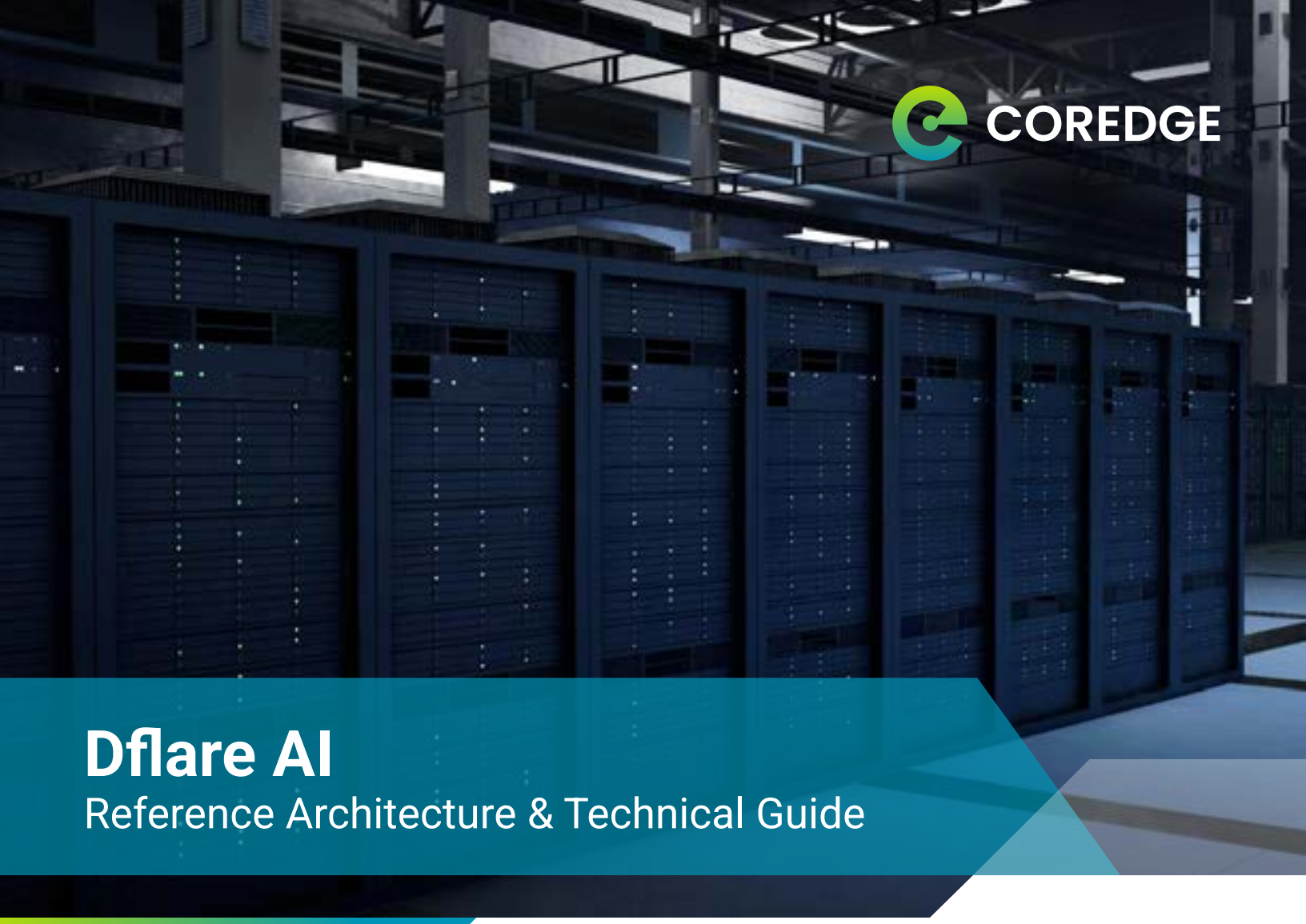


COREDGE

Dflare AI

Reference Architecture & Technical Guide

Technical Guide

A comprehensive technical reference for enterprise GPU infrastructure

Covering system design, architecture decisions, operational behavior, and deployment patterns.

Version 2.0

Contents

Introduction

Dflare AI is an enterprise GPU infrastructure platform designed to deliver bare metal performance with cloud-like usability. This document presents a reference architecture similar in structure to leading cloud providers, focusing on system design, architectural decisions, and operational behavior.

KEY INSIGHT

This guide serves as the definitive technical reference for architects, engineers, and operators deploying or evaluating Dflare AI for production GPU workloads.

Problem Statement

Modern AI workloads require:

- Massive GPU Scale
- Strict multi-tenant isolation
- High-throughput data pipelines
- Support for both cloud-native and HPC workloads

Traditional cloud and on-prem systems fail to deliver all four simultaneously.

KEY INSIGHT

Dflare AI was purpose-built to address this gap — providing unified GPU infrastructure that combines bare metal performance, hardware-enforced isolation, and full lifecycle automation.

The AI Infrastructure Challenge

Bare Metal Performance

Eliminate virtualization overhead to maximize GPU efficiency. Standard GPU slicing via NVIDIA MIG (Multi-Instance GPU) profiles enables partitioning supported GPUs into isolated instances.

Dual Fabric Separation

- Ethernet — control plane + services
- InfiniBand — high-performance data plane

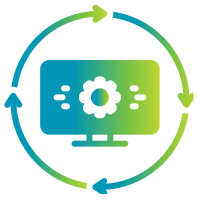
Multi-Tenant Isolation by Design

Isolation enforced across:

- Identity
- Network
- Storage
- Compute

Unified Orchestration Layer

Single control plane manages Kubernetes and Slurm.



Full Lifecycle Automation

Provision > operate > monitor > bill

Reference Architecture

The platform is composed of three primary layers:

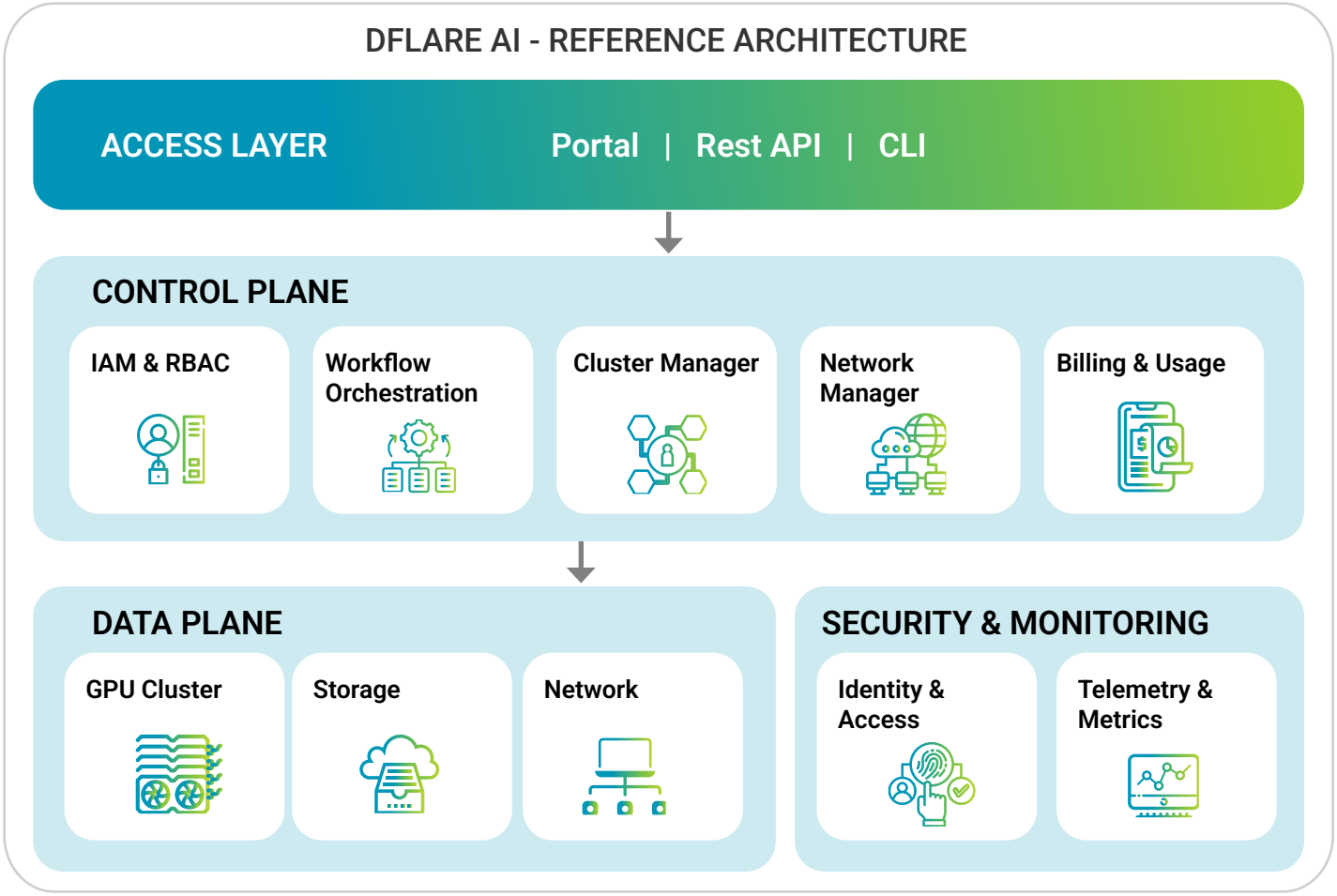


Exhibit 1: Dflare AI reference architecture - from users to infrastructure.

Access Layer

- Portal UI
- REST APIs
- CLI / automation

Control Plane

- Workflow Orchestrator
- Cluster Manager
- Network Manager
- Identity & Access
- Monitoring & Metering

Data Plane

- GPU nodes (bare metal)
- Kubernetes / Slurm workloads
- InfiniBand fabric
- Parallel filesystem

Control Plane Architecture

The control plane is responsible for orchestration, policy enforcement, and maintaining the desired state of the system. It operates as a set of loosely coupled microservices communicating over authenticated internal APIs (gRPC/REST), designed for idempotency and eventual consistency.

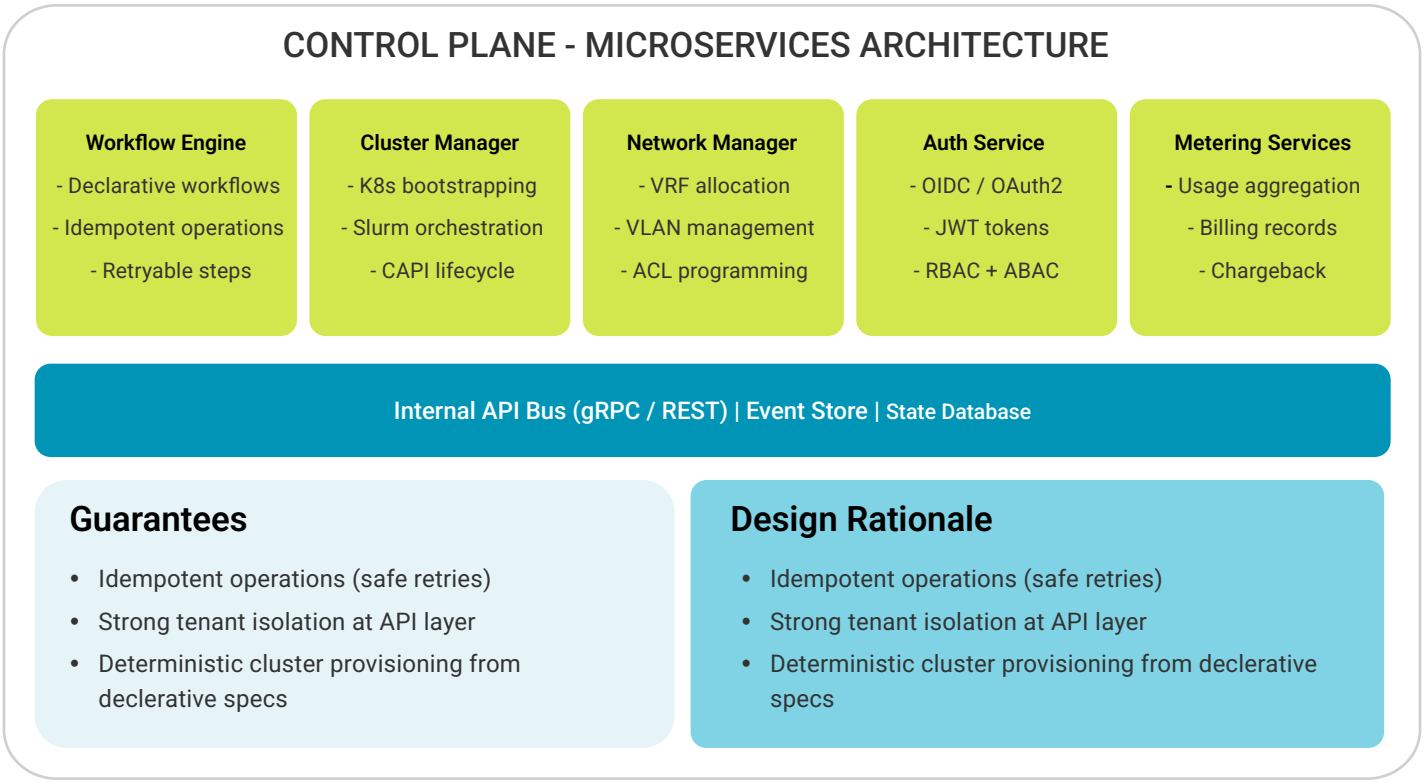


Exhibit 2: Control plane microservices architecture with guarantees.

CORE COMPONENTS

Workflow Engine

Executes declarative workflows for provisioning and lifecycle operations. It consumes high-level intents (e.g., “create cluster”) and decomposes them into ordered, retryable steps. Each step is idempotent and emits state transitions to the orchestration store.

Cluster Manager

Translates cluster specifications into node-level actions. For Kubernetes, powered by CKP (Coredge Kubernetes Platform), it bootstraps control plane components (API server, etcd, scheduler) as static pods and ensures quorum formation. For Slurm, it deploys controller, compute daemons, and accounting services via operator-based orchestration on Kubernetes.

Network Manager

Interfaces with the fabric controller to allocate tenant-scoped VRFs, assign VLANs from managed pools, and bind subnets via SVI interfaces. It enforces L3 isolation at the switch control plane and programs ACLs for east-west and north-south traffic.

Auth Service (IAM)

Integrates with enterprise identity providers (OIDC/OAuth2). Issues short-lived JWTs, enforces RBAC/ABAC at every API boundary, and isolates tenants via dedicated realms and scoped tokens.

Metering Service

Aggregates usage signals from telemetry pipelines and lifecycle events. Produces auditable, tenant-scoped billing records with strong consistency guarantees for chargeback.

GUARANTEES

- Idempotent operations (safe retries)
- Strong tenant isolation at API layer
- Deterministic cluster provisioning from declarative specs

DESIGN RATIONALE

Microservices enable independent scaling and failure isolation — control plane disruptions don't impact workloads.

Data Plane Architecture

The data plane executes workloads and handles all compute and data movement. It is optimized for throughput, latency, and hardware-level efficiency.

Components

Bare metal GPU nodes with direct device access

Container runtime (OCI-compliant)

Kubernetes worker nodes and Slurm compute daemons

GPU runtime operators (drivers, device plugins, telemetry)

Execution Model

Workloads are scheduled onto GPU nodes via Kubernetes or Slurm. GPU allocation is enforced via device plugins (K8s) or GRES (Slurm), ensuring exclusive or partitioned access depending on configuration.

KEY INSIGHT

Separating execution from orchestration ensures that control plane failures do not impact running workloads — a key requirement for long-running AI training jobs.

Guarantee	Mechanism	Benefit
Near-zero overhead	No hypervisor - bare metal	Maximum GPU efficiency
Deterministic allocation	Device plugins / GRES	Predictable performance
Guarantee	cgroups + namespace boundaries	Secure multi-tenancy

Network Architecture

The data plane executes workloads and handles all compute and data movement. It is optimized for throughput, latency, and hardware-level efficiency.

Dual Fabric Network-Architecture

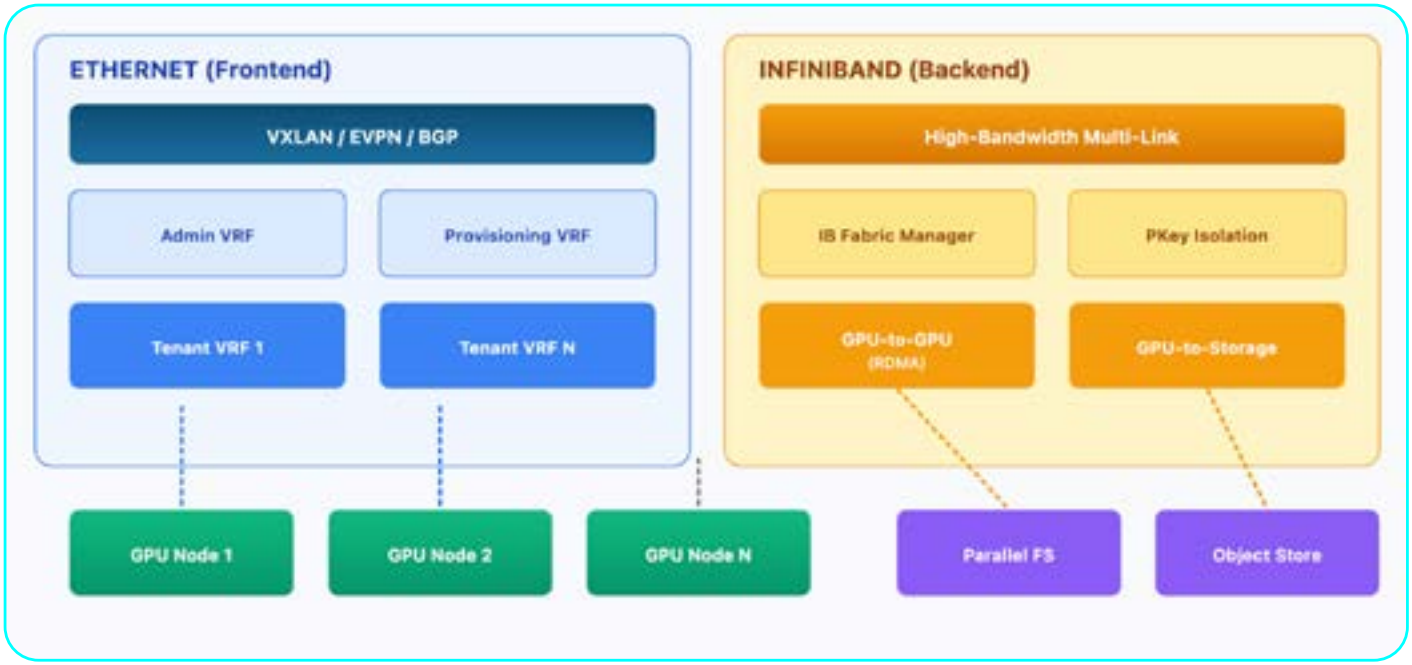


Exhibit 3: Dual-fabric network architecture with hardware-enforced tenant isolation.

Ethernet Fabric (Control Plane & Tenant VPCs)

- VXLAN/EVPN overlay with BGP underlay
- VRF per tenant for L3 isolation
- VLAN per subnet for L2 segmentation

InfiniBand Fabric (High-Performance Data Plane)VPCs)

RDMA-based communication

Partition Keys for tenant isolation

Optimized for east-west GPU traffic

Component Interaction

The Network Manager programs VRFs/VLANs via the fabric controller API. Switches enforce isolation at hardware level. GPU nodes connect to both fabrics: Ethernet for control and InfiniBand for data.

Guarantee	Description
Strict tenant isolation	No cross-VRF routing permitted
Low-latency GPU communication	RDMA over InfiniBand
Deterministic segmentation	Hardware-programmed at switch level

Trade-offs

Dual fabric increases operational complexity

Requires specialized hardware (IB switches)

KEY INSIGHT

Ethernet provides flexibility and ecosystem compatibility, while InfiniBand provides the performance required for distributed AI training.

Storage Architecture

The data plane executes workloads and handles all compute and data movement. It is optimized for throughput, latency, and hardware-level efficiency.

Storage Architecture Dual-Tier

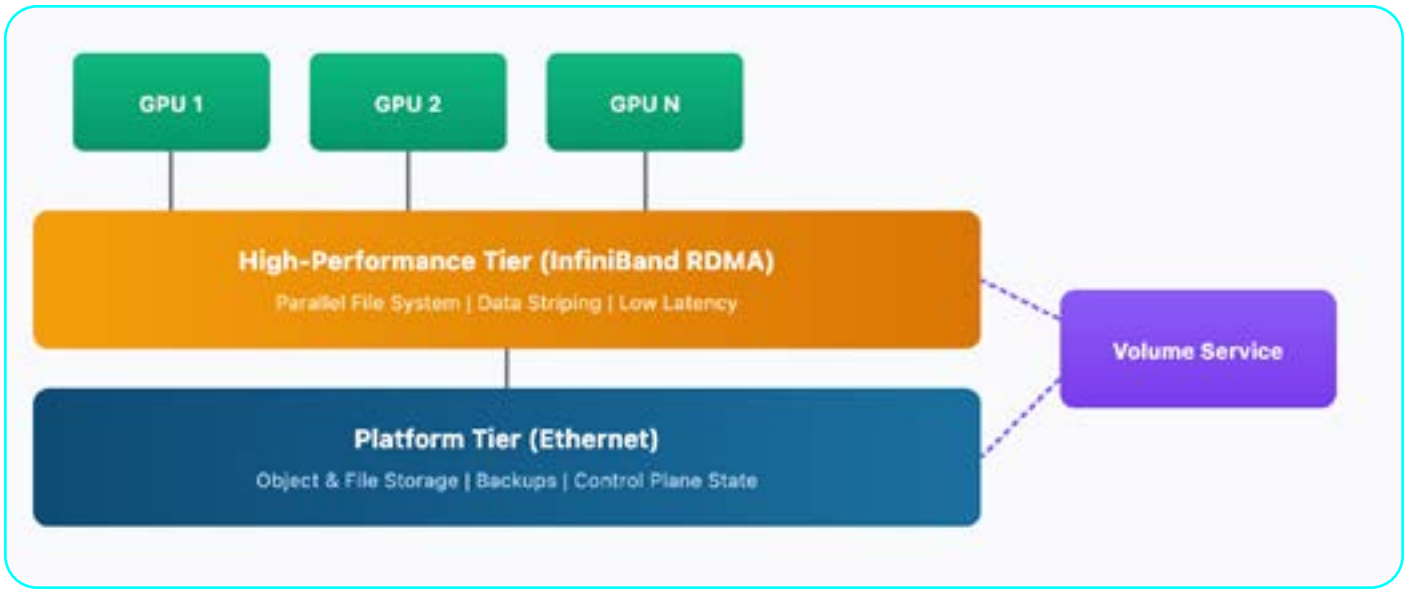


Exhibit 3a: Storage architecture - dual-tier design with RDMA-enabled high-performance path

High-Performance Tier

- Parallel filesystem accessed over InfiniBand
- Data striped across multiple storage targets
- Optimized for large sequential I/O

Platform Tier

- Object and file storage over Ethernet
- Used for backups, logs, and control plane state

Component Interaction

The Volume Service provisions tenant directories, assigns quotas, and configures access control maps. GPU nodes mount storage via RDMA-enabled clients.

Guarantee	Description
High throughput	Aligned with GPU demand via parallel I/O
Dual-layer isolation	IB partition + filesystem ACL
Consistent latency	Maintained under concurrent load

KEY INSIGHT

AI training workloads are throughput-bound; traditional storage systems cannot sustain required bandwidth. Dflare AI's storage tier is purpose-built for GPU-scale I/O.

Compute Orchestration

The data plane executes workloads and handles all compute and data movement. It is optimized for throughput, latency, and hardware-level efficiency.

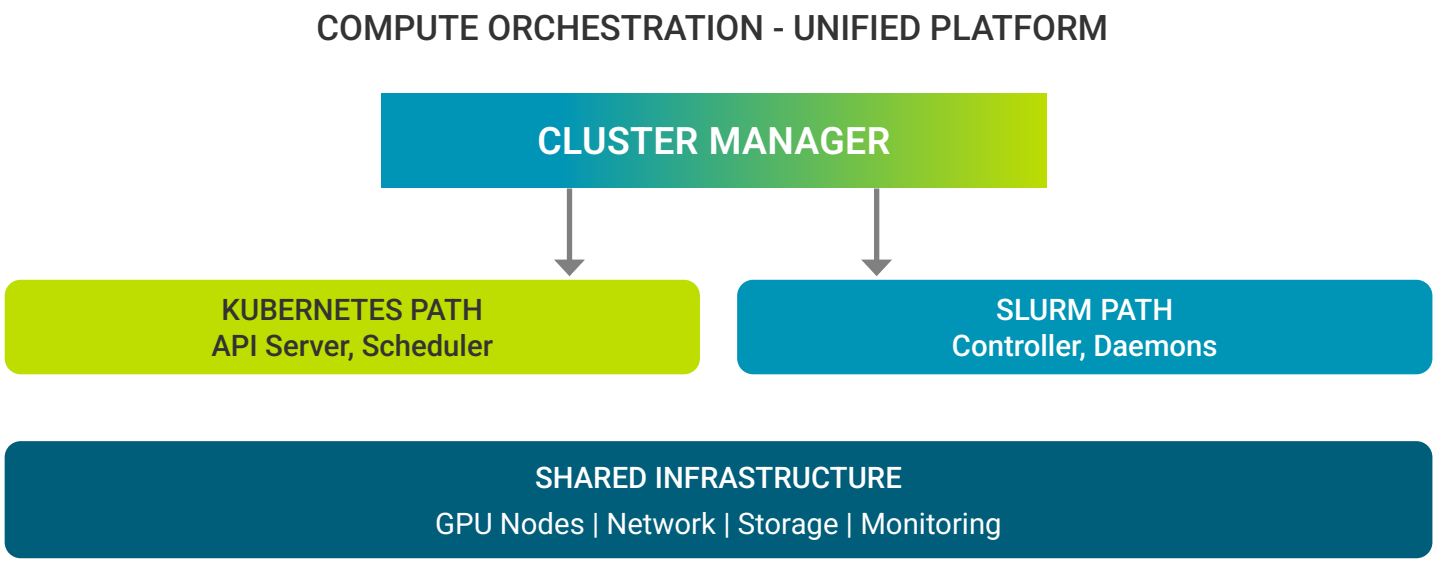


Exhibit 3b: Compute Orchestration - Unified platform supporting both Kubernetes and Slurm paths.

KUBERNETES

Powered by CKP (Coredge Kubernetes Platform), handles containerized workloads using declarative scheduling. Supports CNCF Certified Kubernetes versions (1.29 - 1.35). GPU resources are exposed via device plugins and scheduled using native Kubernetes primitives.

SLURM

Handles batch workloads with GPU-aware scheduling using GRES. Supports fair-share scheduling and detailed job accounting.

Interaction Model

The Cluster Manager provisions Kubernetes first, then optionally deploys Slurm as an overlay. Both share the same underlying nodes, storage, and network.

Guarantee	Mechanism
Consistent scheduling	K8s scheduler + Slurm controller
GPU-aware placement	Device plugins + GRES
Workload isolation	Namespaces + cgroups v2

KEY INSIGHT

Supporting both Kubernetes and Slurm enables the platform to serve both cloud-native and HPC workloads without forcing a trade-off.

Provisioning Workflow

The provisioning lifecycle is fully automated from user request to node readiness.

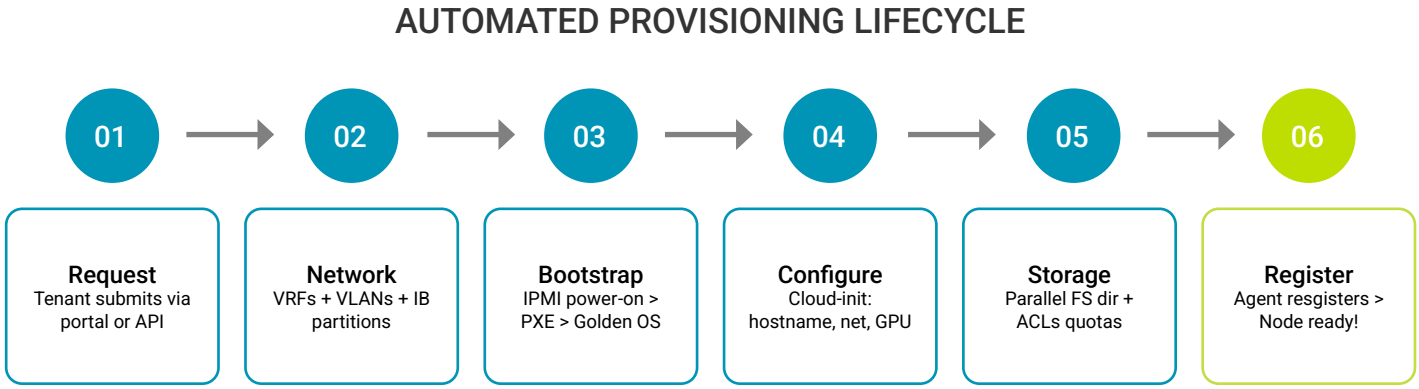


Exhibit 4: Automated bare metal provisioning lifecycle.

Step	Phase	Description
01	Request	Tenant submits via portal or API
02	Network	VRFs + VLANs + IB partitions allocated
03	Bootstrap	IPMI power-on > PXE boot > Golden OS
04	Configure	Cloud-init: hostname, networking, GPU agent
05	Storage	Parallel FS directory + ACLs + quotas
06	Register	Agent registers with portal > Node ready

Cluster Lifecycle

The cluster lifecycle is managed through a declarative state machine:

Step	Phase	Description
01	Cluster Request	Specification submitted via API or portal
02	Control Plane Init	API server, etcd, scheduler bootstrapped
03	Worker Attach	GPU nodes joined to cluster
04	GPU Runtime	Drivers, device plugins, operators deployed
05	Validation	Health checks and conformance tests
06	Operational	Cluster ready for workloads

Workload Execution

The workload execution model covers the full lifecycle from job submission to billing.

WORKLOAD EXECUTION MODEL

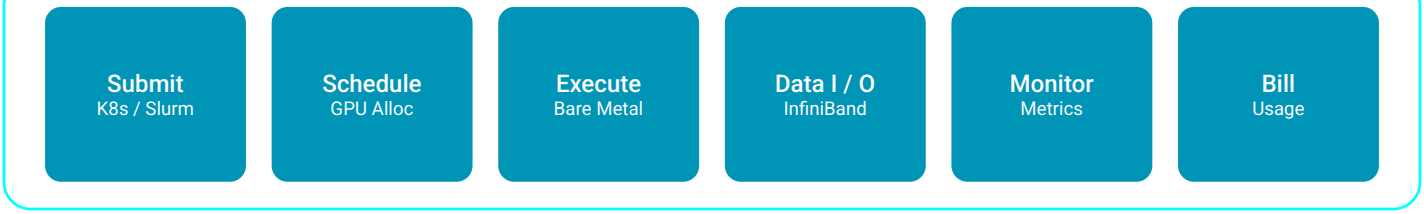


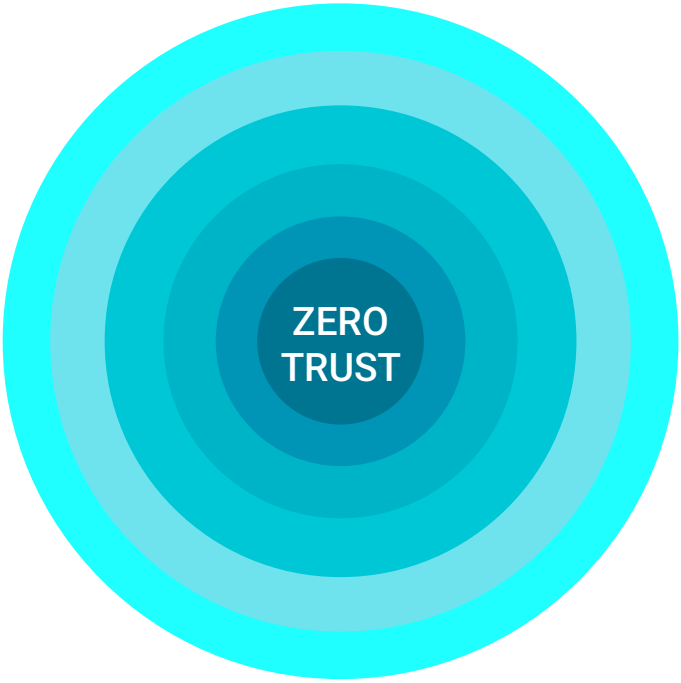
Exhibit 5: End-to-end Workload execution model.

Phase	K8s Path	Description
Submit	kubectl apply / API	sbatch / srun
Schedule	kube-scheduler + device plugin	Slurm controller + GRES
Execute	Pod on bare metal node	Job on compute node
Data I/O	PV mount via RDMA client	Direct mount via RDMA
Monitor	Prometheus + GPU exporter	Slurm accounting + telemetry
Bill	Metering pipeline per namespace	Metering pipeline per account

Security Architecture

Dflare AI implements a defense-in-depth security model based on zero-trust principles.

DEFENSE-IN-DEPTH SECURITY MODEL



- Perimeter Firewall + ACLs
- Network: VRF + VLAN + IB Partition Key
- Transport: TLS 1.2+ / mTLS
- Auth: OAuth2 / RBAC + ABAC
- Compute: Namespaces / cgroups v2
- Audit Immutable logs

Compliance alignment

NIST 800-53 | ISO 27001 | HIPAA

Exhibit 6: Concentric defense-in-depth security model.

Layer	Technology	Implementation
Identity	OAuth2 / OIDC	JWT tokens with per-tenant realms
Authorization	RBAC + ABAC	Scoped tokens at every API boundary
Network	VRF + VLAN + PKey	Hardware-enforced isolation
Data	Storage ACLs	Per-tenant filesystem controls
Transport	TLS 1.2+ / mTLS	Encrypted service-to-service

DEFENSE-IN-DEPTH LAYERS

1. **Perimeter** — Firewall + ACLs + Edge Protection
2. **Network** — VRF + VLAN + InfiniBand Partition Key
3. **Transport** — TLS 1.2+ / mTLS / Certificate Lifecycle
4. **Authentication** — OAuth2 / RBAC + ABAC / Per-Tenant IAM
5. **Compute** — Namespaces / cgroups v2 / Resource Quotas
6. **Zero Trust** — Immutable Audit + Continuous Verification

Observability

METRICS

GPU utilization

Cluster health

Job performance

MONITORING STACKS

Metrics collectors

Time-series database

Dashboards

KEY INSIGHT

Real-time observability across compute, network, and storage layers enables proactive capacity management and rapid incident response.

Billing and Metering

MEASURED RESOURCES

GPU hours

CPU hours

Storage usage

Network usage

PIPELINE

Metrics > Aggregation > Billing records

Every resource consumption event is tracked with tenant, project, and user attribution, providing granular chargeback capabilities.

Scalability Model

SCALABILITY MODEL - LEAF-SPINE ARCHITECTURE

01 **Horizontally scalable GPU nodes**

02 **Fabric expansion via leaf-spine architecture**

03 **Multi-tenant resource pooling**

The leaf-spine network topology enables horizontal scalability by adding spine switches and leaf pairs as needed.

High Availability

The platform is designed for resilience across all layers.

Control Plane

Replicated services | HA databases Leader election

Data Plane

Node-level isolation | Job rescheduling | Health checks

Network

Redundant fabric | Dual uplinks | Path failover

Key Differentiators

Bare Metal GPU Cloud.

Direct GPU access without virtualization overhead. Hardware-level BIOS and OS tuning pre-applied via golden images.

Unified Kubernetes + Slurm.

Run both containerized and Slurm workloads on the same bare metal infrastructure with unified networking, storage, security, and billing.

Hardware-Level Tenant Isolation

Isolation at InfiniBand switch hardware (partition key), filesystem (access control map), and network fabric (VRF/VXLAN).

InfiniBand-Native Architecture

Purpose-built for high-performance GPU-to-GPU and GPU-to-storage communication via RDMA.

Automated Lifecycle Management

From bare metal power-on to production cluster — fully automated. No SSH, no manual configuration.

ML Platform.

Integrated machine learning environment with GPU notebooks, distributed training, LLM inference, fine-tuning, experiment tracking, and dataset management — enabling complete ML lifecycle within workspace isolation.

KEY INSIGHT

Dflare AI uniquely combines unified K8s and HPC orchestration, bare metal performance, hardware-enforced isolation, and full lifecycle automation in a single platform — a combination not available from any single public cloud provider.

Conclusion

Dflare AI provides a unified platform for AI infrastructure, combining cloud-native orchestration with HPC-grade performance. The architecture enables organizations to scale AI workloads efficiently while maintaining strict isolation, high performance, and operational simplicity.

Deployment Architecture

Dflare AI deployments follow a rack-scale topology optimized for density, redundancy, and performance.

Exhibit 9: Rack-level deployment topology with dual-fabric architecture

Key Differentiators

Rack-Level Design

Each rack contains:

- 8-16 GPU compute nodes with direct PCIe Gen5 attached GPUs
- Top-of-Rack (ToR) Ethernet switch for control plane and tenant traffic
- Dedicated InfiniBand switch for high-performance data fabric
- Shared storage nodes (optional per-rack or shared across pods)

Fabric Topology

- **Ethernet:** Leaf-spine with VXLAN/EVPN for multi-tenancy
- **InfiniBand:** Fat-tree Clos for non-blocking bisection bandwidth
- **Design rationale:** Leaf-spine provides deterministic latency and ECMP; IB fat-tree provides full bisection for all-to-all communication

High Availability & Failure Handling Deep Dive

Dflare AI provides no single point of failure through systematic redundancy and isolation.

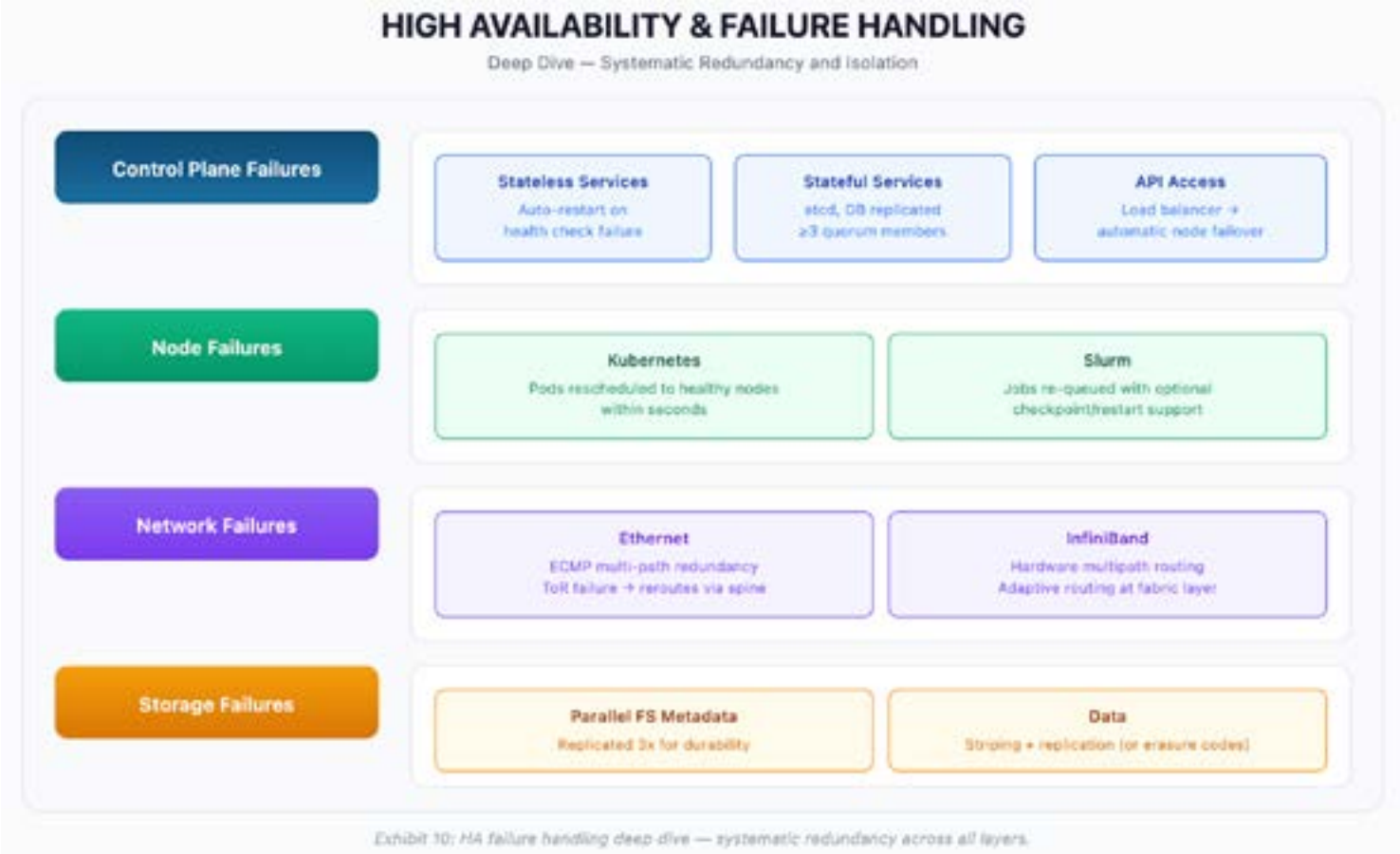


Exhibit 10: HA failure handling deep dive — systematic redundancy across all layers.

Key Differentiators

Control Plane Failures

- Stateless services auto-restarted on health check failure
- Stateful services (etcd, databases) replicated across ≥3 quorum members
- API accessed through load balancer automatic node failover

Node Failures

- **Kubernetes:** Pods rescheduled to healthy nodes within seconds
- **Slurm:** Jobs re-queued with optional checkpoint/restart support

Network Failures

- **Ethernet:** ECMP provides multi-path redundancy; ToR switch failure traffic reroutes via spine
- **InfiniBand:** Hardware multipath routing with adaptive routing at fabric layer

Storage Failures

- **Parallel FS:** Metadata replicated 3x for durability
- **Data:** Striping + replication (or erasure codes) across targets

Scalability Model (Production Scale)

Dflare AI scales horizontally across all dimensions without architectural bottlenecks.

PRODUCTION SCALE DIMENSIONS

Horizontal Scaling Without Architectural Bottlenecks

Dimension	Capacity	Scaling Mechanism
GPU Nodes	0 to 10,000+ nodes	Add nodes linearly — no control plane bottleneck
Network Bandwidth	400 GB/s per-node	Leaf-spine expands with additional spine switches
Storage Throughput	100 GB/s aggregate	Add storage nodes to increase throughput and capacity
Control Plane	10,000+ nodes supported	Proven scheduler managing large-scale GPU clusters

Exhibit 11: Production scale dimensions — horizontal scaling at every layer.

Horizontal Scaling

- **GPU nodes:** Add nodes linearly — no control plane bottleneck
- **Network:** Leaf-spine expands by adding spine switches and leaf pairs
- **InfiniBand:** Additional tiers added for larger fabrics
- **Storage:** Add storage nodes to increase aggregate throughput and capacity

Observed Limits

- **Scheduler:** Proven to manage 10,000+ GPU nodes
- **IB fabric diameter:** Typically 3-4 hops for production scales
- **Metadata bottleneck:** Parallel FS with dedicated metadata servers

Production Scale Dimensions

Dimension	Capacity
GPU Nodes	0 to 10,000+
Network Bandwidth	Per-node 400GB/s
Storage Throughput	100GB/s aggregate
Control Plane Capacity	Support 10k+ nodes

Performance Characteristics

Dflare AI is purpose-built for performance-critical AI workloads.

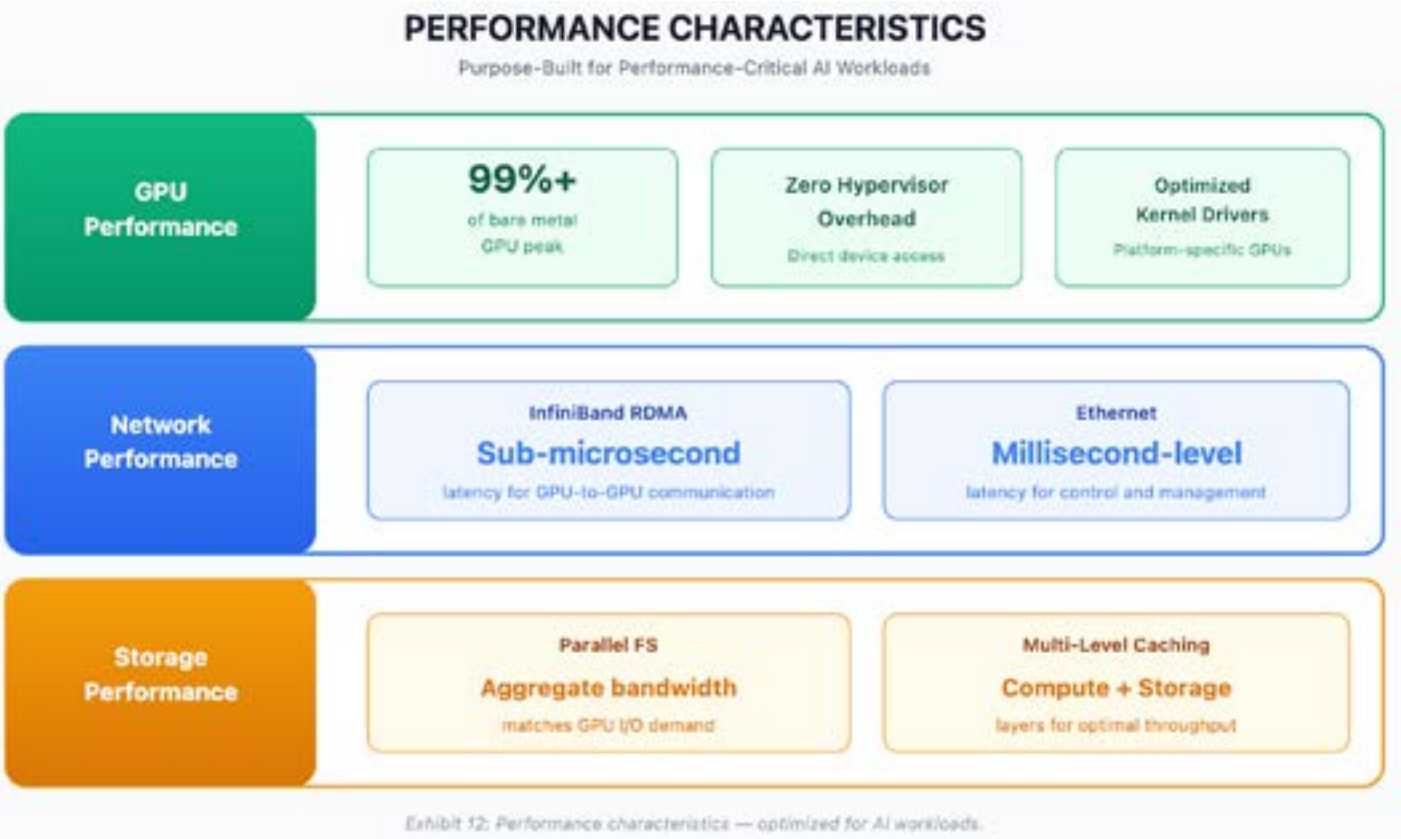


Exhibit 12: Performance characteristics — optimized for AI workloads.

GPU Performance

- Near-native efficiency (99%+ of bare metal GPU peak)
- Zero hypervisor overhead — direct device access
- Optimized kernel drivers for platform-specific GPUs

Network Performance

- **InfiniBand RDMA:** Sub-microsecond latency for GPU-to-GPU communication
- **Ethernet:** Millisecond-level latency suitable for control and management

Storage Performance

- **Parallel FS:** Aggregate bandwidth matches GPU I/O demand
- **Caching:** Multi-level caching at compute and storage layers

Reference Workloads

Dflare AI is optimized for a range of production AI and HPC workloads.

Workload	Characteristics	Key Requirement
Distributed Training	Multi-node, multi-GPU synchronous training	InfiniBand RDMA critical for allreduce operations
Large Model Fine-tuning	Distributed data and model parallelism	High GPU utilization, moderate I/O
Inference at Scale	Kubernetes microservices, auto-scaling	Low latency, variable load
HPC Batch	Long-running Slurm batch jobs, checkpointing	Throughput optimized, fault tolerance
Scientific Computing	Domain-specific codes with heavy compute	Near-bare-metal performance required

User Interaction Model

Dflare AI exposes infrastructure through a declarative, API-first model supporting both enterprise and developer workflows.

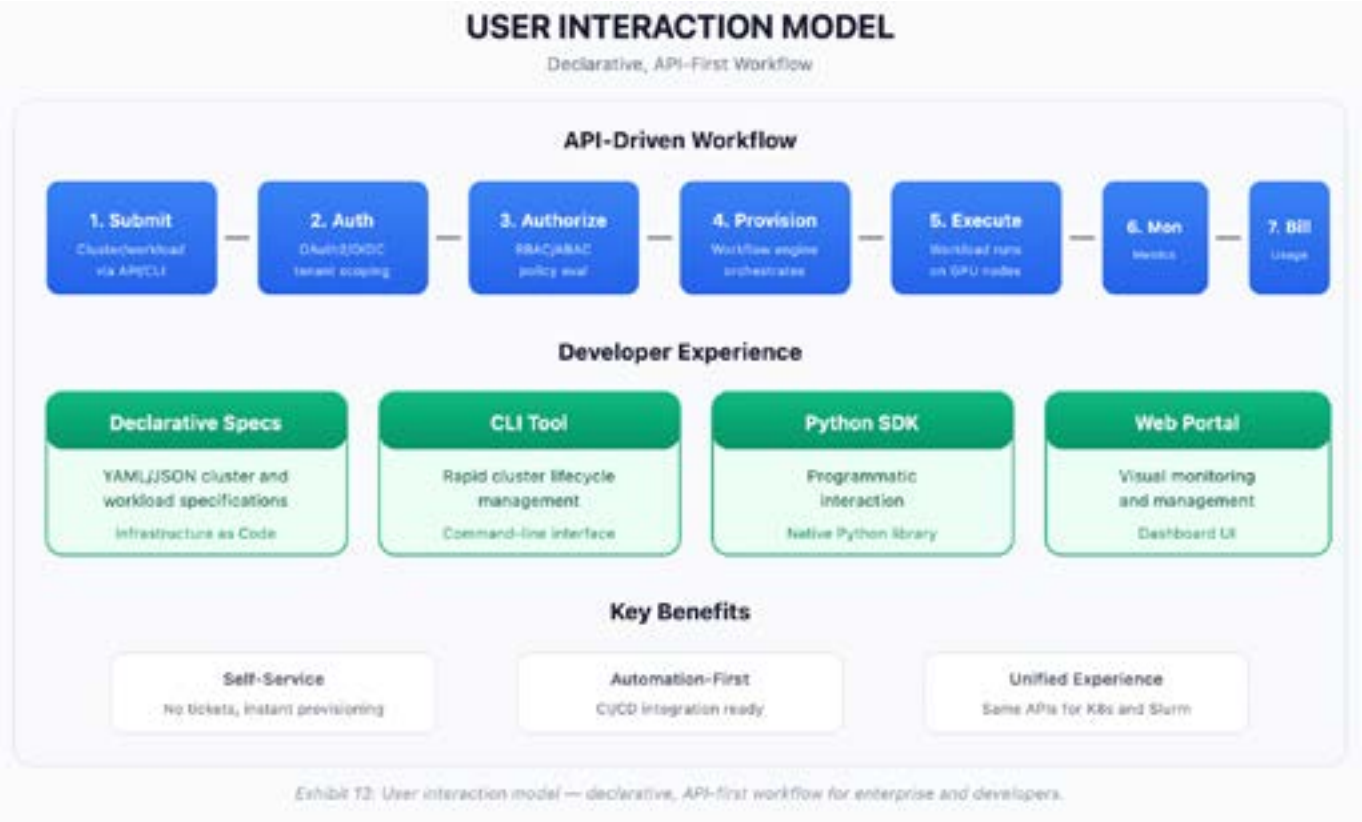


Exhibit 12: User interaction model — declarative, API-first workflow for enterprise and developers.

Reference Workloads

API-Driven Workflow

- Submit: User submits cluster or workload specification via API/CLI
- Authenticate: OAuth2 / OIDC with tenant scoping
- Authorize: RBAC/ABAC policy evaluation at control plane
- Provision: Workflow engine orchestrates infrastructure setup
- Execute: Workload scheduled and runs on GPU nodes
- Monitor: Metrics collected and dashboards updated
- Bill: Usage aggregated and chargeback records generated

Developer Experience

- Declarative cluster and workload specs (YAML/JSON)
- CLI tool for rapid cluster lifecycle management
- Python SDK for programmatic interaction
- Web portal for visual monitoring and management

Final Summary

Dflare AI represents a paradigm shift in enterprise GPU infrastructure. By unifying cloud-native orchestration (Kubernetes), HPC batch scheduling (Slurm), bare metal performance, hardware-enforced isolation, and complete lifecycle automation, Dflare AI enables organizations to:

- Scale AI training from 8 to 10,000+ GPUs without architectural changes
- Run both cloud-native microservices and long-running batch jobs on the same infrastructure
- Enforce strict multi-tenancy through hardware and software controls
- Achieve near-bare-metal performance with cloud-like operational simplicity
- Reduce total cost of ownership through unified infrastructure and automation

Dflare AI is the unified operating system for enterprise GPU infrastructure — delivering bare metal performance, cloud-native agility, and HPC-grade reliability in a single platform.

Get in touch with us

https://coredge.io info@coredge.io