Nexus: A Declarative, Multi-Paradigm Bare-Metal Infrastructure Testbed

Abstract Submission

Simon Volpert*, Robert Keßler*, Marlon Engels*, Max Stempel*, Paul Suhr*, and Stefan Wesner*
*PDS, University of Cologne
Cologne, Germany

Abstract—Contemporary research environments demand the integration of heterogeneous computing paradigms. This extended abstract introduces Nexus, an incubator research project and empirical testbed designed to deploy complex, dynamic bare-metal infrastructure. Nexus bridges bare-metal immutability with cloud-native orchestration to enable the automated convergence of disjoint computational paradigms onto unified, reproducible hardware.

I. Introduction

Modern computational research environments require a diverse set of computing paradigms to operate efficiently. These range from traditional High-Performance Computing (HPC) for large-scale simulations to Cloud Computing frameworks like Kubernetes for microservices, and Big Data Analytics (BDA) platforms such as Apache Spark. In the past, these paradigms were relegated to statically partitioned, disjoint physical clusters. This rigid separation leads to resource underutilization when computational demand shifts between different types of workloads. Furthermore, manually re-provisioning bare-metal servers to adapt to these shifts is slow, error-prone, and hinders scientific experimentation. Testing complex, multi-paradigm scheduling is difficult without a dynamic physical testbed that can transition hardware roles.

To address these limitations, we introduce Nexus. Nexus is an incubator research project and proof-of-concept infrastructure that acts as a foundational enabler for researching the dynamic interaction, structural malleability, and convergence of disparate distributed systems. Rather than serving as a production environment, Nexus is designed as a scientific instrument for validating novel concepts, conducting empirical research, and serving as an educational sandbox. It provides students and researchers with hands-on opportunities to explore system interactions, isolation boundaries, and advanced scheduling behaviors.

II. Core Principles of the Architecture

The Nexus architecture bridges declarative infrastructure and bare-metal provisioning by adhering to four core design principles. In brief, we utilize the following principles:

1) Immutability: Enforced by leveraging NixOS. This ensures that every server configuration is reproducible, establishing a deterministic software environment that aligns with computational reproducibility best practices.

2) Idempotency: The system utilizes GitOps reconciliation mechanisms to converge the cluster toward a declared target state. This satisfies artifact evaluation guidelines by ensuring that the desired state and the actual state of the infrastructure are always synchronized.

3) Impermanence: Nexus employs ephemeral Preboot Execution Environment (PXE) boot sequences paired with in-memory (tmpfs) root filesystems. This approach minimizes accumulated state drift across re-provisioning cycles, ensuring that all scientific experiments execute within clean operating environments.

4) Declarative Delivery: The entire computational stack—from base operating systems and bare-metal firmwares to user-facing research applications—is defined as code. This eliminates ad-hoc manual interventions and yields an auditable provisioning pipeline.

III. Cloud-Native Malleability

By enforcing immutability and impermanence at the fundamental node level, researchers using Nexus can reconfigure shared physical hardware across mutually exclusive computational paradigms without leaving lingering state artifacts. The infrastructure facilitates the declarative definition of complex, heterogeneous topologies, as illustrated on the accompanying poster.

Nexus supports diverse research configurations on a unified hardware base. It is capable of provisioning conventional HPC schedulers, such as Slurm, and analytics frameworks via NixOS to achieve maximum performance. Simultaneously, it can deploy a foundational Kubernetes substrate to orchestrate containerized HPC workloads alongside standard microservices. Furthermore, Nexus leverages virtualization technologies and nested cluster APIs to provision isolated tenant environments. This forms the basis for providing Reproducibility-as-a-Service (RaaS) on demand. The bare-metal substrate (Layer 1) and the orchestration abstractions (Layer 2) are decoupled and managed via a unified GitOps pipeline, ensuring auditable and extensible workflows across the entire stack.

IV. Vision and Planned Research

The primary goal of the Nexus cluster is to enable the empirical measurement of the dynamic interaction

between divergent systems. Declarative node provisioning enables high state consistency, structural malleability, and strict tenant isolation, as detailed in the capability grid on the accompanying poster.

Nexus serves as the foundational physical testbed for ongoing research in two primary domains, alongside its role in HPC education. The first domain is the isolation of co-located HPC workloads, investigating isolation mechanisms, resource contention, and performance interference when executing traditional HPC tasks on shared infrastructure. The second domain explores the dynamic elasticity and malleability of HPC clusters when augmented with cloud-native paradigms and big data analytics. Furthermore, the cluster provides a safe, reproducible environment for students to learn bare-metal provisioning. By treating physical node roles as declarative code, Nexus converges disjoint research workflows and educational projects onto unified hardware, pushing the boundaries of reproducible computational science.