

< [Nomad Home](#)

≡ Filter sidebar

Documentation

Install Nomad

Upgrading

Release Notes

Integrations

Concepts

Configuration

Commands (CLI)

Glossary

Job Specification

Other Specifications

Task Drivers

Device Plugins

Schedulers

Developer / Nomad / Documentation / **Nomad vs. Kubernetes**

v1.9.x (latest)

Nomad vs. Kubernetes

Interested in talking with HashiCorp about your experience building, deploying, or managing your applications? [Set up a time to chat!](#)

Kubernetes is an orchestration system for containers originally designed by Google, now governed by the Cloud Native Computing Foundation (CNCF) and developed by Google, Red Hat, and many others. Kubernetes and Nomad support similar core use cases for application deployment and management, but they differ in a few key ways. Kubernetes aims to provide all the features needed to run Linux container-based applications including cluster management, scheduling, service discovery, monitoring, secrets management and more. Nomad only aims to focus on cluster management and scheduling and is designed with the Unix philosophy of having a small scope while composing with tools like Consul for service discovery/service mesh and Vault for secret management.

The following characteristics generally differentiate Nomad from Kubernetes:

Simplicity

Kubernetes is designed as a collection of more than a half-dozen interoperating services which together provide the full functionality. Coordination and storage is provided by etcd at the core. The state is wrapped by API controllers which are consumed by other services that provide higher level APIs for features like scheduling. Kubernetes supports running in a highly available configuration but is operationally complex to setup.

Nomad is architecturally much simpler. Nomad is a single binary, both for clients and servers, and requires no external services for coordination or storage. Nomad combines a lightweight resource manager and a sophisticated scheduler into a single system. By default, Nomad is distributed, highly available, and operationally simple.

Flexible Workload Support

While Kubernetes is specifically focused on Linux containers, Nomad is more general purpose. Nomad supports virtualized, containerized and standalone applications, including Docker, Java, IIS on Windows, Qemu, etc. Nomad is designed with extensible drivers and support will be extended to all common drivers.

Consistent Deployment

A full Kubernetes installation for a production environment is time consuming, operationally complex, and resource intensive. An increasing number of implementations are created by the Kubernetes community to mitigate these challenges, such as minikube, kubeadm, k3s, and more. These trimmed versions of Kubernetes offer easier adoption for development and testing, but lead to inconsistency in capabilities, configuration, and management when moving into production.


In contrast to Kubernetes' fragmented distributions, Nomad as a single lightweight binary can be deployed in local dev, production, on-prem, at the edge, and in the cloud in a consistent manner, and provides the same operational ease-of-use across all environments.

Scalability

[Kubernetes documentation](#) states that they support clusters up to 5,000 nodes and 300,000 total containers. As the environment grows, the interoperating components with different constraints compound the operational complexity. [Even operators at Google revealed the significant challenges of managing the system at scale](#). The lack of maturity in the Federation project and the additional overhead of managing a centralized management plane also make it a hard experience to deploy a distributed system that spans multiple clusters.

Nomad has been proven to scale to cluster sizes that exceed 10,000 nodes in real-world production environments. It can be deployed across multiple availability zones, regions, and data centers with a single cluster or multiple clusters. Nomad is designed to natively handle multi-cluster deployments without the overhead of running clusters on clusters. This makes it easier to scale the application deployment across multiple datacenters, regions, and clouds with no additional complexity.

Nomad has performed strenuous benchmark on scalability with [1 million container challenge](#) in 2016 and [2 million container challenge](#) in 2020. These tests are aimed to validate Nomad's architectural design and ensure that Nomad performs under the most extreme requirements.

 [Edit this page on GitHub](#)

On this page:


Nomad vs. Kubernetes

Simplicity

Flexible Workload Support

Consistent Deployment

Scalability



Theme

System

Certifications

System Status

Cookie Manager

Terms of Use

Security

Privacy

Trademark Policy

Trade Controls

Accessibility

Give Feedback