Reimagining Container Runtimes

Security Without Sacrifice

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

EDERA

Agenda

- Introduction
- The Container Security Problem
- Edera's Different Approach
- Demo
- Closing

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Show of hands

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Modern container platforms force Organisations to choose between ***security***, ***performance***, *and* ***resource utilisation***.

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

The Container Security Problem

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | Security vs. Performance vs. Resource Utilization

The Core Problem: Modern container platforms force Organisations into an impossible choice between three critical needs:

- **Security**: Protection against container escapes and multi-tenant isolation
- **Performance**: Near-native speed for production workloads
- **Resource Utilization**: Efficient use of expensive infrastructure

Why This Choice Exists:

- Standard containers share a kernel for performance but sacrifice security
- Secure alternatives (gVisor, Kata) add significant performance overhead
- Resource allocation inefficiencies stem from static resource assignment

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | Resource Waste Crisis

The 60% Problem: Standard containers are fast but vulnerable and waste up to 60% of infrastructure resources through inefficient resource allocation

Root Causes:

- **Static Resource Allocation:** Containers get fixed CPU/memory allocations that can't adapt
- **Over-provisioning:** Teams allocate resources for peak loads, leaving them idle most of the time
- **Lack of Dynamic Scaling:** Traditional runtimes can't efficiently allocate resources between workloads
- **No Live Migration:** Can't move containers to optimize resource usage across nodes

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | The Security Vulnerability Reality

Recent Attack Surface: Between 2022 and 2024 alone, seven significant container escape vulnerabilities were discovered, each exploiting the fundamental issue of shared kernel state

Why These Attacks Succeed:

- **Shared Kernel Problem:** Traditional container technologies provide what we call "weak isolation" - controls implemented within a shared kernel
- **Large Attack Surface:** All containers share the same kernel, creating a single point of failure
- **Namespace Limitations:** Linux namespaces were never intended as hard security boundaries

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | Container Escape CVEs (2022–2024)

CVE-2022-0185	Linux kernel vulnerability
CVE-2022-0492	cgroup release_agent bypass allowing privilege escalation
CVE-2022-0811	(cr8escape) - Container escape vulnerability
CVE-2022-0847	(Dirty Pipe) - Unprivileged users to write to read-only pages
CVE-2022-23648	Container runtime vulnerability
CVE-2024-0132	GPU driver vulnerability
CVE-2024-21626	(Leaky Vessels) - runc vulnerability providing access host filesystem

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | Alternative Solutions Fall Short

Hardware-Dependent Options:

- **Kata Containers/Firecracker:** Require virtualization extensions not available on all hardware
- **Only 7% of AWS instances** include virtualization extensions
- **Cost Impact:** Specialized hardware is significantly more expensive

Performance Trade-offs:

- **gVisor:** User-space kernel approach creates substantial overhead
- **Traditional VMs:** Heavy resource consumption and slow startup times
- **Unikernel Approaches:** Require application rebuilding and limit flexibility

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | The Developer Experience Problem

Complexity Burden example: Sovereign Infrastructure (specifically AI)

- SPIFFE/SPIRE Solutions:
  - Significant complexity challenges that impede adoption.
  - The operational overhead of certificate management, agent deployment, and attestation configuration taxes already stretched security teams

Ecosystem Fragmentation:

- Security solutions often break compatibility with existing Kubernetes tooling
- Teams need separate infrastructure for secure vs. standard workloads
- Additional operational overhead for managing multiple systems

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Traditional Approach Limitations | Real-World Impact

Business Consequences:

- Organisations run most workloads without strong isolation due to performance costs
- Security-critical applications get isolated on expensive, specialized infrastructure
- Global average cost of a data breach reaching $4.9 million in 2024—a 10% increase over the previous year
- Teams can't leverage cloud economics for sensitive workloads

The Status Quo Problem: This segmentation leaves most workloads vulnerable to container escapes while forcing organisations to maintain multiple, incompatible infrastructure stacks.

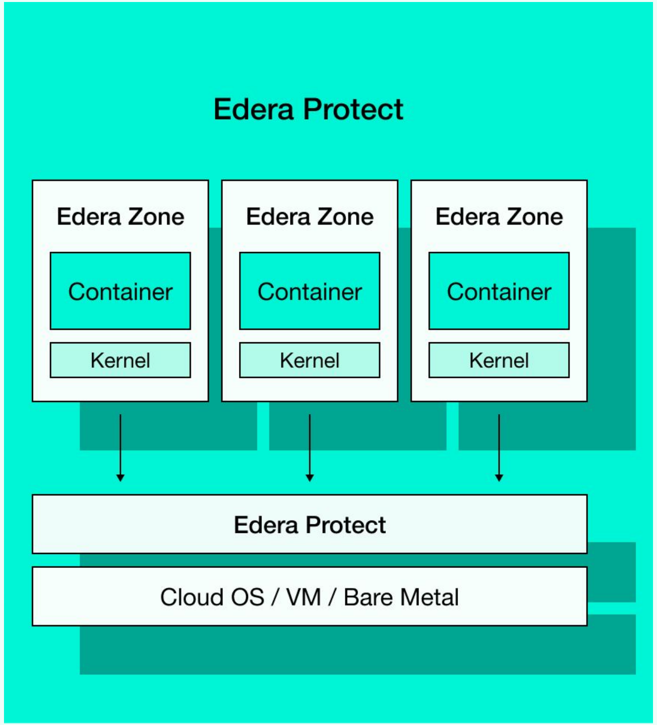
Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera's Different Approach

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

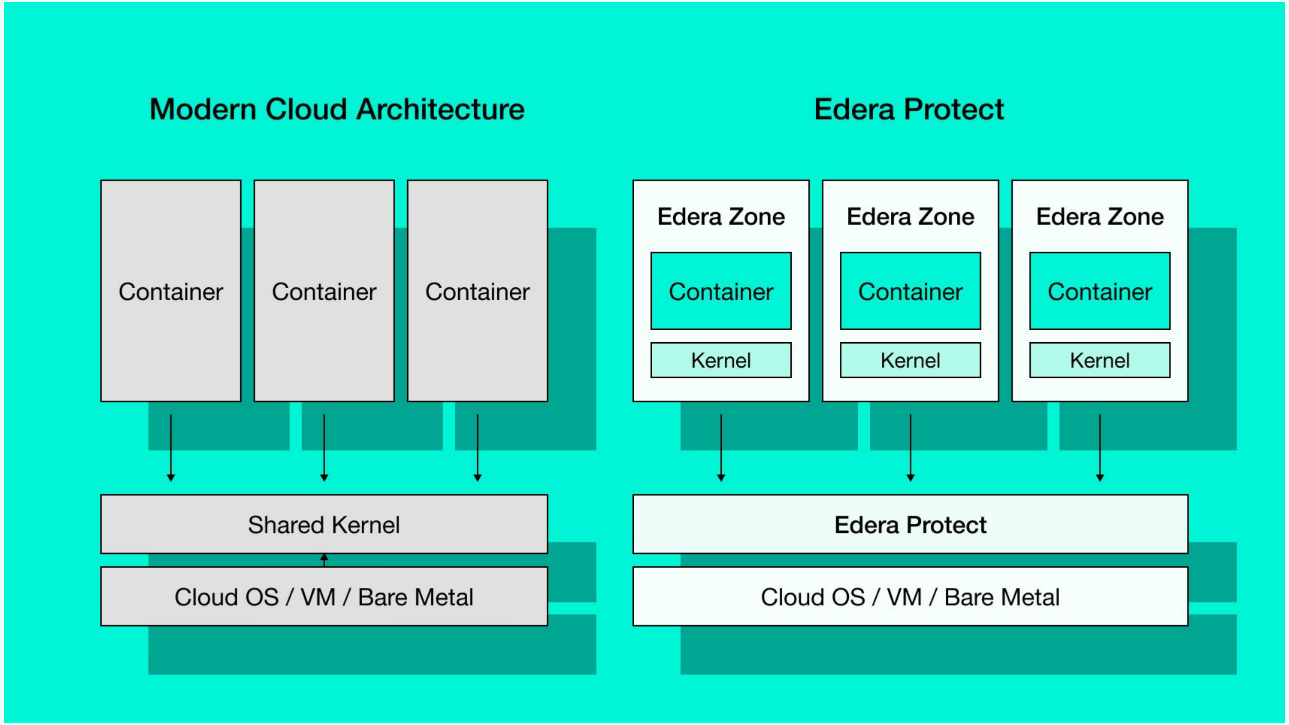
Edera's Different Approach | Container-Native Hypervisor: A New Paradigm



EDERA

https://edera.link/privileged

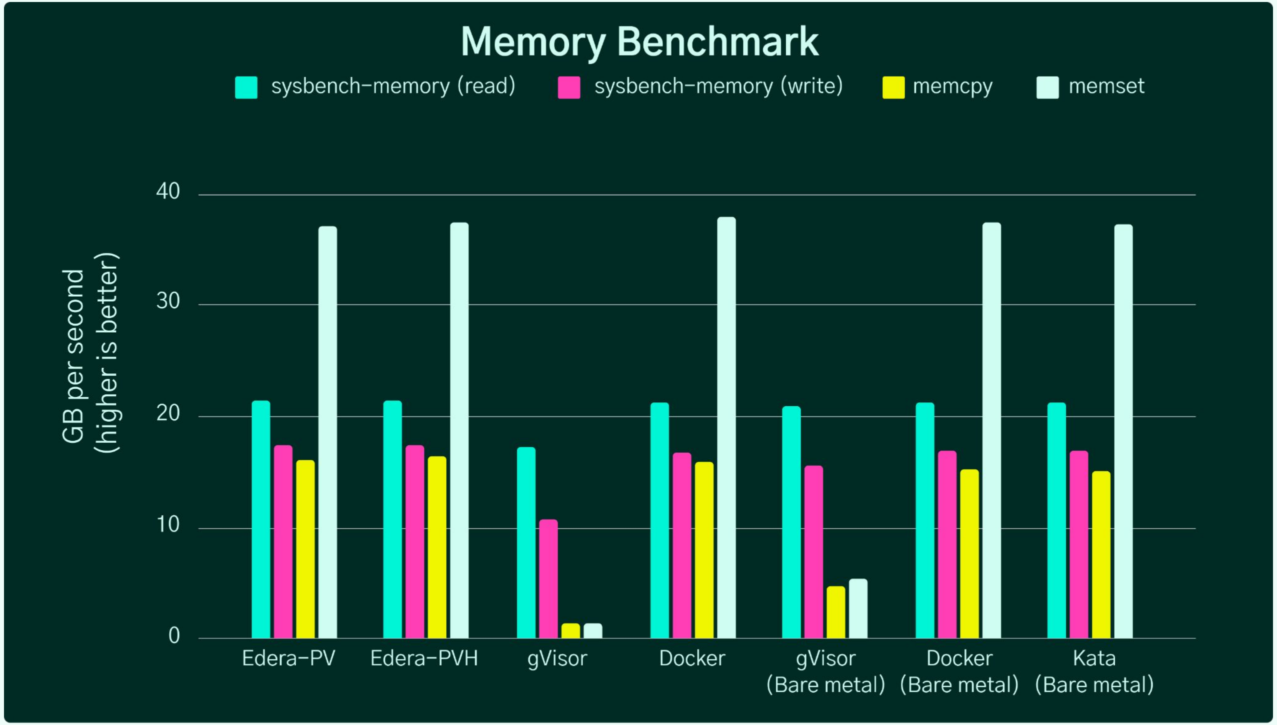
Edera's Different Approach | From Weak to Strong: A Clear Definition



EDERA

https://edera.link/privileged

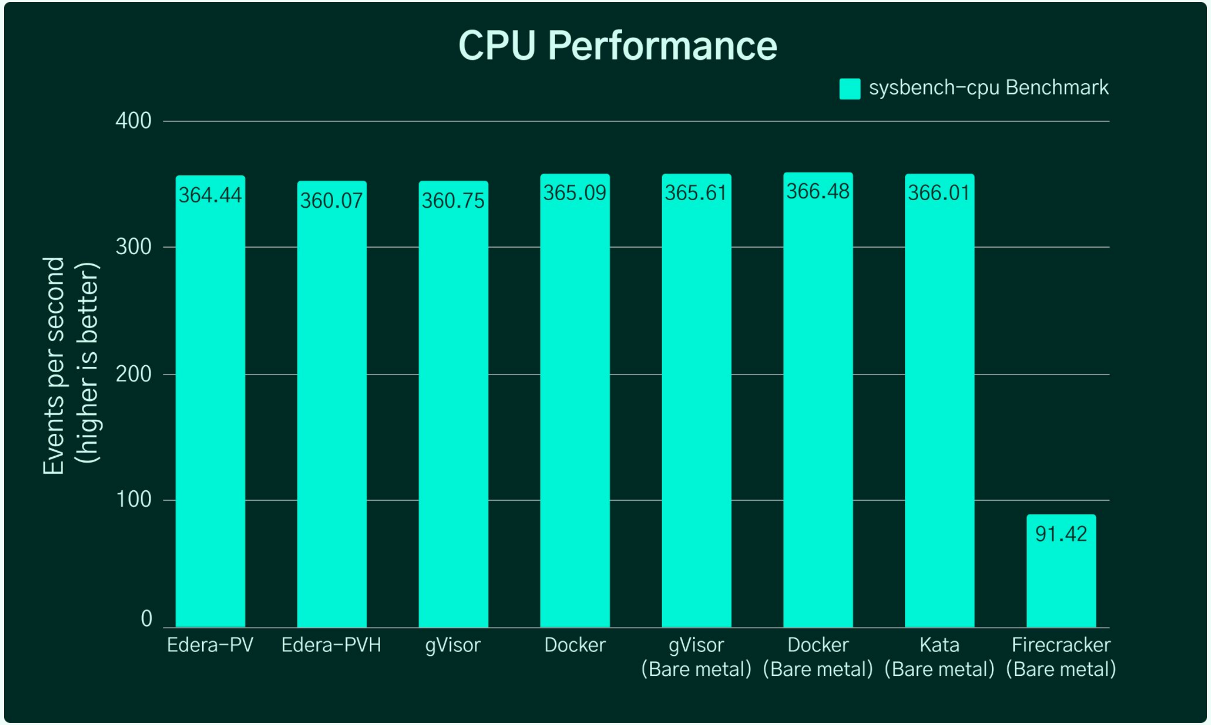
Edera's Different Approach | Benchmarks Don't Lie: Security That Actually Performs



EDERA

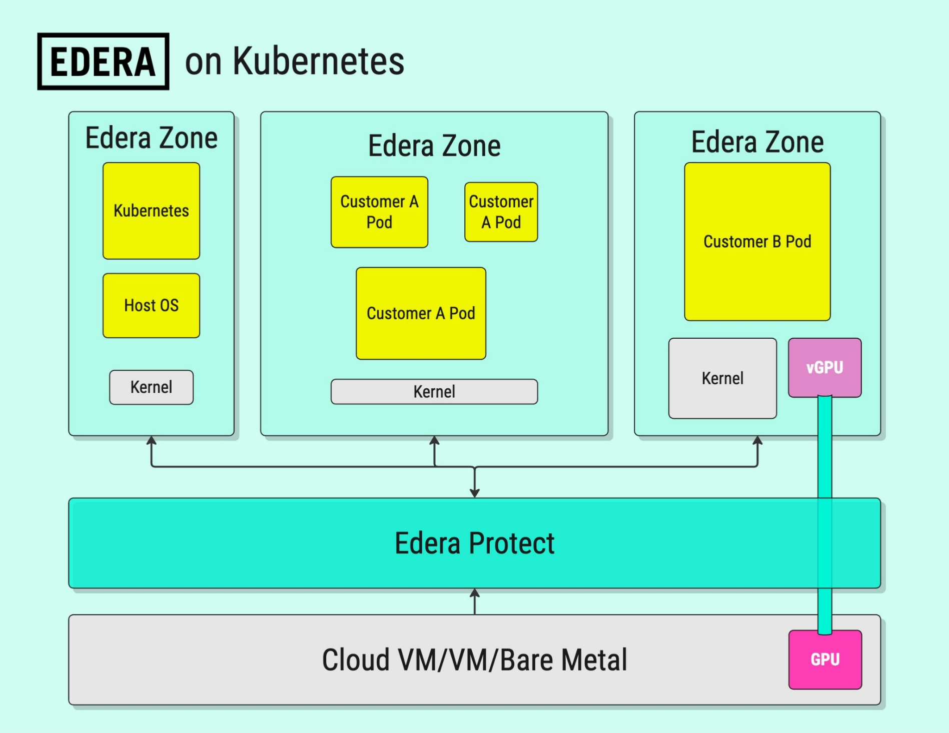
https://edera.link/benchmarks

Edera’s Different Approach | Benchmarks Don’t Lie: Security That Actually Performs



https://edera.link/benchmarks

Edera's Different Approach | How It Works: Zones and Hypervisor Design



https://edera.link/diagram

Edera’s Different Approach | Runs Everywhere: No Special Hardware Required

The Industry Problem:

- Kata Containers/Firecracker require virtualization extensions
- Only 7% of AWS instances include these extensions
- Specialized hardware significantly more expensive

Edera's Solution:

- Uses paravirtualization for performance improvements
- Works on any commodity hardware
- Optional PVH mode for systems with virtualization extensions

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera’s Different Approach | The Impossible Made Possible: Secure Privileged Mode

Traditional Risk:

- Privileged containers bypass isolation mechanisms
- Full access to host system
- Primary vector for container escapes

Edera's Innovation:

- Privileged mode support
- Containers requiring elevated privileges run in isolated zones
- Strong security boundaries maintained

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera's Different Approach | Securing the AI Infrastructure: GPU Driver Isolation

The GPU Problem:

- GPU drivers are complex, proprietary code (gigabytes)
- 20+ CVEs in GPU drivers in 2024 alone
- Driver bugs can corrupt host kernel

Edera's Solution:

- GPU drivers run in isolated zones
- Applications communicate via NVIDIA vGPU
- Driver compromise only affects the GPU zone

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera’s Different Approach | Drop-In Compatibility

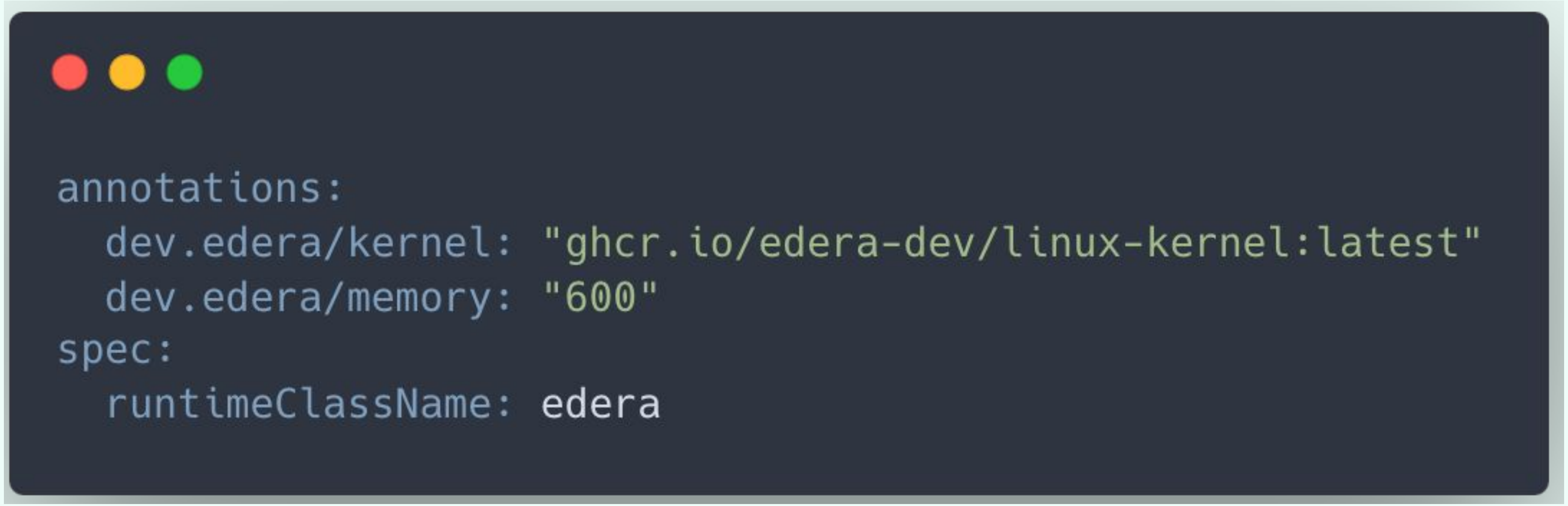
Zero Disruption Deployment:

- Seamless integration through simple runtime class
- Compatible with existing Kubernetes tooling
- No changes to developer workflows

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera’s Different Approach | Ship with Containers, Run with Edera



```
annotations:
  dev.edera/kernel: "ghcr.io/edera-dev/linux-kernel:latest"
  dev.edera/memory: "600"
spec:
  runtimeClassName: edera
```

EDERA

https://carbon.now.sh

Edera’s Different Approach | Beyond Runtime: A Platform Approach

Edera Protect Kubernetes:

- Container isolation and FIPS kernel support
- Autoscale and dynamic workload resources
- Live container migration & memory ballooning

Edera Protect AI:

- Driver isolation and secure vGPUs
- Support for all hardware accelerators (GPUs, TPUs, DPUs)
- Confidential computing support

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera’s Different Approach | Security Without Sacrifice: The New Standard

What We've Achieved:

- Strong isolation without performance penalties
- Universal hardware compatibility
- Kubernetes ecosystem preservation
- Revolutionary GPU security

Industry Impact:

- Eliminates the security vs. performance trade-off
- Enables secure multi-tenancy at scale
- Unlocks AI workloads in regulated environments

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Apple Just Validated Our Approach

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Apple Just Validated Our Approach | The Dev-to-Prod Security Gap is Now Obvious

What Apple Built:

- Open-sourced their own version of Kata Containers for macOS
- Written entirely in Swift using Containerization Framework
- Provides hypervisor-isolated containers for development

What This Validates:

- **Hypervisor-level isolation** is the right approach
- **Sub-second container start times** with full isolation are possible
- **No performance trade-offs** needed for security

EDERA

https://edera.link/apl-container

Demo

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

https://demo.edera.dev
passphrase: *feeltheteal*

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

One more thing…

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Closing

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

Edera Resources

White Paper: Deeper dive into Edera : https://edera.link/whitepaper

Diagram: Architecture of Edera: https://edera.link/diagram

One-pager: High level "What does Edera do": https://edera.link/one-pager

Why Edera: Why isolation matters: https://edera.link/defining-iso

AI Security: What we're doing with AI: https://edera.link/sovereign-ai

Privileged: How Edera works with privileged containers: https://edera.link/privileged

Styrolite OSS: Edera's open source container runtime: https://edera.link/styrolite-oss

Benchmarks: Edera benchmarks: https://edera.link/benchmarks

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev

THANK YOU

EDERA

Lewis Denham-Parry
Staff Solutions Engineer
lewis@edera.dev