hypersphere

TECHNICAL DOCUMENTATION

# HyperSphere DNA™ for AWS

# DEPLOYMENT GUIDE

Installation, configuration, and operations reference for HyperSphere DNA™ (Data Neutralization Appliance)

DOC VERSION
1.1

AVAILABILITY
AWS Marketplace
Early Access

CONTACT
support@hyperspheretech.com
sales@hyperspheretech.com
security@hyperspheretech.com

CONTENTS

# IN THIS GUIDE

# 01 PRODUCT OVERVIEW

## 1.1 WHAT IS HYPERSPHERE DNA?

HyperSphere DNA (Data Neutralization Appliance) is a drop-in S3-compatible encryption proxy. It sits between your application and your object storage, transparently encrypting every object with AES-256-GCM before it reaches storage. Your existing S3-compatible tools, SDKs, and workflows require no code changes.

The core security property is simple: no encryption key is ever written to disk, wrapped in another key, or stored alongside your data. Keys are derived on demand in protected memory and destroyed after use. Even with full access to your storage layer, an attacker cannot decrypt your data without the key shares needed to reconstruct the root key.

## 1.2 PRIMARY USE CASES

- Regulated industries needing cryptographic separation of key material from ciphertext — financial services, healthcare, public sector.
- Development teams replacing bespoke envelope-encryption code with a drop-in S3-compatible API that handles encryption and key lifecycle automatically.
- Security teams consolidating on a single key-governance model across multiple buckets, accounts, and regions.
- Organizations seeking a no-persistent-data-keys architecture that limits blast radius in the event of a storage or credential breach.

## 1.3 HOW IT WORKS

HyperSphere DNA runs as a hardened virtual machine inside your own cloud environment. Your application sends standard S3 API calls to the HyperSphere DNA endpoint.
The appliance:

1. Authenticates the caller using your existing OIDC identity provider or SigV4 credentials.
2. Derives a unique AES-256-GCM key for each 1 MiB object frame — entirely in protected memory.
3. Encrypts the frame and writes only ciphertext to your storage target.
4. Zeroizes all key material immediately after use.
5. Writes a tamper-evident audit log entry to your storage target.

On read, the process reverses transparently. No plaintext ever leaves the appliance.

## 1.4 DEPLOYMENT OPTIONS

| OPTION | DESCRIPTION | STATUS |
|---|---|---|
| **A — Single node per region** | Each node is a stateless appliance — all encrypted data lives on your storage target, not on the node itself. Team tier and above require a minimum of 2 nodes for resilience; 3 are recommended for production. Recovery from node loss requires only relaunching a replacement and unsealing it. | Available |
| **B — Warm standby** | Active/passive configuration where a standby node tracks the primary. Failover is operator-initiated — the operator promotes the standby and restarts it as primary. Opt in via environment variables at launch. | Available (operator-initiated) |
| **C — Independent regional** | Separate single-node deployments per region with storage-level replication for encrypted frames. Each region is cryptographically independent. | Available |

> **NOTE** Because the node holds no durable state, it is disposable — data survives node loss with zero data loss (RPO = 0) since all ciphertext lives on your storage target. Team tier and above run a minimum of 2 nodes for resilience. Additional nodes provide availability, not throughput — scale up instance size and worker processes for performance. Warm standby (Option B) requires operator-initiated promotion; failover is not automatic.

# 02 BEFORE YOU BEGIN

## 2.1 WHAT THE APPLIANCE INCLUDES

HyperSphere DNA ships as a pre-built machine image. You do not supply an operating system, database, or web server.

| COMPONENT | WHAT IT DOES |
|---|---|
| **Hardened Ubuntu 24.04 LTS** | Minimal, CIS-aligned operating system baseline with systemd service sandboxing. |
| **Encryption engine (hsdna-worker)** | The core S3 proxy and encryption pipeline. Runs as an unprivileged system user. |
| **Admin CLI (hsdnactl)** | Command-line tool for unsealing, key operations, vault management, audit verification, and diagnostics. Installed on the node. |
| **Data-plane CLI (hsdna-cli)** | Command-line tool for S3 operations and access credential management. Separate client tool not installed on the node. |
| **Nginx (TLS 1.3)** | Reverse proxy handling TLS termination and HSTS enforcement. |
| **Prometheus + Grafana** | Bundled observability stack with pre-built dashboards and alert rules. |
| **React web console** | Browser-based management interface for vaults, credentials, and monitoring. |

> **NOTE** The appliance does not include a bundled object store or identity provider. Your own S3-compatible storage target and OIDC provider are configured at first boot.

## 2.2 WHAT YOU NEED BEFORE DEPLOYING

- A supported cloud account with sufficient compute quota for your chosen tier.
- A private subnet with outbound connectivity to your storage target, your OIDC identity provider, and the HyperSphere license endpoint.
- An existing object storage bucket (or S3-compatible equivalent) to use as the storage target.

- An OIDC identity provider — Okta, Azure AD, Google Workspace, or a self-hosted Keycloak instance. You will need the issuer URL, JWKS URL, and audience.
- A DNS name for the appliance (for TLS certificate provisioning via Let's Encrypt or your certificate authority).
- A secure location for each key share produced during the first-boot setup process.

### 2.3 SKILLS REQUIRED

- Basic cloud infrastructure administration (networking, compute instances, IAM roles, storage buckets).
- Familiarity with your OIDC identity provider's configuration.
- Basic Terraform knowledge for the recommended deployment path.
- Basic Linux command-line administration.

No cryptographic expertise is required. The appliance handles all key derivation and encryption operations automatically.

## 2.4 RESPONSIBILITIES

| RESPONSIBILITY | HYPERSPHERE TECHNOLOGIES | YOU (THE OPERATOR) |
|---|---|---|
| **Appliance software and security** | Builds, hardens, signs, and publishes the machine image. Publishes patched images monthly; emergency patches within 24 hours of a critical fix. | Selects and deploys the published image. Rolls to new images during your change windows. |
| **Infrastructure** | n/a | Owns and operates the compute, networking, storage, and identity infrastructure. |
| **Key share custody** | Provides the key share setup tools. | Stores key shares securely and presents them at unseal. |
| **Encryption key material** | Keys never leave the appliance or touch disk. | Holds the key shares that reconstruct the root key at boot. |
| **Encrypted data** | Provides the encryption engine. | Owns all data in your storage target. Responsible for backup, retention, and residency. |
| **Monitoring** | Ships bundled Prometheus + Grafana dashboards and alert rules. | Reviews dashboards and alerts. Optionally forwards metrics and logs to your monitoring stack. |
| **Support** | Provides tiered support (see §9). | Reports issues and operates the customer-side infrastructure. |

# 03 ARCHITECTURE

## 3.1 SINGLE-NODE TOPOLOGY

Figure 1 shows a complete single-node deployment. The appliance runs in a private subnet. Your application connects to the HyperSphere DNA endpoint using standard S3 API calls over TLS. Encrypted frames are written to your storage target. No plaintext ever leaves the appliance.

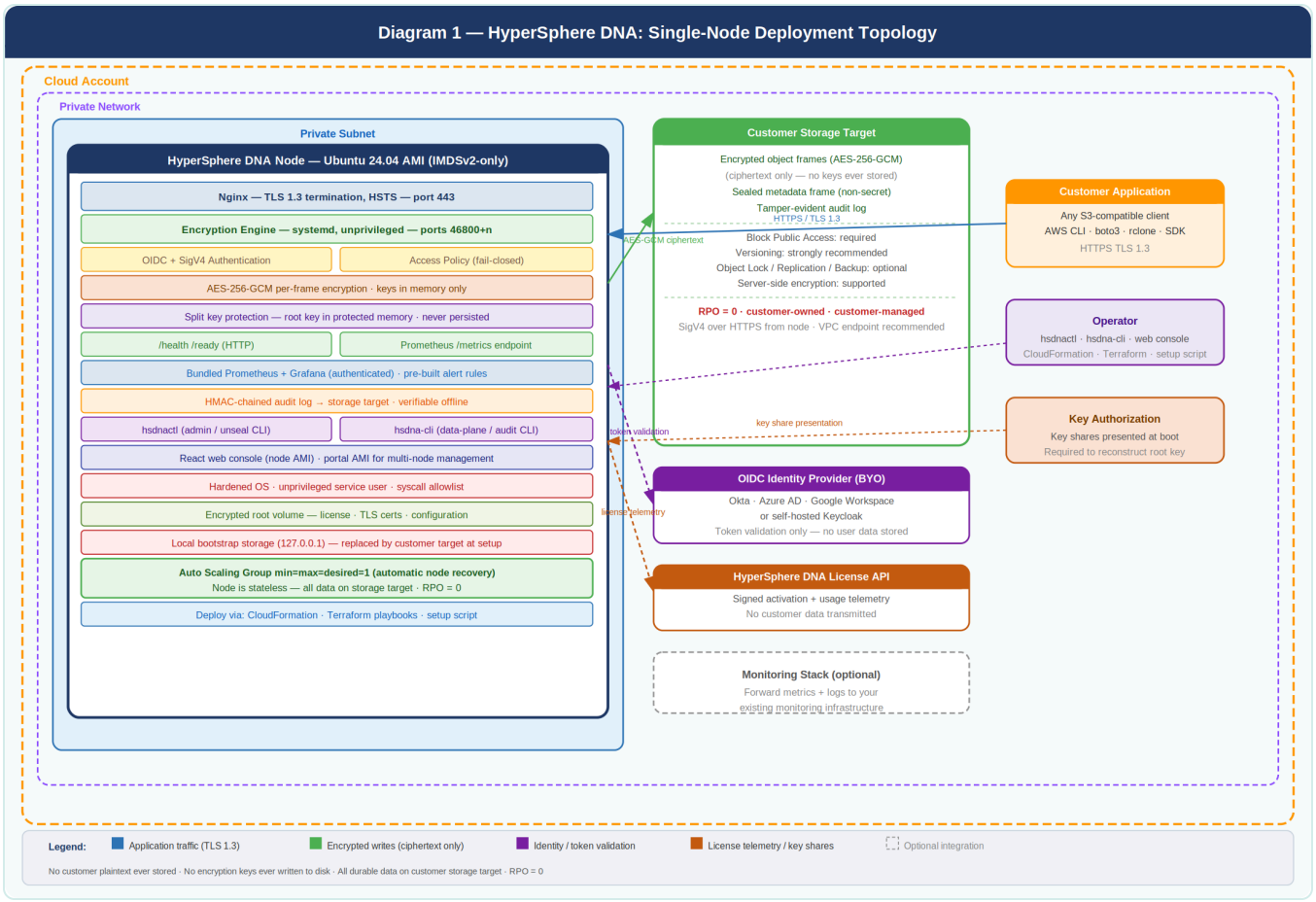


FIGURE 1 · SINGLE-NODE DEPLOYMENT TOPOLOGY

## 3.2 KEY PROTECTION MODEL

HyperSphere DNA's security foundation is that no encryption key is ever written to disk, stored in a database, or sent off the appliance. Here is how that works in practice:

- At boot, the required key shares are presented to reconstruct the root key in protected memory. The root key exists only in RAM and is never persisted anywhere.

- Every encryption key used to protect your data is derived from the root key on demand — computed in memory at the moment it is needed, used immediately, then destroyed. Keys are never cached, never wrapped in another key, and never stored.
- The only values that live on your storage target are non-secret identifiers used as inputs to the derivation process. These are meaningless without the root key — they cannot be used to decrypt anything on their own.
- There is no key database, no key escrow, and no back door. If too many key shares are lost, the encrypted data cannot be recovered by anyone — including HyperSphere Technologies.

This design means that a breach of your storage layer alone yields only ciphertext. An attacker who gains full access to your storage bucket has no path to plaintext without the key shares needed to reconstruct the root key.

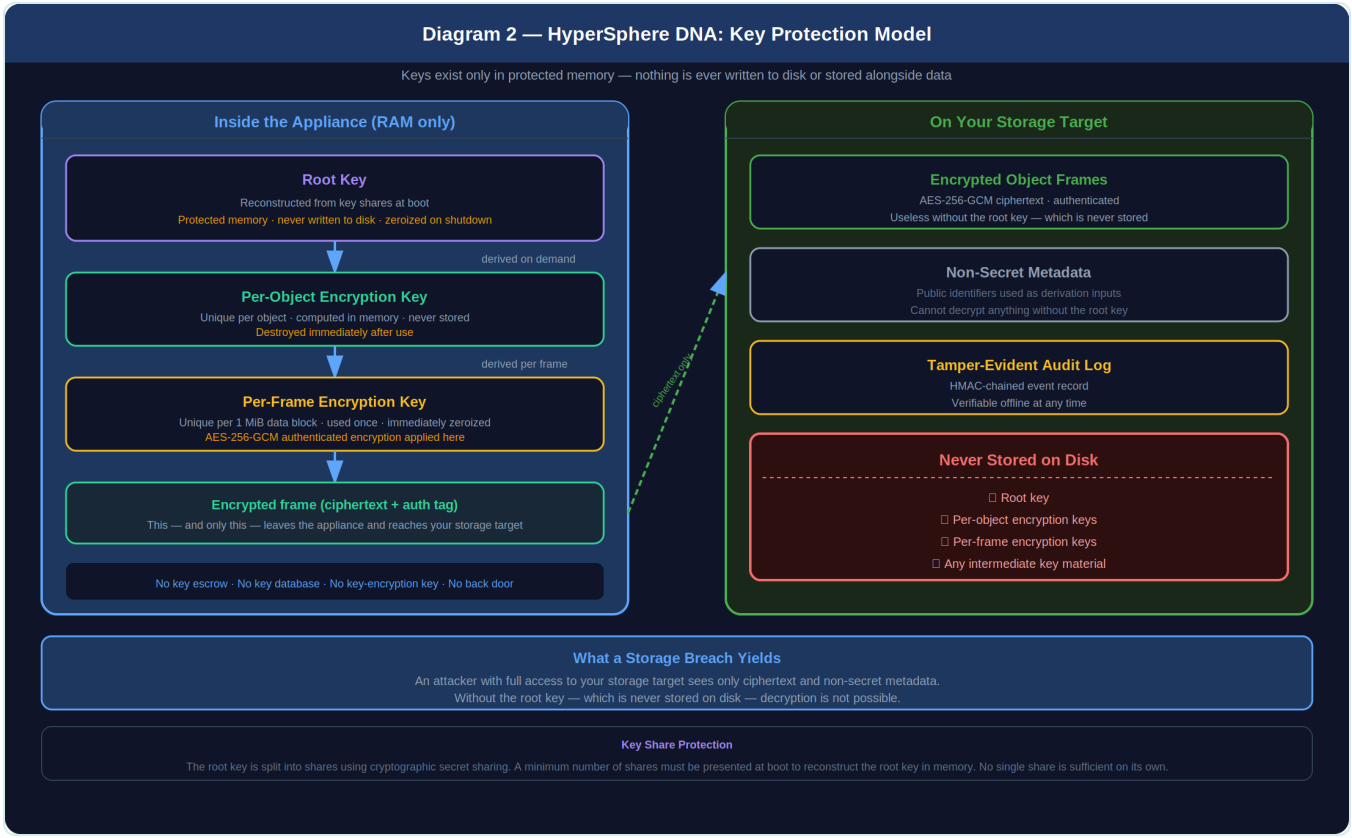


FIGURE 2 · KEY PROTECTION MODEL — KEYS EXIST ONLY IN MEMORY, NEVER AT REST

## 3.3 DATA FLOW

Figure 3 shows the complete PUT encryption pipeline and GET decryption path. Every step — authentication, authorization, key derivation, encryption, and audit logging — happens inside the appliance before ciphertext reaches your storage target.

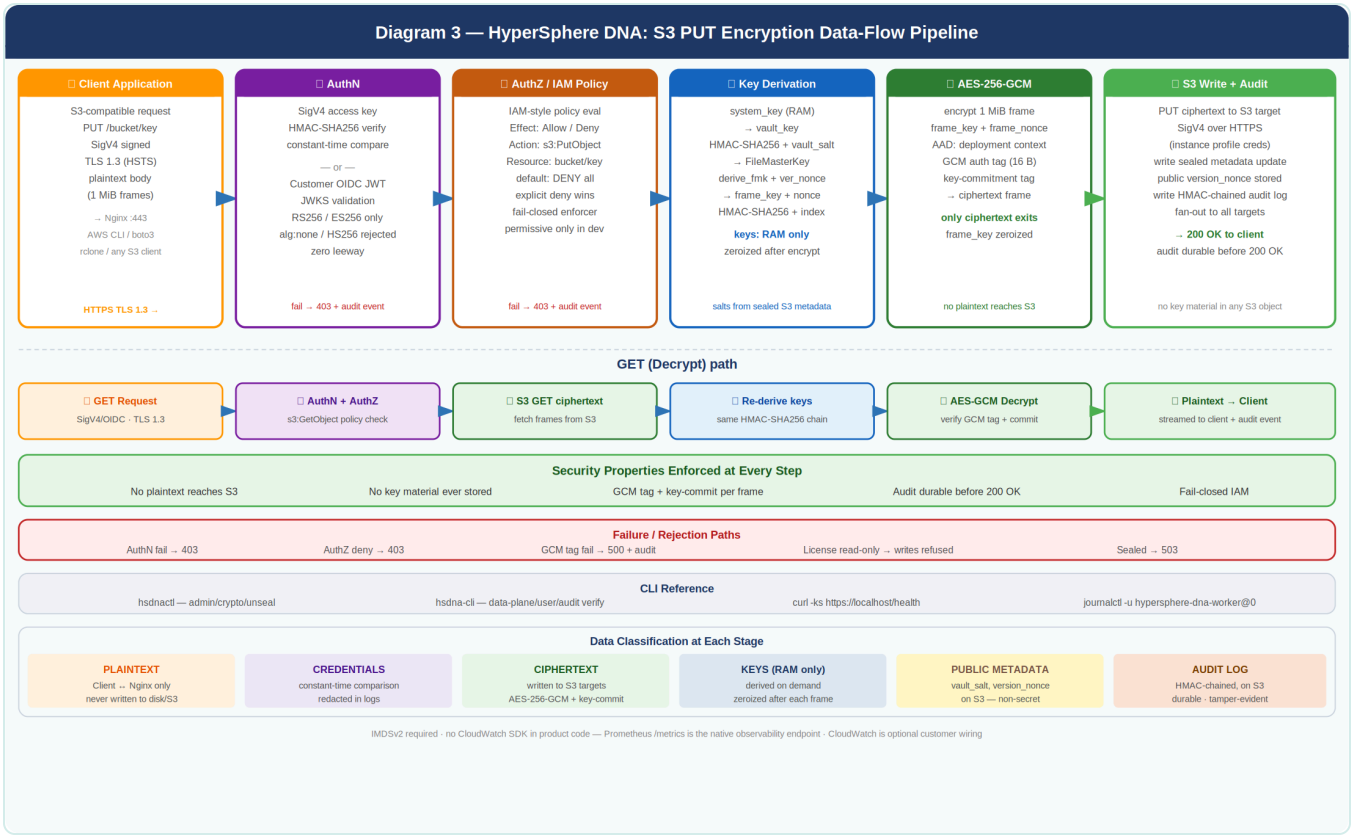


FIGURE 3 · S3 PUT ENCRYPTION DATA-FLOW PIPELINE

## 3.4 AUTHENTICATION AND ACCESS

HyperSphere DNA supports two authentication paths for callers. Both paths use the same fail-closed IAM-style authorization engine on the appliance.

| PATH | HOW IT WORKS | BEST FOR |
|---|---|---|
| **OIDC (recommended for humans)** | Callers present a JWT from your OIDC identity provider. The appliance validates the token against your JWKS endpoint. RS256 and ES256 only — alg:none and HS256 are rejected. Zero clock leeway. | Administrators, console users, human operators. |
| **SigV4 credentials** | Callers use HyperSphere API credentials (not cloud IAM keys) minted by the appliance. Verified using HMAC-SHA256 with constant-time comparison. | Application workloads, automated processes, CLI access. |

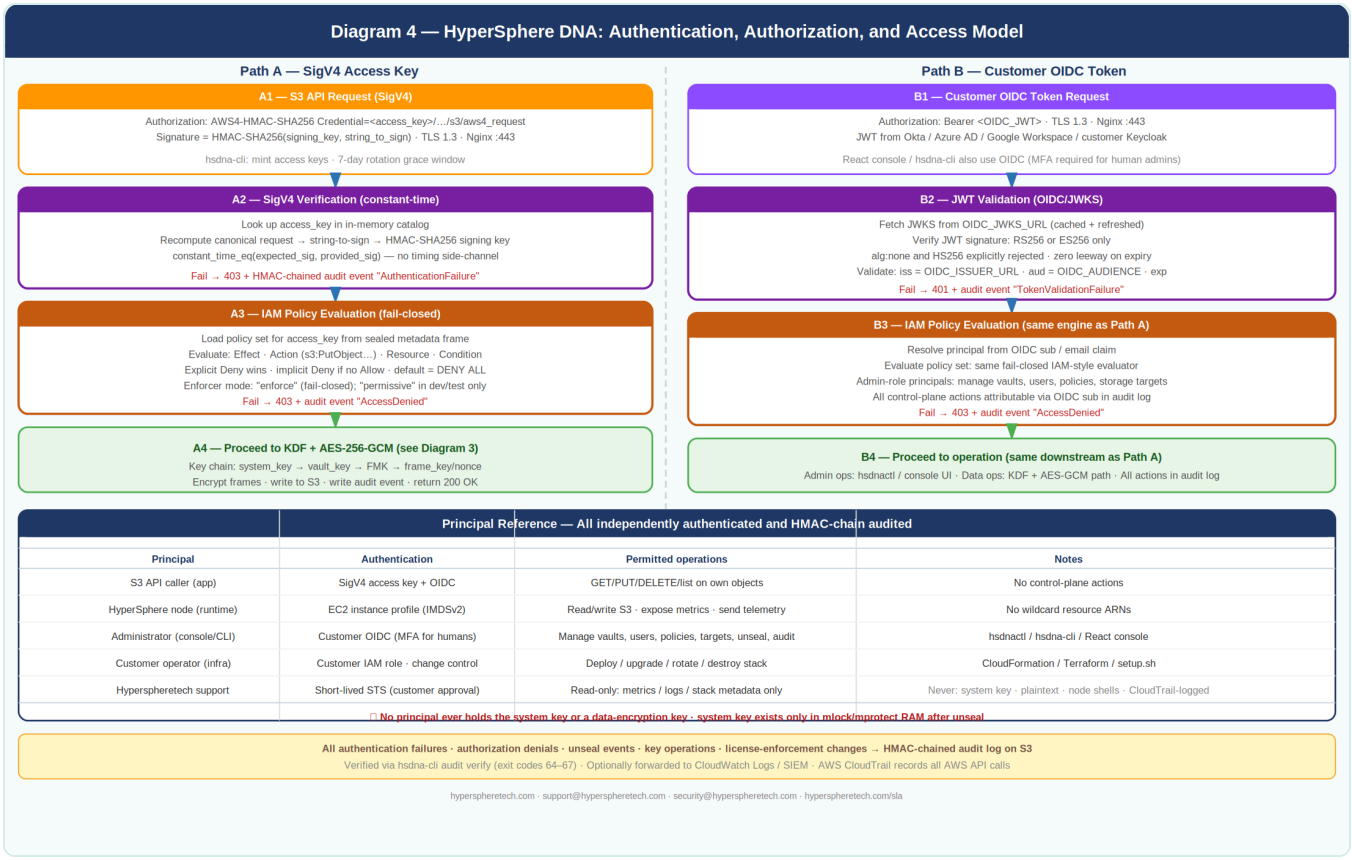


FIGURE 4 · AUTHENTICATION, AUTHORIZATION, AND ACCESS MODEL

## 3.5 INTEGRATION POINTS

| INTEGRATION | DIRECTION | NOTES |
|---|---|---|
| **S3-compatible API (your application)** | Your app → HyperSphere DNA | HTTPS / TLS 1.3. No application code changes required. |
| **Storage target (your bucket)** | HyperSphere DNA → your bucket | SigV4 over HTTPS. Uses the appliance's instance role — no user credentials. |
| **OIDC identity provider** | HyperSphere DNA → your IdP | HTTPS. JWKS endpoint for token validation. BYO — Okta, Azure AD, Google Workspace, or Keycloak. |
| **License / telemetry** | HyperSphere DNA → HyperSphere API | HTTPS. Signed activation and usage telemetry. No customer data transmitted. |
| **Observability** | Prometheus scrapes /metrics | Bundled Prometheus + Grafana. Optionally forward to your monitoring stack. |

# 04 SECURITY

## 4.1 OPERATING SYSTEM HARDENING

The machine image is built on Ubuntu 24.04 LTS with a CIS-aligned baseline. Key hardening measures:

- Minimal package set — only software required to run HyperSphere DNA is installed.
- The encryption engine runs as an unprivileged system user with no login shell.
- Systemd unit hardening: `NoNewPrivileges`, `ProtectSystem=strict`, `PrivateTmp`, `PrivateDevices`, capability and syscall allowlist.
- IMDSv2 enforced at image build time. Set `HttpTokens=required`, `HttpPutResponseHopLimit=1` on instance launch.
- No default credentials of any kind are present in the image.

## 4.2 KEY MANAGEMENT

- No encryption key is ever written to disk, stored in a database, or wrapped in another key.
- The system key exists only in mlock/mprotect-protected memory after the unseal ceremony.
- All lower keys (vault key, FileMasterKey, frame key/nonce) are derived on demand and zeroized after each use.
- There is no key-encryption key (KEK) and no key escrow.
- The only values stored on your storage target are public salts and nonces — these are not keys and cannot decrypt anything.

## 4.3 NETWORK SECURITY

- The appliance listens on port 443 (TLS 1.3, HSTS) for S3 API and console traffic only.
- The self-signed TLS certificate present at first boot is for initial connectivity only. Replace it with a CA-issued or Let's Encrypt certificate before handling production traffic.
- SSH access should be restricted to your administrator CIDR and disabled once the appliance is configured.
- No public IP address is required when clients connect from within the same private network.

- A VPC gateway endpoint for S3 is recommended to keep storage traffic on the cloud provider's internal network.

**TLS** Let's Encrypt is the default TLS configuration. The appliance only falls back to a self-signed certificate when the endpoint is a bare IP address (no DNS name). For production deployments, ensure the node has a DNS name configured so Let's Encrypt can issue a valid certificate, or terminate TLS at a load balancer with a certificate from your CA.

## 4.4 AUDIT LOGGING

Every security-sensitive event is written to a tamper-evident HMAC-chained audit log on your storage target before the operation returns. The chain can be verified offline at any time.

| WHAT IS LOGGED | WHERE IT GOES |
|---|---|
| Authentication successes and failures | HMAC-chained audit log on your storage target |
| Authorization decisions (allow and deny) | HMAC-chained audit log on your storage target |
| Unseal events and key-share presentations | HMAC-chained audit log on your storage target |
| License enforcement state changes | HMAC-chained audit log on your storage target |
| S3 object operations (caller, object hash, outcome) | Structured application logs (system journal) |
| All cloud API calls made by the appliance | Your cloud provider's API audit trail (automatic) |

Verify audit chain integrity at any time using the `hsdnactl` admin CLI on the node:

```
hsdnactl audit verify
```

# 05 DEPLOYMENT

## 5.1 CHOOSE YOUR TIER

HyperSphere DNA is available in five tiers. Choose the tier that matches your throughput and data volume requirements. For current pricing, visit the AWS Marketplace listing or contact sales@hyperspheretech.com.

| TIER | INSTANCE | DATA BAND | MIN / REC NODES | USE CASE |
|---|---|---|---|---|
| **Developer** | `t3.medium` | 10 GB | 1 / 1 | Development, testing, and evaluation only — not for production. |
| **Team** | `m7i.large` | 100 GB | 2 / 2 | Small production workloads. |
| **Business** | `m7i.xlarge` | 1 TB | 2 / 3 | Standard production. |
| **Capacity** | `m7i.2xlarge` | 10 TB | 2 / 3 | High-throughput production. |
| **Enterprise / Gov** | Custom | Unlimited / custom | Custom | Large-scale, regulated, or custom SLA requirements. Contact sales@hyperspheretech.com. |

**SCALING PRINCIPLE**

Additional nodes provide resilience, not performance. Scale up (larger instance size + more worker processes per node) for throughput. Scale out (additional nodes) for availability. The minimum of 2 nodes for Team and above is a resilience recommendation, not a throughput requirement.

**FREE TRIAL**

30 days, single node, any instance size, with a 10 GB data band cap. Auto-converts to a paid subscription at the end of the trial period.

**UPGRADE PATH**

To move between tiers, subscribe to the new tier on the Marketplace listing, relaunch on the tier-appropriate instance size, re-provision against the same storage target, and unseal. All data is preserved — RPO = 0.

## 5.2 SIZING

Throughput scales by running additional encryption engine processes on the same node (ports 46800+n) or by moving to a larger instance size within the same tier. If your workload grows beyond a tier's data band, upgrade to the next tier — the instance size and data band are linked to your subscription level.

## 5.3 STEP-BY-STEP DEPLOYMENT

### STEP 1 LAUNCH THE APPLIANCE

Deploy using one of the following paths:

**Path A — CloudFormation template (recommended):**

```
aws cloudformation deploy \
  --stack-name hypersphere-dna-prod \
  --template-file deploy/cloudformation/hypersphere-node.yaml \
  --capabilities CAPABILITY_IAM \
  --parameter-overrides AmiId=<ami-id> VpcId=<vpc-id> \
      PrivateSubnetId=<subnet-id> StorageBucketName=<bucket> \
      AppClientCidr=<cidr> AdminCidr=<cidr>
```

**Path C — Manual:** launch the image, attach a security group (port 443 from your application CIDR, SSH from your admin CIDR), attach the instance role, and attach a 50 GB encrypted volume.

### STEP 2 RUN FIRST-BOOT CONFIGURATION

When deploying via Terraform, first-boot runs automatically via cloud-init. When deploying manually (Path C), SSH in and run:

```
sudo /opt/hypersphere-dna/setup.sh
```

The setup script will:

1. Activate your license against the HyperSphere license API.
2. Configure your storage target and OIDC identity provider.
3. Generate key shares and store them securely in separate locations.
4. Configure Nginx and TLS (Let's Encrypt by default; self-signed fallback for bare IP endpoints).

5. Start the encryption engine service.

> **CRITICAL** Key shares are displayed once during the first-boot setup process. Record and store each share securely in a separate location immediately. Losing too many shares makes your encrypted data permanently unrecoverable — there is no back door.

## STEP 3 UNSEAL THE APPLIANCE

HyperSphere DNA boots in a sealed state and will not serve traffic until unsealed. After every reboot, present the required key shares to reconstruct the root key in memory:

```
hsdnactl unseal --share-file <key-share.toml>
```

Repeat for each share until the minimum threshold is met. The appliance will then transition to ready state and begin accepting traffic.

## STEP 4 CREATE A VAULT AND CREDENTIALS

Use the web console (https://<node-ip>/) or the `hsdnactl` CLI to create a vault, attach your storage target, and create API credentials for your application.

## STEP 5 CONNECT YOUR APPLICATION

Point your application's S3 endpoint at the HyperSphere DNA node. Use any S3-compatible client — AWS CLI, boto3, rclone, or any SDK. No code changes required.

```
curl -ksSf https://<node-ip>/ready   # 200 = ready; 503 = sealed
aws s3 --endpoint-url https://<node-ip> ls s3://your-bucket
```

# 06 HEALTH CHECKS AND MONITORING

## 6.1 HEALTH ENDPOINTS

The appliance exposes two health endpoints with distinct purposes:

| ENDPOINT | HTTP STATUS | RESPONSE | MEANING |
|---|---|---|---|
| `GET /health` | 200 | `{"status":"alive","service":"hsdna_worker","version":"...",...}` | Liveness check only. Confirms the process is running. Does not indicate seal state. |
| `GET /ready` | 200 | (empty body) | Readiness check. The appliance is unsealed and accepting S3 traffic. |
| `GET /ready` | 503 | (empty body) | Not ready. The appliance is sealed or the storage target is unreachable. Check seal state. |

**SEAL STATE** GET /health does not report seal state. Use GET /ready to determine whether the appliance is unsealed and ready to serve traffic. A 503 from /ready means the appliance needs to be unsealed.

## 6.2 BUNDLED OBSERVABILITY

The appliance ships with Prometheus and Grafana pre-configured. Access Grafana at https://<node-ip>/grafana/ using the credentials set during first-boot configuration.

Pre-built dashboards cover:

- Request rate and error rate
- Storage target round-trip latency
- Seal state and unseal events
- License and data band usage
- Encryption engine throughput per worker process

Alert rules ship in deploy/prometheus/alerts.yaml. Review and adjust thresholds to match your SLOs before going to production.

## 6.3 FORWARDING TO YOUR MONITORING STACK

The appliance exposes a Prometheus /metrics endpoint. Forward metrics and logs to your existing monitoring infrastructure by running the CloudWatch agent or a compatible log shipper on the instance. No monitoring SDK is embedded in the appliance — all native observability is Prometheus-based.

## 6.4 KEY DIAGNOSTIC COMMANDS

| COMMAND | WHAT IT DOES |
|---|---|
| `curl -ks https://localhost/health` | Liveness check — confirms the process is running. |
| `curl -ks https://localhost/ready` | Readiness check — 200 means unsealed and serving; 503 means sealed or not ready. |
| `hsdnactl unseal --share-file <share>` | Present a key share to unseal the appliance. |
| `hsdnactl audit verify` | Verify the integrity of the HMAC-chained audit log (run on the node). |
| `journalctl -u hypersphere-dna-worker@0` | View live encryption engine logs. |
| `hsdnactl crypto credential-rewrap` | Re-seal stored storage-target credentials. |

# 07 BACKUP AND RECOVERY

## 7.1 WHAT NEEDS TO BE PROTECTED

| DATA | WHERE IT LIVES | HOW TO PROTECT IT | HOW TO RECOVER |
|---|---|---|---|
| **Encrypted object frames** | Your storage target | Enable versioning, cross-region replication, object lock, and automated backup on your bucket. | Restore from S3 version or replica. |
| **Sealed metadata frame** | Your storage target | Same protections as object frames — it lives in the same bucket. | Restored automatically when the appliance connects to the bucket. |
| **Key shares** | Secure offline storage | Store each share in a separate, secure location away from the appliance and the storage target. | Present shares using hsdnactl unseal to reconstruct the root key at boot. |
| **Appliance configuration** | Instance volume | Snapshot the root volume periodically, or re-provision using the same parameters. | Restore from snapshot, or rerun setup.sh with the same configuration. |
| **Audit logs** | Your storage target | Same protections as object frames. | Restored with the storage target. Verify chain with hsdnactl audit verify. |

**CRITICAL** Losing too many key shares makes your encrypted data permanently unrecoverable. There is no back door and no key escrow. Store shares in separate, secure locations and maintain at least one more share than the minimum threshold.

## 7.2 RECOVERY SCENARIOS

| SCENARIO | DATA LOSS (RPO) | RECOVERY TIME (RTO) | STEPS |
|---|---|---|---|
| **Engine process crash** | Zero | Under 30 seconds | The service restarts automatically. |
| **Node loss** | Zero (data on storage target) | 10-20 minutes | Launch a new node from the image, run setup.sh pointing at the same storage target, unseal. |
| **Storage target loss** | Last version or replica | Depends on storage restore | Restore from versioning or cross-region replica. |
| **Insufficient key shares** | N/A | Unrecoverable | Store shares separately. Maintain more shares than the minimum threshold required. |

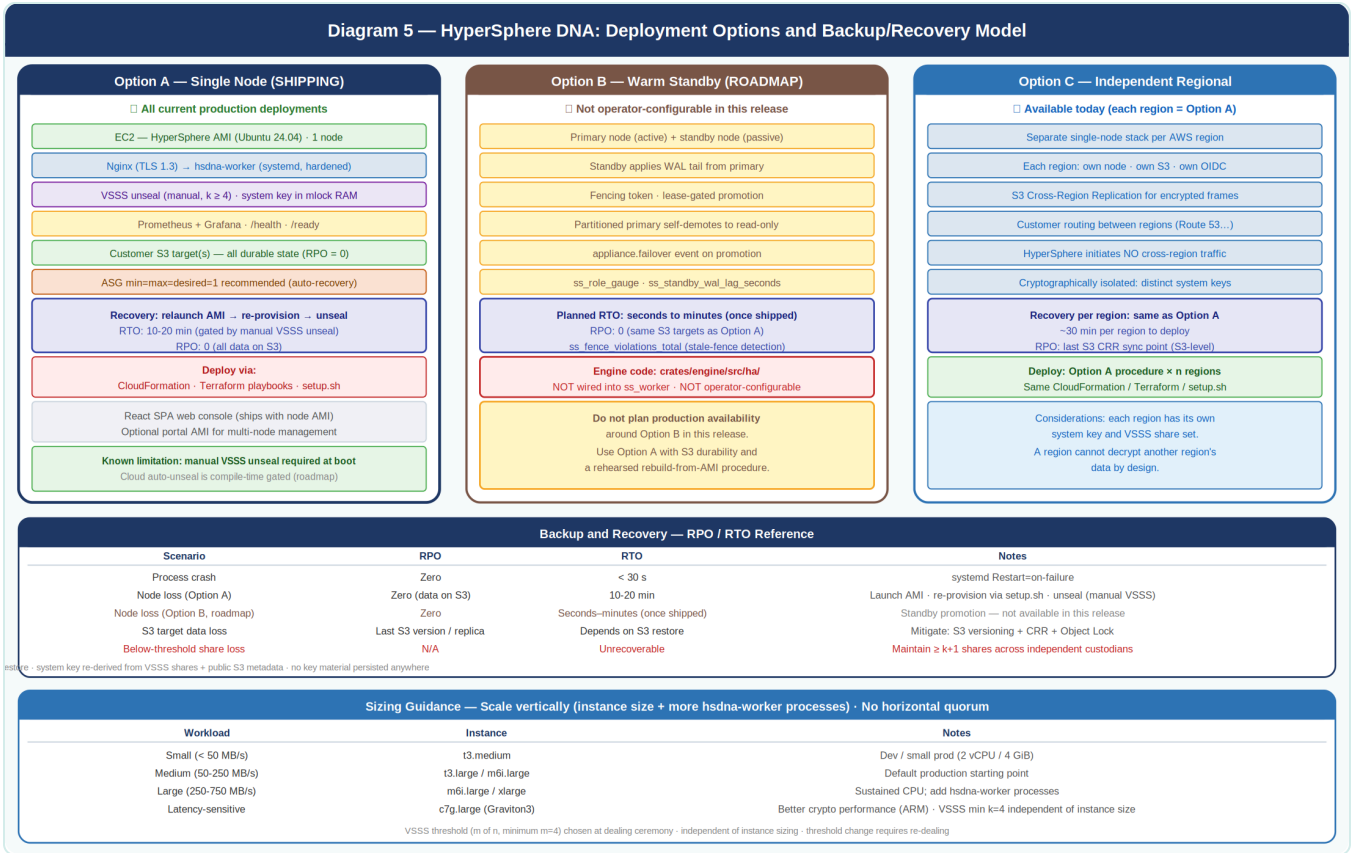


FIGURE 5 · DEPLOYMENT OPTIONS, SIZING, AND RECOVERY REFERENCE

# 08 MAINTENANCE

## 8.1 ROUTINE OPERATIONS

| TASK | HOW OFTEN | HOW TO DO IT |
|---|---|---|
| **Review Grafana dashboards and alert rules** | Weekly | Open https://<node-ip>/grafana/ and check for any firing alerts. |
| **Verify audit chain integrity** | Monthly | Run hsdnactl audit verify and confirm exit code 0. |
| **Rotate API credentials** | Per your security policy | Use hsdna-cli or the web console. A 7-day grace window allows credential overlap during rotation. |
| **Rotate key shares** | Per security policy or as needed | Run the hsdnactl key ceremony to produce new shares and store them in secure locations. |
| **Monitor data band usage** | Monthly | Check the license and data band dashboard in Grafana. Upgrade tier before the limit is reached. |
| **Renew TLS certificate** | Annually or as required | ACM auto-renews if using a load balancer. Let's Encrypt auto-renews if configured during setup. |

## 8.2 UPGRADING THE APPLIANCE

HyperSphere Technologies publishes updated machine images monthly. Emergency images are published within 24 hours of a critical security fix. To upgrade:

1. Subscribe to release notifications at support.hyperspheretech.com.
2. Review the release notes for any breaking changes.
3. Launch a new node from the updated image and provision it against the same storage target and OIDC configuration.
4. Unseal the new node and verify it is healthy.
5. Cut traffic over from the old node to the new one.
6. Terminate the old node.

If an issue is found after cutover, roll back by relaunching the previous image version. Your data is unaffected — it remains on your storage target.

## 8.3 LICENSE AND TIER MANAGEMENT

- Keep your subscription active at the correct tier.
- Monitor data band usage in Grafana. Exceeding the tier limit triggers read-only enforcement — writes are refused while reads continue.
- To upgrade tier: subscribe to the new tier, relaunch on the appropriate instance size, reprovision, and unseal. Data is preserved with RPO = 0.
- Ensure the appliance has outbound HTTPS access to the license endpoint at all times. Sustained connectivity loss triggers read-only enforcement after the grace period.

## 8.4 FAULT REFERENCE

| SYMPTOM | LIKELY CAUSE | RESOLUTION |
|---|---|---|
| `GET /ready` returns `503` | Appliance is sealed — rebooted and needs unsealing, or storage target unreachable. | Present the required key shares using hsdnactl unseal until the threshold is met. |
| **S3 client 403** | Invalid API credentials. | Verify credentials using hsdna-cli. Check that the vault and storage target are configured correctly. |
| `SignatureDoesNotMatch` | Clock skew between client and appliance exceeds 15 minutes. | Synchronise NTP on both the client and the appliance. |
| **Connection refused on port 443** | Nginx is not running, or TLS is misconfigured. | Check the Nginx service status and error log. |
| **Writes refused, reads work** | License grace period expired, or data band exceeded. | Renew or upgrade your subscription and confirm the appliance has license endpoint connectivity. |
| **Unseal threshold not met** | Insufficient key shares presented. | Gather the remaining required key shares and present them using hsdnactl unseal. |
| **High latency on large objects** | Instance is undersized for the workload. | Upgrade to the next tier, or add additional worker processes on the current node. |

# 09 SUPPORT

## 9.1 GETTING HELP

Open a support case via support.hyperspheretech.com, email support@hyperspheretech.com, or post on AWS re:Post (tag hyperspheretech). For security vulnerability reports: security@hyperspheretech.com.

When opening a case, include your deployment identifier and a diagnostics bundle — the output of journalctl -u hypersphere-dna-worker@0 and hsdnactl audit verify.

## 9.2 SUPPORT TIERS

| TIER | INCLUDED WITH | CHANNELS | COVERAGE |
|---|---|---|---|
| **Standard** | All subscriptions | Support portal, email, community forum, documentation | Business hours, Monday–Friday |
| **Premium** | Production deployments | Portal, email, chat, phone | 24×7 for Sev-1 and Sev-2; business hours for Sev-3 and Sev-4 |
| **Enterprise** | Enterprise / Gov tier | Portal, email, chat, phone, named Technical Account Manager | 24×7 for Sev-1 through Sev-3 |

## 9.3 RESPONSE TIMES

| TIER | SEV-1 (CRITICAL OUTAGE) | SEV-2 (IMPAIRED) | SEV-3 (NON-CRITICAL) | SEV-4 (QUESTION) |
|---|---|---|---|---|
| **Standard** | 4 business hours | 8 business hours | Next business day | 3 business days |
| **Premium** | 2 hours | 4 hours | 8 business hours | 2 business days |
| **Enterprise** | 1 hour | 2 hours | 4 business hours | 1 business day |

Full SLA terms: hyperspheretech.com/sla

END OF DEPLOYMENT GUIDE