Technical Assessment

AWS Lakehouse to Sovereign OpenStack Migration

Assessment for lift-and-shift migration services targeting EU sovereign cloud infrastructure

1. Current State Architecture (AWS)

Layer	AWS Service	Function
Storage	S3	Object storage with strong consistency
Table Format	Iceberg	Open table format (portable)
Catalog	Glue Data Catalog	Hive Metastore-compatible, schema registry
SQL Engine	Athena	Serverless Trino-based query engine
Processing	Spark (EMR/Glue)	Heavy transformations, compaction

2. Target State Architecture

The target architecture replaces AWS-managed services with open-source equivalents running on Kubernetes, deployed on sovereign OpenStack infrastructure:

Query Layer: Trino on Kubernetes (Interactive SQL) + Spark on Kubernetes (Batch ETL) **Catalog Layer:** Apache Polaris or Nessie (REST Catalog API) **Table Format:** Apache Iceberg (unchanged - this is the portability win) **Storage Layer:** Ceph RadosGW or MinIO (S3-compatible object storage)

3. Component-by-Component Analysis

3.1 Storage Layer

Solution	Pros	Cons	Recommendation
Ceph RadosGW	Battle-tested at scale, native OpenStack integration, erasure coding	Complex operations, needs dedicated expertise	Large deployments (>500TB)
MinIO	Simple, fast, excellent S3 compatibility, easy K8s deployment	Less mature multi-site replication	Small-medium (<500TB)

Critical S3 compatibility requirements for Iceberg: ListObjectsV2 with consistent listing, atomic PutObject/DeleteObject, multipart upload support, S3A Hadoop connector compatibility.

Migration approach: Iceberg data files are immutable Parquet + metadata JSON. Bulk copy with rclone or s5cmd—no transformation needed.

3.2 Catalog Layer

Glue Catalog functions to migrate:

Glue Function	Required?	Replacement
Table metadata store	Yes	Polaris / Nessie
Schema registry	Yes	Built into Iceberg
Partition management	Yes	Built into Iceberg
Access control	Yes	Polaris RBAC / External (OPA)
Crawlers	No	Not migrating
ETL job definitions	No	Not migrating

Recommendation: Apache Polaris

Native Iceberg REST Catalog implementation, fine-grained RBAC built-in, multi-engine support (Spark, Trino, Flink). Young project (incubating) but strong backing from Snowflake donation.

3.3 SQL Query Engine

Athena → **Trino** is the natural path. Athena is Trino under the hood.

Aspect	Compatibility	Notes
SQL syntax	~98%	Minor function name differences
Iceberg operations	100%	Same connector lineage
MERGE INTO	Supported	Full Iceberg connector support
Time travel	Supported	FOR VERSION AS OF / FOR TIMESTAMP AS OF
Athena-specific functions	Partial	approx_percentile params differ slightly

Key operational difference: Athena is serverless (pay per query). Trino requires cluster sizing, autoscaling configuration, query queue management, and memory tuning.

3.4 Spark Processing

Spark on Kubernetes is production-ready (native since 3.1, mature in 3.4+). Deployment via Spark Operator with dynamic allocation.

Glue job migration: Most Glue ETL jobs are PySpark. Migration involves extracting scripts, removing Glue-specific context (GlueContext, DynamicFrame), replacing with standard SparkSession + Iceberg catalog config, packaging and submitting via Spark Operator.

3.5 Scaling Architecture

Kubernetes is the orchestration layer. On OpenStack:

Approach	When to use
Magnum (OpenStack-native K8s)	Deep OpenStack integration, if available on target cloud
Rancher / RKE2	Multi-cluster management, hybrid scenarios
Vanilla K8s + Cluster API	Maximum control, cloud-agnostic

Node pool design:

• **trino-workers:** 8 vCPU/32GB, autoscale 2-20 nodes, trigger on pending pods + CPU

• **spark-executors:** 8 vCPU/32GB, autoscale 0-50 nodes (scale to zero), trigger on Spark Operator requests

• **system:** Fixed 3 nodes (HA) for Polaris, monitoring, ingress

4. Migration Execution Phases

Phase	Timeline	Activities
1. Infrastructure	Week 1-2	Provision K8s cluster, deploy Ceph/MinIO storage, deploy Polaris catalog, network/DNS/TLS
2. Parallel Operation	Week 3-4	Copy data files S3→sovereign, register tables in Polaris, deploy Trino, validate query parity
3. Processing Migration	Week 5-6	Convert Glue jobs to Spark Operator, test in parallel, validate Iceberg maintenance
4. Cutover	Week 7-8	Switch applications to new endpoints, monitoring, performance tuning, decommission AWS

5. Risk Matrix

Risk	Likelihood	Impact	Mitigation
S3 API edge case incompatibility	Low	Medium	Thorough testing with actual workload patterns
Trino performance tuning gap	Medium	Medium	Engage Trino expertise, use Starburst resources
Catalog migration data loss	Low	High	Dual-write period, rollback procedure
Operational burden underestimated	High	Medium	Build runbooks, consider managed Trino
Query performance regression	Medium	Medium	Benchmark critical queries before/after

6. Service Deliverables

A migration engagement would deliver:

1. **Assessment tooling** — Automated inventory of Glue catalog, job dependencies, query patterns
2. **Migration automation** — Scripts/tooling for data copy, catalog registration, job conversion
3. **Reference architecture** — Terraform/Helm for target stack deployment
4. **Validation framework** — Query result comparison, performance benchmarking
5. **Runbooks** — Day-2 operations for the self-managed stack

Document generated for internal discussion purposes. Technical specifications subject to validation against specific customer environments.