Expansion of the namespace is achieved by adding more clusters and racks within a data center, at the edge, or in a public, private or hybrid cloud.

- **Simplicity**. Minimalism is a guiding design philosophy at MinIO. Simplicity reduces opportunities for errors, improves uptime and delivers reliability while serving as the foundation for performance. MinIO can be installed and configured within minutes from our intuitive graphical user interfaces or our simple command line interface. The amount of configuration options and variations is kept to a minimum, resulting in near-zero system administration tasks and very few paths to failure. Upgrading MinIO is done with a single command which is non-disruptive and incurs zero downtime - lowering total cost of ownership.

Collectively, these philosophical pillars enable MinIO to seamlessly deliver multi-instance, multi-tenant object storage across any hardware and to any workload. That means MinIO can run anywhere Kubernetes runs, from VMware's Tanzu to AWS itself and everywhere in between.

High Performance Object Storage

Every feature of MinIO's object storage suite was architected to deliver performance, scale and resiliency. As a software-defined solution, MinIO can be paired with hundreds of different compute and storage configurations from Intel Cascade Lake, ARM Graviton, or Atom processors on the compute side to Optane and NVMe SSDs and traditional spinning disk.

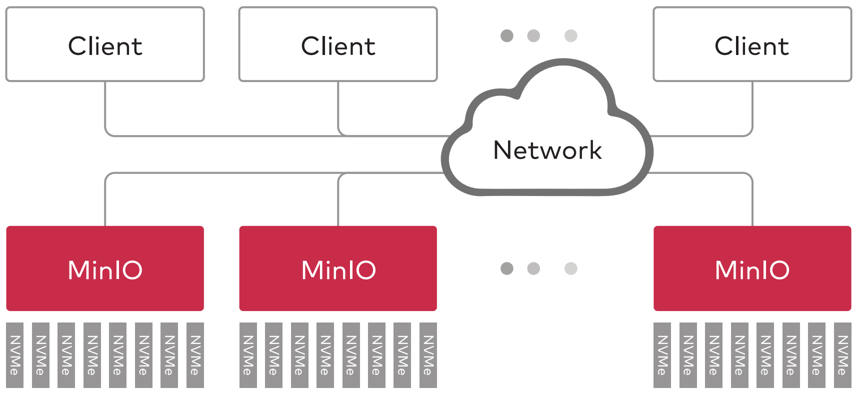


Figure 3: A typical MinIO deployment.

MinIO's software defined object storage suite consists of a server, and optional components such as a client, a management console, a Kubernetes Operator and Operator Console and a software development kit (SDK):

MinIO Server

MinIO is a distributed object storage server. It boasts the most comprehensive implementation of the Amazon S3 API to be found anywhere outside of Amazon itself. MinIO is feature-complete, providing enterprise-grade encryption, identity management, access control, and data protection capabilities, including inline erasure code, bitrot protection, immutability, active-active replication and other features.

MinIO Client

Called mc, the MinIO Client is a modern and cloud-native alternative to the familiar UNIX commands like ls, cat, cp mirror, diff, find and mv. This client provides advanced functionality that is suitable for web-scale object storage deployments. For example, powerful data replication tools work between multiple sites for HA (highly availability) and DR (disaster recovery) purposes and support generating shared, time-bound links for objects. Further, extensive scripting capabilities enable automation for DevOps teams.

MinIO Console

The MinIO Console is a browser-based GUI that incorporates all the functionality of MinIO Client in a design that feels familiar for IT admins and developers alike. Built to support cloud-scale deployments with minimal operational overhead, MinIO Console enables administrators and users to provision multi-tenant object storage as a service, visually inspect the health of the system, perform key audit tasks and simplify integration (via webhooks and API) with other components.

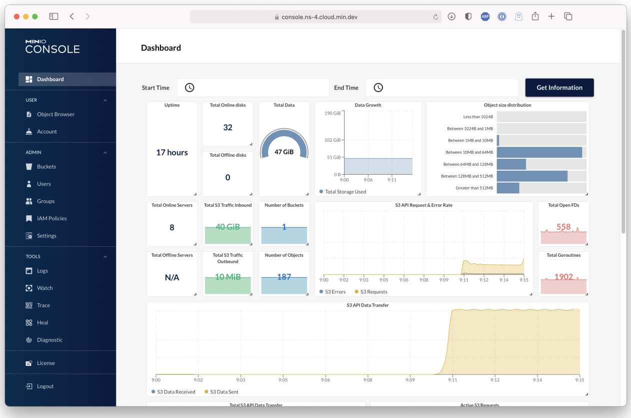


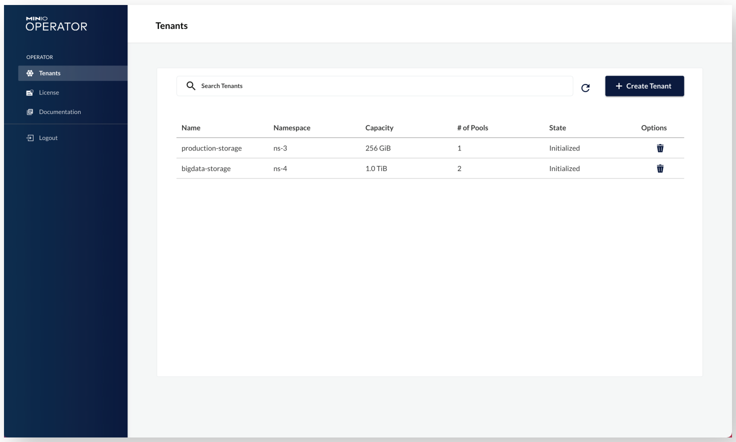
Figure 4: MinIO Console streamlines object storage operations with a full-featured GUI.

MinIO Kubernetes Operator

MinIO Kubernetes Operator adds a plugin to the Kubernetes CLI that allows DevOps teams to deploy and manage MinIO object storage on Kubernetes. A straightforward list of commands makes it easy to execute MinIO's key capabilities. Actions can be scripted and automated, making it easy to deploy and consume object storage within the DevOps and Kubernetes-centric world.

Kubernetes Operator Console

MinIO Kubernetes Operator Console provides a graphical user interface (GUI) that makes it even easier to create, deploy and manage muti-tenant, object storage as a service to internal and external stakeholders alike.



Name	Namespace	Capacity	# of Pools	State	Options
production-storage	ns-3	256 GiB	1	Initialized	🗑
bigdata-storage	ns-4	1.0 TiB	2	Initialized	🗑

Figure 5: MinIO Kubernetes Operator Console is an essential component of object storage as a service.

MinIO SDKs

The MinIO Client SDKs provide simple APIs to access any Amazon S3-compatible object storage. MinIO repositories on Github offer SDKs for popular development languages such as Go, JavaScript, .Net, Python and Java.

The features of MinIO's Object Server are notable for their breadth, depth and focus on the enterprise. As a cloud-native implementation, the range of features exceed those in legacy or bolt-on implementations while the attention to engineering-first principles ensure exceptional performance.

Key Features

MinIO only does object storage and as a result, has a broad range of features that are designed to create a persistence layer that is performant, resilient, secure and scalable across the hybrid cloud.

S3 Select

To deliver high-performance access to big data, analytic and machine learning workflows requires server-side filtering features - also referred to as "predicate pushdown".

MinIO has developed a SIMD accelerated version of the S3 Select API which is essentially SQL query capabilities baked right into the object store. Users can execute SELECT queries on their objects, and retrieve a relevant subset of the object, instead of having to download the whole object. With the S3 Select API, applications can now download a specific subset of an object - only the subset that satisfies the given SELECT query. This directly translates into efficiency and performance by reducing bandwidth requirements, optimizing compute and memory resources meaning more jobs can be run in parallel - with the same compute resources. As jobs finish faster, there is better utilization of analysts and domain experts. This capability works for objects in CSV, JSON and Parquet formats and is effective on compressed objects as well.

Erasure Coding

MinIO protects data with per-object, inline erasure coding which is written in assembly code to deliver the highest performance possible. MinIO uses Reed-Solomon code to stripe objects into data and parity blocks - although these can be configured to any desired redundancy level. This means that in a 12 drive setup with 6 parity configuration, an object is striped across as 6 data and 6 parity blocks. Even if you lose as many as 5 ((n/2)–1) drives, be it parity or data, you can still reconstruct the data reliably from the remaining drives. MinIO's implementation ensures that objects can be read or new objects written even if multiple devices are lost or unavailable.

Erasure code protects data without the limitations of RAID configurations or data replicas. For example, RAID-6 only protects against a two-drive failure whereas erasure code allows MinIO to continue to serve data even with the loss of up to 50 percent of the drives and 50 percent of the servers. Replication results in 3 or more copies of the object on each of the sites. Erasure-code offers a significantly higher level of protection while only consuming a fraction of the storage space as compared to replication.

Finally, MinIO applies erasure code to individual objects, which allows the healing at an object level granularity. For RAID-protected storage solutions, healing is done at the RAID block layer, which impacts the performance of every file stored on the volume until the healing is completed.

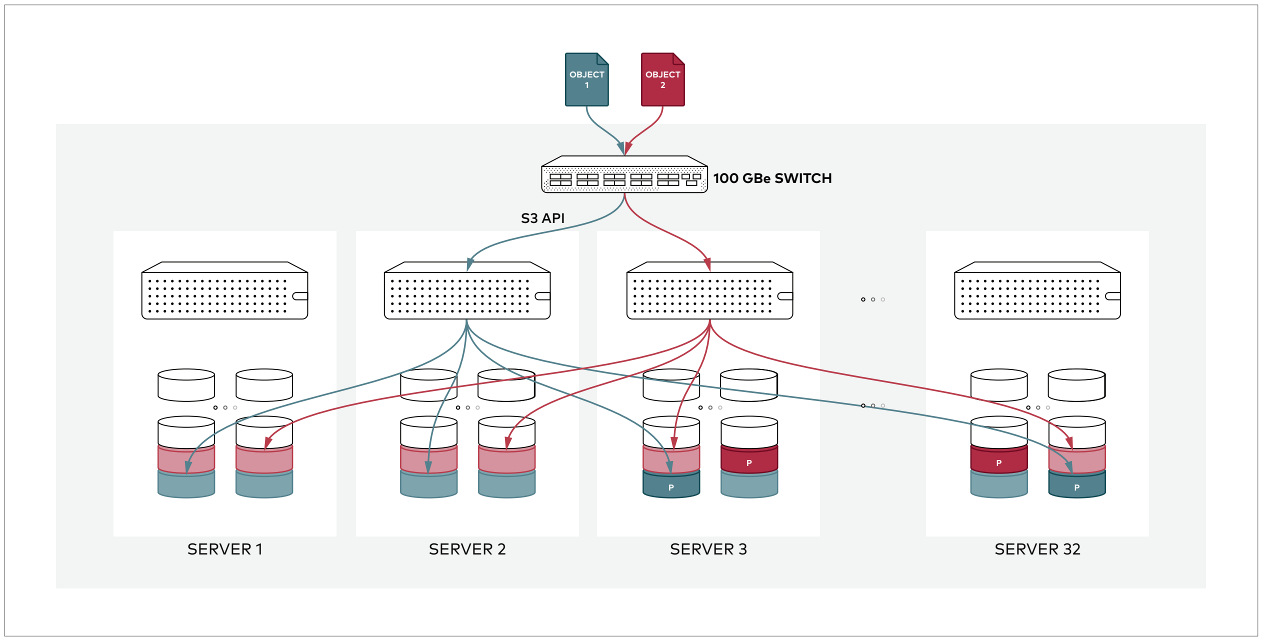


Figure 6: Erasure code protects data without the overhead associated with alternative approaches.

BitRot Protection

Silent data corruption or bitrot is a serious problem for drives resulting in the corruption of data without the user's knowledge. As the drives get larger and larger and the data needs to persist for many years, this problem is more common than we imagine. The data bits decay when the magnetic orientation flips and loses polarity. Even solid state drives are prone to this decay when the electrons leak due to insulation defects. There are also other reasons such as wear and tear, voltage spikes, firmware bugs and even cosmic rays.

MinIO's SIMD accelerated implementation of the HighwayHash algorithm ensures that it will never return corrupted data - it captures and heals corrupted objects on the fly. Integrity is ensured from end to end by computing hash on WRITE and verifying it on every READ from the application, across the network and to the memory/drive. The implementation is designed for speed and can achieve hashing speeds over 10 GB/sec per core on Intel CPUs.

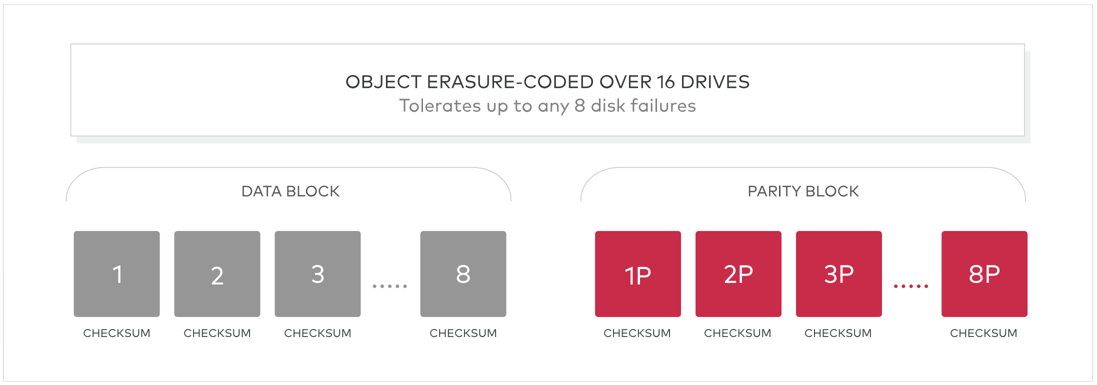

OBJECT ERASURE-CODED OVER 16 DRIVES
Tolerates up to any 8 disk failures
DATA BLOCK
PARITY BLOCK
1
2
3
8
1P
2P
3P
8P
CHECKSUM

Figure 7: MinIO's data protection schemes cover failure and silent data corruption.

Identity and Access Management

MinIO supports the most advanced standards in identity management, offering an interal IAM while also integrating with OpenID connect and LDAP compatible IDP providers.

MinIO's internal IAM approach employs the access key and secret key credential framework. Applications use those credentials to authenticate every time they perform operations on the MinIO cluster. Access policies are fine grained and highly configurable via API, which means that supporting multi-tenant and multi-instance deployments become simple.

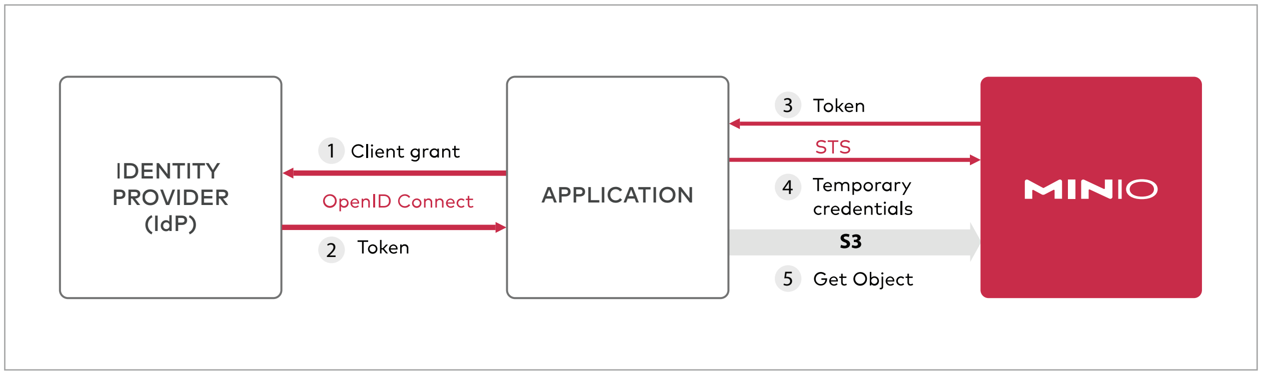


Figure 8: Identity protection and single sign on (SSO) are critical enterprise features.

MinIO also supports leading third-party external identity providers (IDP). These standalone systems specialize in the creation, authentication, and management of user identities. Currently supported vendors include Keycloak, Facebook, Google, Okta, Active Directory and OpenLDAP. In addition to internal and external user identities, the MinIO Console supports the creation of Service Accounts.

On the Access Management side, MinIO controls the authorization of an authenticated application, using AWS IAM-compatible Policy-Based Access Control (PBAC).

Encryption

It is one thing to encrypt data in flight and another to protect data at rest. MinIO supports multiple, sophisticated server-side encryption schemes to protect data - wherever it may be. MinIO's approach ensures confidentiality, integrity and authenticity with negligible performance overhead. Server side and client side encryption are supported using AES-256-GCM, ChaCha20-Poly1305 and AES-CBC. Encrypted objects are tamper-proofed with AEAD server side encryption. Additionally, MinIO is compatible with and tested against commonly used Key Management solutions (e.g. HashiCorp Vault).

MinIO uses key-management-systems (KMS) or cryptographic key management system (CKMS) to support SSE-S3. If a client requests SSE-S3, or auto-encryption is enabled, the MinIO server encrypts each object with a unique object key which is protected by a master key managed by the KMS. Given the exceptionally low overhead, auto-encryption can be turned on for every application and instance.

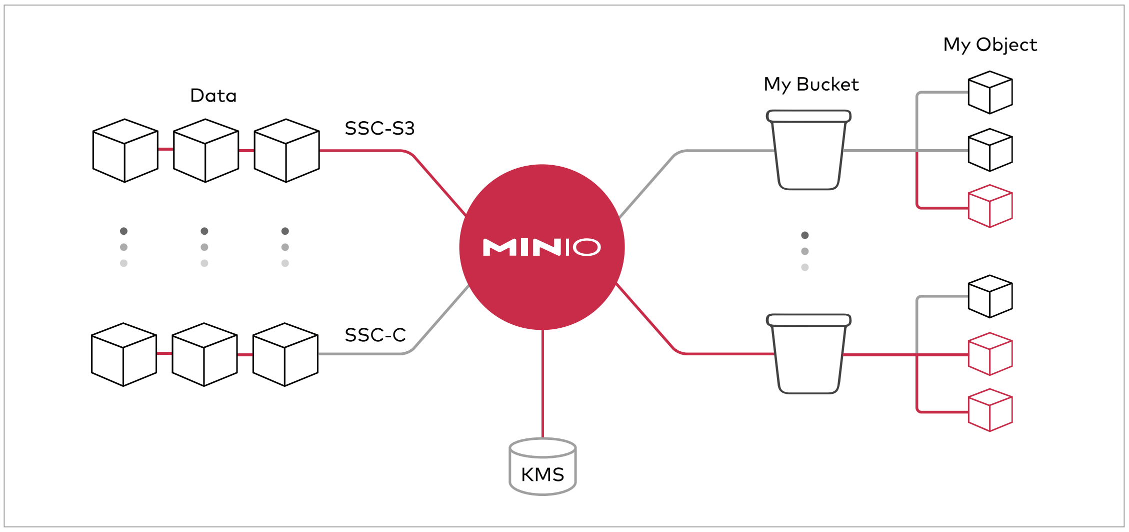


Figure 9: Encryption and WORM protect data in flights and at rest.

Finally, MinIO has introduced its own Key Encryption Service (KES). Stateless and distributed (KES) was designed to be run inside Kubernetes and distribute cryptographic keys to performance oriented applications. KES operates as a bridge between a central KMS and cloud-native applications, as an abstraction layer over different KMS vendors and as a scale-out load balancer for cryptographic operations in distributed systems.

Data Lifecycle Management and Tiering

MinIO lifecycle management tools allow administrators to define how long data remains on disk before being removed. MinIO protects data within and across clouds with a wide range of policies built on object and tag filters to declare expiry rules. Bucket expiration rules are fully compliant with MinIO WORM locking and legal holds. MinIO can programmatically tier objects across storage mediums and cloud types to optimize for performance and cost. MinIO is frequently used as the primary application storage layer within a public cloud. In this use case, applications are written against a MinIO endpoint. As objects age, MinIO may move data based on tiering policy. Tiering is transparent to end users and applications as MinIO continues to serve objects through the original endpoint.

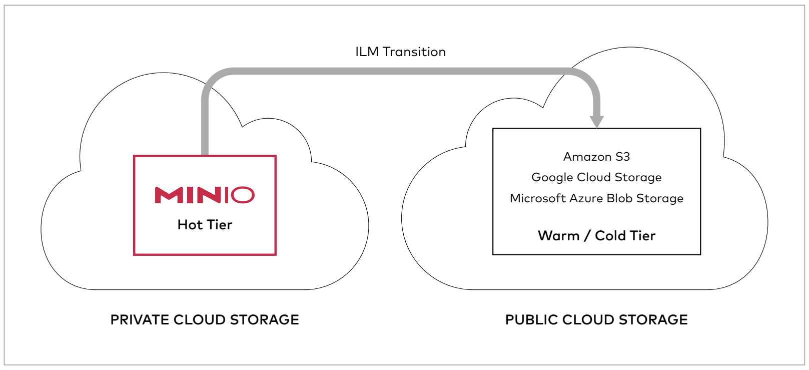


Figure 10: Lifecycle management policies enable tiering of objects across cloud storage.

Bucket and Object Immutability

MinIO supports object locking, retention, legal holds, governance and compliance for objects and buckets. Object locking is frequently combined with versioning to eliminate the risk of data tampering or destruction. Retention rules ensure that an object is WORM protected for a configurable period of time. This capability is critical for ransomware use cases and can be used in conjunction with leading backup vendors to ensure fast backup/restore across multiple workloads. MinIO's implementation earned validation from Cohasset Partners that MinIO meets the requirements of SEC 17a-4(f), FINRA 4511(c) and CFTC 1.31(c)-(d).

Bucket and Object Versioning

MinIO's object-level versioning provides data protection and serves as the foundation for data lifecycle management, tiering and locking. MinIO follows Amazon's S3 structure/implementation to independently version objects. This allows users to retain multiple variants of every object at the bucket level, eliminating the need for a separate snapshot process. Buckets can exist as unversioned, versioning-enabled or versioning-suspended. When a versioned object is deleted it is not removed permanently. Instead, a delete marker is created and when that version of the object is requested MinIO returns a 404 Not Found message.

Scalability

MinIO scales out, or horizontally, through server pools. Each server pool is an independent group of nodes with their own compute, network and storage resources. In multi-tenant configurations, each tenant is a cluster of server pools in a single namespace, fully isolated from the other tenants' server pools. Capacity can easily be added to an existing system by pointing MinIO at a new server pool and MinIO automatically prepares it for and places it in service.

Replication

The challenge with traditional replication approaches is that they do not scale effectively beyond a few hundred TB. Having said that, everyone needs a replication strategy to support disaster recovery (DR) and that strategy needs to span geographies, data centers and clouds. MinIO's continuous replication feature supports Active-Active Replication, Active-Passive Replication and Backup and DR uses. Server-side replication relies on the lambda notification API to efficiently track changes across petabytes of data. This approach pushes changes instantly to the remote sites without requiring expensive namespace scans and batched operations.

MinIO uses near-synchronous replication to update objects immediately after any mutation on the bucket. In contrast, other vendors may take up to 15 minutes to update the remote bucket. MinIO can be configured to follow strict consistency within the data center and eventual-consistency across the data centers to protect the data.

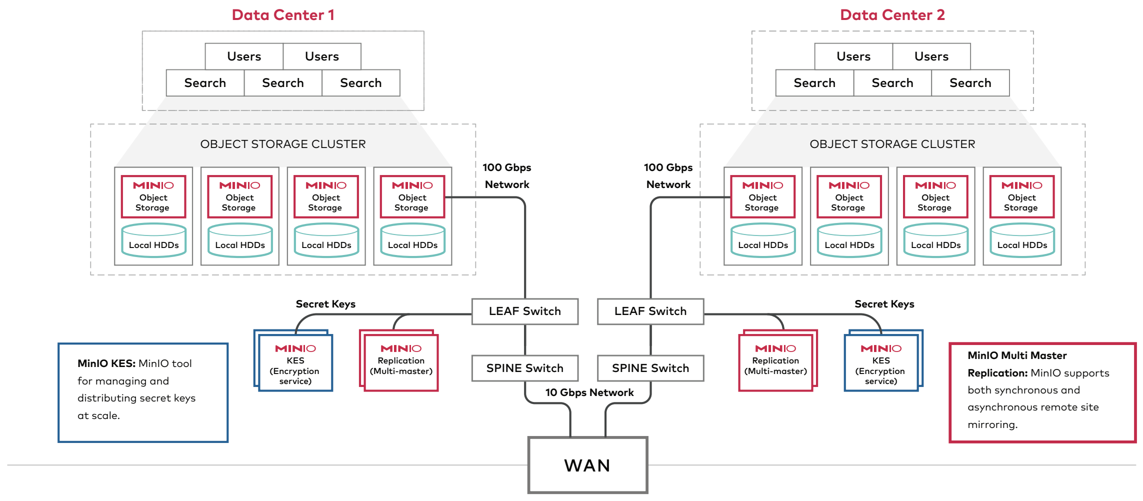

Data Center 1
Users
Users
Search
Search
Search
OBJECT STORAGE CLUSTER
MINIO Object Storage
Local HDDs
100 Gbps Network
Secret Keys
LEAF Switch
SPINE Switch
MINIO KES (Encryption service)
MINIO Replication (Multi-master)
MinIO KES: MinIO tool for managing and distributing secret keys at scale.
Data Center 2
100 Gbps Network
10 Gbps Network
WAN
MinIO Multi Master Replication: MinIO supports both synchronous and asynchronous remote site mirroring.

Figure 11: MinIO supports very large deployments in each data center.

Active-Active Cross Region/Zone Replication

MinIO's active-active replication enables organizations to use object storage across multiple data centers and clouds in a manner that is resilient and scalable that can withstand a data center failure with no down time. In this scenario, developers and applications access object storage in each MinIO cluster independently. Read-write operations can be conducted on either cluster and data is replicated in both directions between them. MinIO replicates objects and their metadata on a bucket level, using near-synchronous replication to update objects immediately after any change. Organizations can load-balance between sites to improve performance and failover between sites to maintain high availability.

Active-Passive Cross Region/Zone Replication

Active-passive replication is another strategy for leveraging the advantages of running object storage in multiple data centers and clouds. Read-write operations can be conducted on one system, while read operations are conducted on another. MinIO is commonly used in this manner to support read-only content sharing, geographic caching and disaster recovery. This one-way replication is recommended for sharing data between MinIO and third-party object storage or NAS vendors.

Replication for Backup and Disaster Recovery

MinIO's replication is frequently used for backup and disaster recovery. While developers and applications conduct read-write operations on one MinIO cluster, active-passive replication as described above can be used to back up objects for data protection. A deleted or modified object on the primary cluster can be restored from the secondary cluster. Extending this strategy, in the event that the primary cluster fails, applications and developers can be redirected to the secondary cluster temporarily where they work with the most recent backup.

Metadata Architecture

MinIO has no separate metadata store. All operations are performed atomically at object level granularity with strong-consistency. This approach isolates any failures to be contained within an object and prevents spillover to larger system failures. Each object is strongly protected with erasure code and bitrot hash. You can crash a cluster in the middle of a busy workload and still not lose any data. Another advantage of this design is strict consistency which is important for distributed machine learning and big data workloads. This architecture is particularly well suited to small objects and massive scale.

Monitoring, Logging and Alerting

MinIO provides complete visibility into clusters with per-operation logging and detailed performance and utilization metrics. MinIO exports a wide range of granular hardware and software metrics through a Prometheus-compatible endpoint. Enterprise can use MinIO's Grafana dashboard for visualizing metrics and leverage Prometheus integrations to route MinIO metrics to storage, messaging and alert services. Metrics can be tracked on a tenant level and for the entire hybrid cloud MinIO deployment. MinIO also provides a healthcheck endpoint for probing node and cluster liveness.

MinIO logs can be audited via MinIO Console and MinIO Client. MinIO supports Amazon-compatible Lambda Event Notifications to automatically send bucket and object events such as access, creation and deletion to third-party applications. The events can be delivered using industry standard messaging platforms like Apache Kafka, NATS, AMQP, MQTT, Webhooks, or a database such as Elasticsearch, Redis, Postgres and MySQL for event-driven processing in serverless or function-as-a-service computing frameworks.

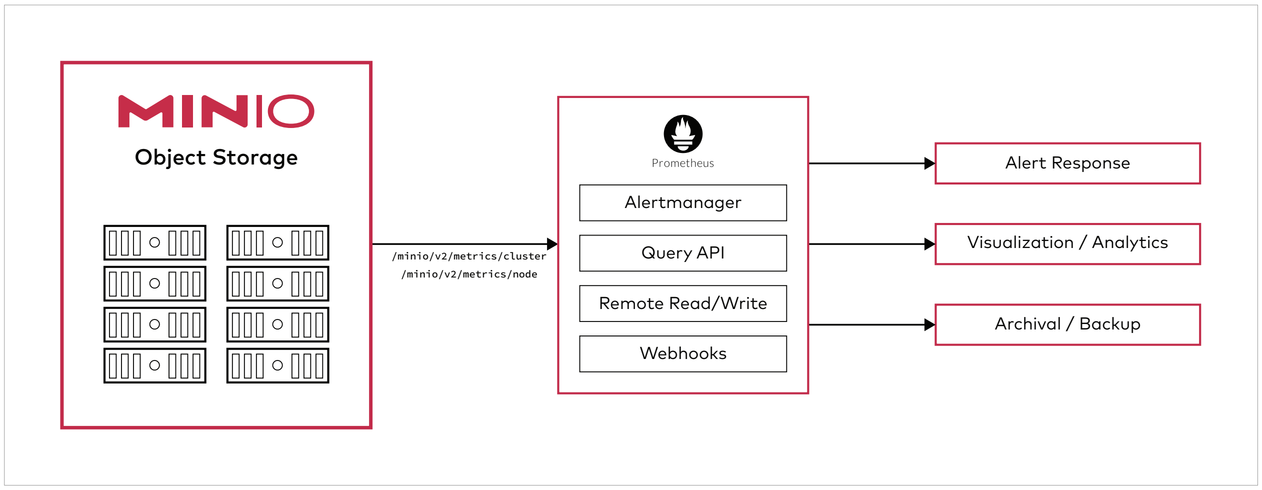


Figure 12: MinIO provides a Prometheus-compatible endpoint for customizable monitoring, logging and alerting.

Sidekick Load Balancing

Given MinIO's architecture, a standard HTTP load balancer or round-robin DNS may be employed to distribute load across MinIO nodes. For high performance applications, however, there may be a need for a more streamlined approach. Traditional load balancer appliances have limited aggregate bandwidth and introduce an extra network hop. This architectural limitation is also true for software-defined load balancers running on commodity servers.

Sidekick solves the network bottleneck challenge by taking a sidecar approach instead. It runs as a tiny sidecar process alongside each of the client applications. This way, the applications can communicate directly with the servers without an extra physical hop. Since each of the clients run their own Sidekick in a shared-nothing model, it can be scaled to any number of clients.

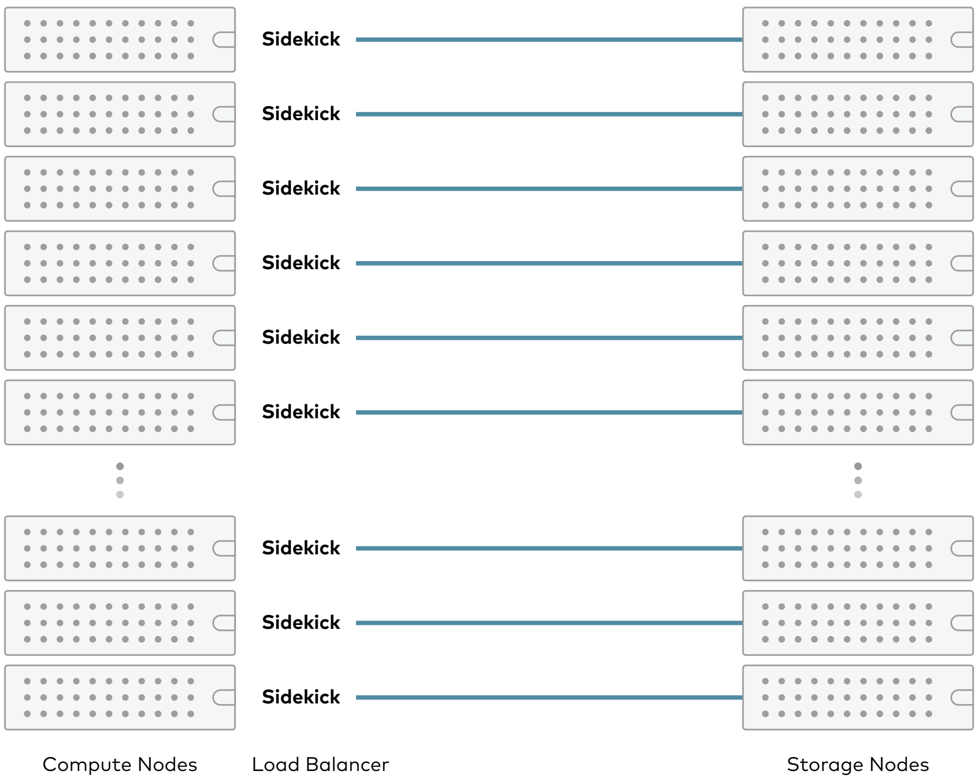


Figure 13: Sidekick provides high-performance load balancing for MinIO object storage.

In a cloud-native environment like Kubernetes, Sidekick runs as a sidecar container. Sidekick can be added to existing applications without any modification to the application binary or container image.

Multi-Tenant Object-Storage-as-a-Service with Kubernetes

The primary unit of managing MinIO on Kubernetes is the tenant. MinIO was designed and built as a multi-tenant and multi-user system that scales seamlessly from TBs to any size. The tenants are fully isolated from each other in their own namespace. Each tenant may have multiple users with varying levels of access privileges. Each tenant cluster operates independently of each other.

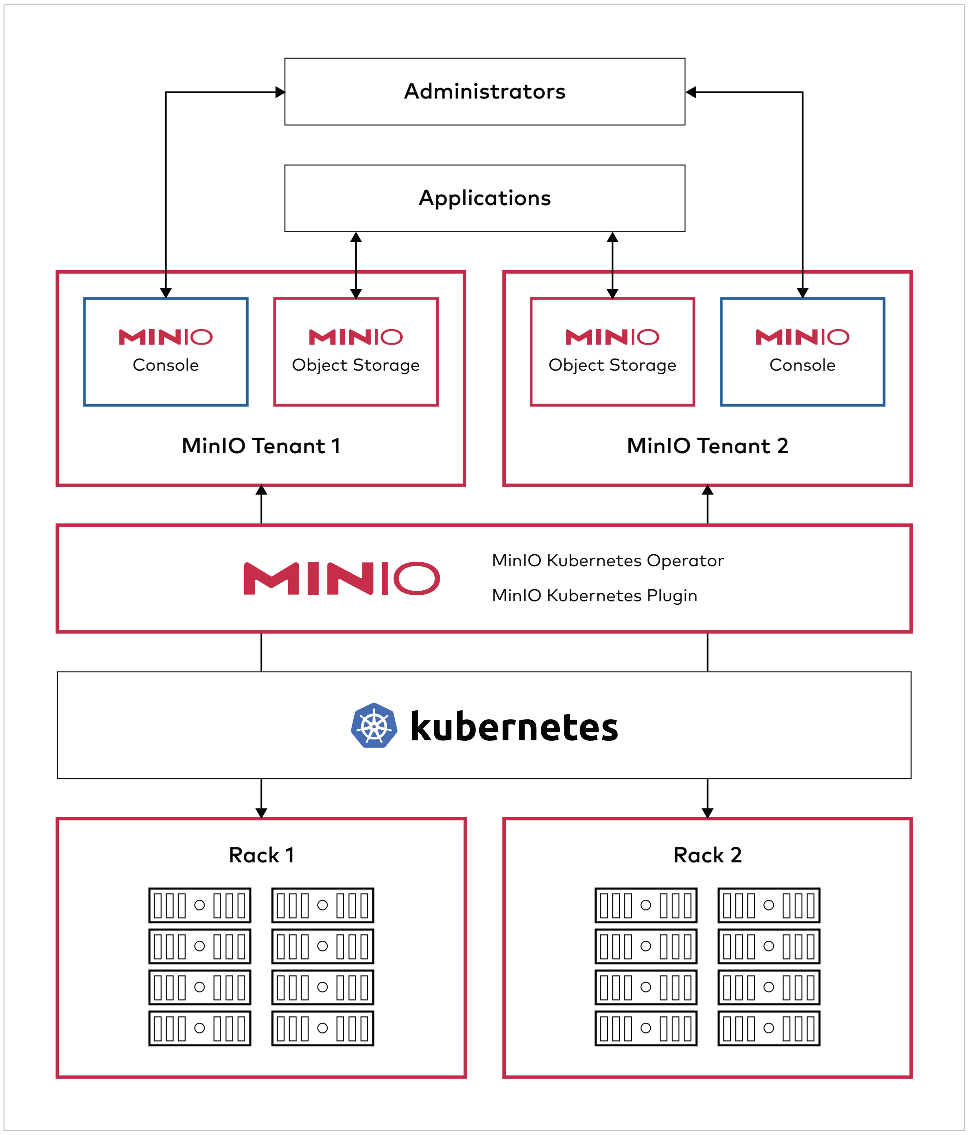


Figure 14: Architecture for a MinIO and Kubernetes multi-tenant object storage-as-a-service.

In this way, Kubernetes serves as the intersection between hardware and software. The Operator Console provides a self-service interface for tenants, and each fully isolated tenant is protected from possible disruption due to another tenant's upgrades, updates or security incidents. Each tenant scales independently across geographies and cloud infrastructures.

Hybrid Cloud

Collectively, these features enable the most powerful hybrid cloud object storage solution on the market today.

Today's enterprises require a storage strategy capable of operating in a wide range of environments, including those found in the public cloud, private cloud and at the edge.

Hybrid cloud storage follows the model established in the public cloud where the dominant storage class is object, and public cloud providers have unanimously adopted cloud-native object storage. The success of the public cloud effectively rendered file and block storage obsolete. Every new application is written for the AWS S3 API - not POSIX. In order to scale and perform like cloud-native technologies, older applications must be re-written for the S3 API and refactored into microservices to be container compatible.

Leveraging the same small binary (~45MB), MinIO enables companies to run applications on the private cloud (including Kubernetes distributions), at the edge or in the public cloud with no modification. This minimizes operational overhead, and provides flexibility to move data and applications as business requirements change, preventing lock-in to a specific cloud provider or proprietary architecture.

MinIO runs on bare metal, Kubernetes and every public cloud, including non-S3 providers like Google, Microsoft and Alibaba. More importantly, MinIO ensures that data looks exactly the same from an application and management perspective via the Amazon S3 API.

Kubernetes plays a key role in MinIO's hybrid cloud functionality. As favored by DevOps teams, Kubernetes-native design requires an operator service to provision and manage a multi-tenant object-storage-as-a-service infrastructure. Each of these tenants run in their own isolated namespace while sharing the underlying hardware resources. The operator pattern extends Kubernetes's familiar declarative API model with custom resource definitions (CRDs) to perform common operations like resource orchestration, non-disruptive upgrades, cluster expansion and to maintain high-availability.

MinIO is purpose-built to take full advantage of the Kubernetes architecture. Since the server binary is fast and lightweight, MinIO's operator is able to densely co-locate several tenants without running out of resources, enabling scalable and resilient object storage for containerized workloads.

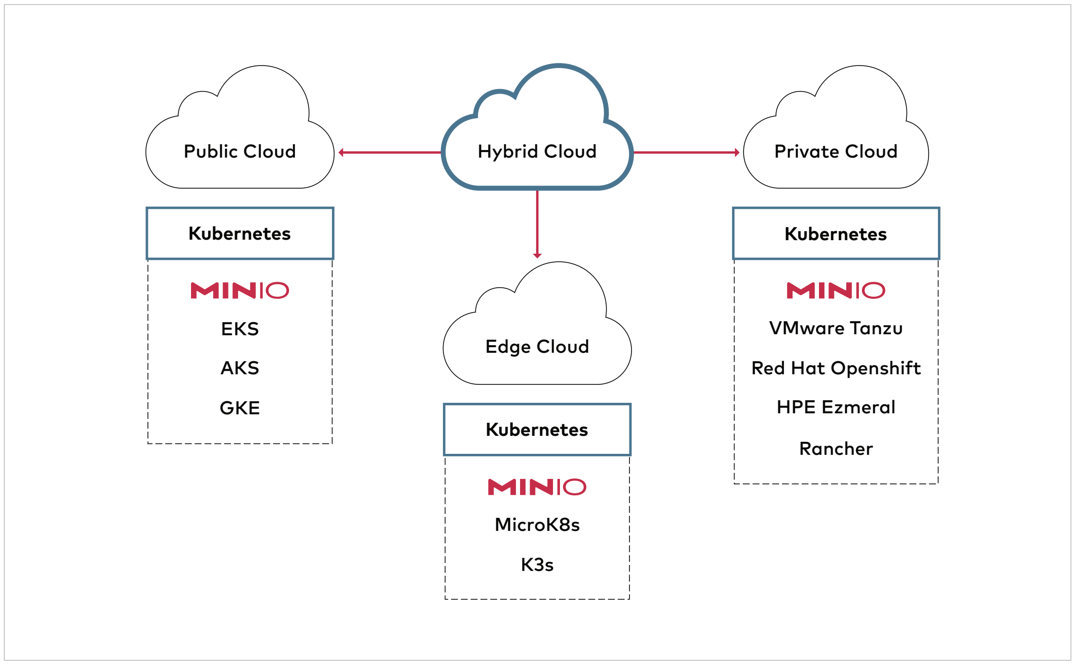


Figure 15: MinIO and Kubernetes create powerful and flexible hybrid cloud object storage.

Each tenant benefits from everything MinIO has to offer, while enjoying complete isolation from other tenants. Each tenant has its own set of protections to ensure data integrity across multiple locations.

Benchmark Performance

MinIO's performance claims are backed by well-documented benchmarks. MinIO tests against a number of different benchmarks from Warp, S3-benchmark to Hadoop DFSIO. The following represents the summary results of our S3 Bench testing on commodity and high performance hardware. Full documentation of the testing, setup and environments can be found on MinIO's website.

Our results running on an 8 node MinIO cluster can be summarized as follows:

Setup	Avg Read Throughput (GET)	Avg Write Throughput (PUT)
Distributed	46.54 GB/s	34.4 GB/s
Distributed with Encryption	46.4 GB/s	34.6 GB/s

Our results running on a 32 node MinIO cluster can be summarized as follows:

Setup	Avg Read Throughput (GET)	Avg Write Throughput (PUT)
Distributed	183.2 GB/s	171.3 GB/s
Distributed with Encryption	162 GB/s	114.7 GB/s

In addition, MinIO compares to HDFS across key tests with outstanding results, proving that enterprises do not need to trade off performance to achieve the benefits of disaggregation.

Benchmark	HDFS	MinIO	MinIO is X % faster
Distributed	1005s	820s	22.5%
Sort	1573s	793s	98.3%
Wordcount	1100s	787s	39.7%

An Enduring Commitment to Open Source

MinIO products are 100% open source under the GNU Affero General Public License, Version 3.0 (AGPLv3). It is MinIO's commitment that as long as it is independent that it will continue to be 100% open source.

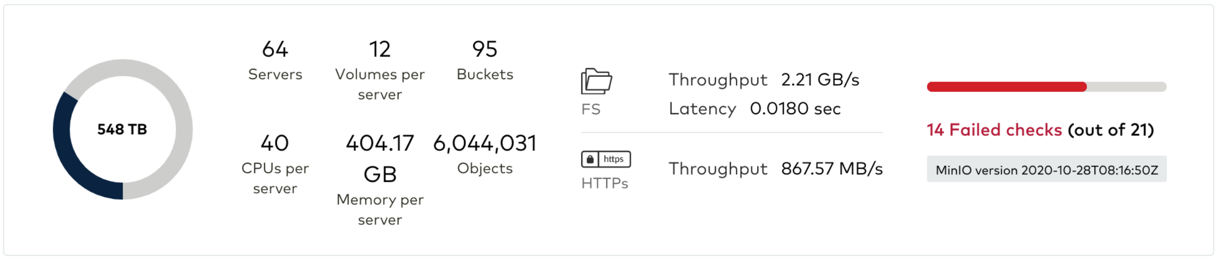
The advantages of Open Source are well understood. These include the avoidance of vendor lock-in, security, consistent innovation, transparency, and the reliability that comes with millions of community members hammering every release from every possible angle.

MinIO remains the owner of the MinIO object storage project and as such controls the quality and development through its weekly release cadence. MinIO runs a suite of acceptance tests for every pull request and every MinIO server release.

Understanding the MinIO Subscription Network

While MinIO is available under the AGPLv3, many customers choose to purchase the software on an annual subscription basis. Their reasons for doing so differ, but they are unified in the value they see in the software coupled with a desire to have a deeper relationship with the team behind MinIO.

All MinIO subscribers receive unrivaled 24/7/365 direct-to-engineer support to help build and maintain their large scale data infrastructure. In addition, MinIO subscribers can leverage SUBNET Health to run diagnostics across all distributed components, and get a comprehensive, automated and deeply descriptive report that helps diagnose and address the root cause orders of magnitude faster than otherwise possible.


548 TB

64 Servers | 12 Volumes per server | 95 Buckets
40 CPUs per server | 404.17 GB Memory per server | 6,044,031 Objects

FS: Throughput 2.21 GB/s, Latency 0.0180 sec
HTTPs: Throughput 867.57 MB/s

14 Failed checks (out of 21)
MinIO version 2020-10-28T08:16:50Z

Figure 16: SUBNET Health reports on system health and assesses component health.

The MinIO Subscription Network benefits fit into three core categories:

Commercial License - A commercial license provides exceptions to the obligations inherent in the MinIO GNU AGPL v3 license. As the copyright holder, only MinIO can provide this exception. While many enterprises require commercial licenses where AGPL v3 is present, subscribers still enjoy the benefits associated with open source - namely freedom from lock-in and freedom to inspect the source.

Data Loss Prevention - Large scale data infrastructure must consider and plan for failure as part of the design. The MinIO Subscription Network delivers access to technology and talent to manage and minimize this risk. These include direct-to-engineering support, one hour SLA, access to the Panic Button, performance and optimization audits as well as key diagnostic capabilities designed to assess the health of customer deployments.

Data Breach Prevention - Modern data infrastructure is in a state of constant change, and change brings risk. Customers of the MinIO Subscription Network have 24/7/365 access to cryptographic experts, quarterly security audits, and a suite of other technologies that will ensure MinIO's industry leading security features are properly configured while proactively identifying vulnerabilities and evolving risk.

The MinIO Subscription Network offers two tiers - Standard and Enterprise, which customers select based upon their SLA and legal requirements. The Standard Tier is priced at $.01 per GB per month. The Enterprise Tier is priced at $.02 per GB per month. The ceiling function turns off the capacity meter at 5PB or 10PB per site depending on the plan. No further charges above that threshold are applied. Pricing is month to month and subscribers can cancel at any time and with no questions asked.

Conclusion

MinIO is the fastest growing object storage system in the world for a reason. It was designed from scratch to be a key part of the cloud-native application and data stack, solving critical problems for enterprises while seamlessly integrating with existing infrastructure. MinIO delivers performance, scalability and simplicity alongside an enterprise grade feature set.

Enterprises require a fully functional, performant and resilient hybrid cloud storage infrastructure. DevOps teams have brought the concept of infrastructure-as-code to life via the Kubernetes ecosystem. Together, MinIO and Kubernetes allow enterprises to write portable microservices-based applications and run them on the optimal cloud for the workload. While DevOps is innovating software, IT storage admins can ensure performance and cost goals are being met with powerful management tools and integrations.

As importantly, MinIO is 100% open source with all of the attendant benefits. Finally, for our production customers we offer the security that comes from a direct engagement with MinIO engineering via a SUBNET subscription, including our automated troubleshooting tool, SUBNET Health.

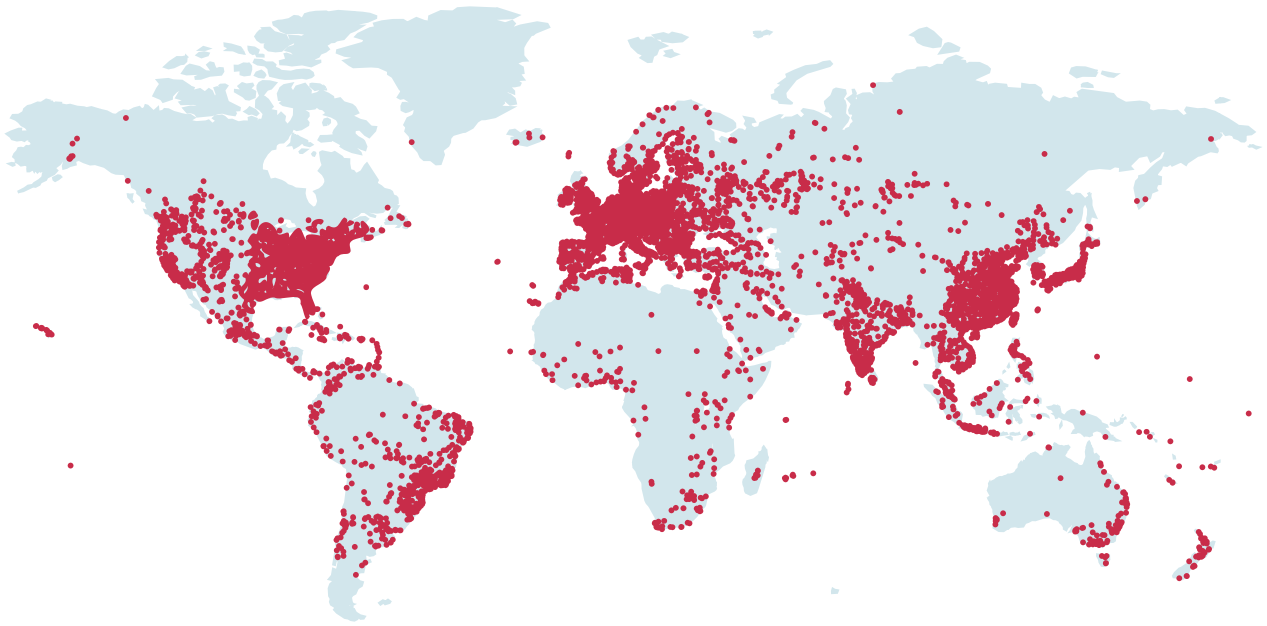
The result is the industry's most comprehensive solution for the rapidly growing world of object storage, capable of running the most demanding workloads on the most well-suited cloud anywhere across your hybrid cloud footprint.

About MinIO

Founded in 2014, MinIO is now the world's fastest growing object storage system. Backed by some of the smartest minds in storage and venture capital including Nexus, General Catalyst, Dell Technologies Capital, Intel Capital, AME Cloud Ventures and key angel investors, the company has raised $23.3M through its Series A round.

Additional Information
Email: hello@min.io
MinIO Inc.
530B University Avenue,
Palo Alto, CA 94301

Resources
https://min.io
https://docs.min.io./
https://blog.min.io./



27K+	543.3M+	12.3K+	700
GITHUB STARS	DOCKER PULLS	SLACK MEMBERS	CONTRIBUTORS

© 2021 MinIO, Inc.

*numbers in map as of April 2021