

Container Technology

A comprehensive overview: how software is developed and deployed faster, simpler, and more reliably through container solutions

Contents

1. The Basics

2. The Implementation

3. Security and Network

4. Getting Your Team on Board

The Basics

1. WHAT IS CONTAINER TECHNOLOGY OR CONTAINER VIRTUALISATION?

A container is a standardised, self-contained software unit. Container virtualisation enables an application to run independently of the host operating system (OS). The container encapsulates an entire application – including code, dependencies, and configuration – in one clearly defined, portable format based on the **Open Container Initiative (OCI)** standard.

The OCI standard means that images built with any OCI-compliant tool (Docker, Podman, Buildah, Kaniko, etc.) run without modification on any OCI-compliant container runtime and orchestration platform, including Kubernetes.

2. WHAT ARE THE BENEFITS PROVIDED BY A CONTAINER SOLUTION?

The standardisation of a container and its independence from the host OS provide attractive benefits for developing and running software:

- **Applications scale more easily:** Container virtualisation is based on autonomous, independently functioning units. Applications can be scaled up or down quickly and flexibly as needed.
- **Containers start faster:** No dedicated guest operating system needs to boot up: startup times are measured in milliseconds, not minutes.
- **Containers use resources efficiently:** Without the overhead of a dedicated OS per application, containers actively save physical resources that are available for other workloads.
- **More agile software development:** Containers are fast to build and deploy, and they are reproducible: a container started in the development environment is identical to the one in production.
- **Containers are portable:** Container applications run largely independently of the host OS and can be migrated across infrastructure providers with minimal effort.

3. WHAT ARE THE DIFFERENCES BETWEEN CONTAINER SOLUTIONS AND VIRTUAL MACHINES (VMS)?

Containers and virtual machines represent two different approaches to hosting and running applications in a virtual environment. For VMs, a hypervisor distributes the server's physical resources to virtual machines, each requiring a dedicated operating system. Containers, on the other hand, share the host OS kernel: virtualisation is carried out at the operating system level. Therefore, containers use available resources significantly more efficiently than VMs and start up more quickly.

Containers and VMs are not mutually exclusive: most production Kubernetes environments run containerised workloads on top of virtual machine nodes, combining the isolation benefits of VMs with the operational efficiency of containers.

4. WHAT ARE THE RISKS ASSOCIATED WITH A CONTAINER SOLUTION?

Those who have looked into container technology may have encountered the following concerns:

"Containers are not secure enough for our requirements."

Containers are no less secure than any other environment. Security depends primarily on how the environment is set up and maintained: base image selection, network policies, Pod Security Standards, image scanning, and secret management all play a role. A well-configured container environment is often more auditable and consistent than traditional deployments.

"Container management is far too complex for us."

Just like any change, migrating to a container-based system comes with an initial investment. Applications need to be prepared, teams need to be trained, and deployment processes need to be automated. But this investment pays off quickly. The complexity can be significantly reduced by using a managed Kubernetes platform, allowing your team to focus on application development, not infrastructure management.

"We already have a deployment strategy. We don't need containers."

Containers do not replace a deployment strategy: they complement and optimise it. There are various options for integrating containers into an existing IT strategy, including alongside or supplemental to existing VM-based infrastructure.

The Implementation

5. WHAT SPECIFIC IMPROVEMENTS DOES CONTAINER TECHNOLOGY BRING TO UPDATE AND DEPLOYMENT PROCESSES?

During update and deployment processes, speed and reliability are the key metrics. This is exactly where containers excel:

- **Faster boot-up:** containers start in milliseconds, enabling rapid scaling and deployment
- **Shorter feedback cycles:** developers get test results faster through optimised CI pipelines
- **Easier scaling:** set the desired replica count or autoscaling policy and the platform handles the rest automatically
- **Rolling deployments without interruptions:** new versions become available immediately, without maintenance windows
- **Identical environments:** the container in the CI pipeline is identical to the one in production, eliminating "works on my machine" issues

6. WHAT ARE CONTAINER ORCHESTRATION PLATFORMS?

Container orchestration platforms manage individual containers at scale. They form a dynamic system that groups and places containers, manages their network connections, and ensures communication between them. This type of platform (or cluster) can be run on-premise or on cloud platforms. The most widely adopted orchestration platform is Kubernetes.

7. WHAT IS KUBERNETES?

Kubernetes is an open-source platform for orchestrating containerised workloads and services. It automates the processes of deploying, running, scaling, and managing containerised applications. According to the CNCF Annual Survey 2024, **80% of organisations now use Kubernetes in production:** it is the established standard for running containers at scale.

Kubernetes is currently at version **v1.33** (April 2025) and releases three new minor versions per year, reflecting the rapid pace of development in the ecosystem.

8. WHAT IS DEVOPS?

DevOps as an organisational and leadership culture is based on three principles ("the three ways"): the flow principle, the feedback principle, and the principle of continuous learning and experimentation. These serve as a framework for processes, procedures, and practices focused on streamlining collaboration between development and operations teams, with the overarching goal of delivering software faster and more reliably.

9. ARE DEVOPS AND THE USE OF CONTAINERS ALWAYS INSEPARABLE?

DevOps as a philosophy and containers as a technology make an excellent team, but they are not interdependent. DevOps methods can be applied without containers, and containerisation is possible without a DevOps organisation. That said, container-based workflows do naturally encourage automation, reproducibility, and short feedback cycles, all central to the DevOps way of working.

10. ARE THERE BEST PRACTICES FOR DEPLOYING CONTAINER ENVIRONMENTS?

Helm charts are the de-facto standard for packaging Kubernetes applications. They simplify the deployment of complex multi-component applications and are ubiquitous in the Kubernetes ecosystem. Start by learning Helm: most third-party software (databases, monitoring, ingress controllers) is distributed as Helm charts.

For Kubernetes configuration management, **Kustomize** (built into kubectl) provides a simple, template-free way to manage environment-specific configuration overlays. For teams who prefer writing infrastructure in code, tools like **cdk8s** or **Pulumi** offer more programmatic approaches.

For production deployments, the **GitOps** pattern – using tools like ArgoCD or Flux to continuously reconcile cluster state with declarations in Git – is now widely considered best practice.

11. HOW STEEP IS THE LEARNING CURVE FOR WORKING WITH CONTAINERS?

This depends on the desired degree of autonomy. When running the container platform is fully managed by an external provider, basic container knowledge is sufficient: your team can focus on writing application code. When managing the platform in-house, migrating to a container-based architecture requires comprehensive training for developers and systems engineers: Docker and OCI image concepts, Kubernetes architecture, YAML manifests, networking, security policies, and more. Without a solid foundation, self-managing a Kubernetes cluster in production is risky.

12. WHAT IS THE BEST WAY TO ENSURE A GOOD START TO A CONTAINER PROJECT?

A hands-on training on Docker and Kubernetes provides the ideal foundation. We offer free [Kubernetes webinars](#) (one hour, fundamentals and NKE in practice) and the full-day [Nine Kubernetes Academy](#) (CHF 800 / CHF 600 for NKE customers) for teams who want deeper, practical knowledge. Both have been developed and delivered by our CKA-certified engineers and cover everything from cluster architecture to production-ready deployment patterns.

13. CONTAINER TECHNOLOGY: MANAGED OR UNMANAGED?

Due to the complexity of building and running the underlying infrastructure – and the rapid development pace of the Kubernetes ecosystem (three minor releases per year, each with security updates and breaking changes) – we strongly advise handing the platform management over to a managed service provider. By doing this, your team can focus entirely on application development and running your workloads, without the operational burden of cluster upgrades, certificate management, monitoring, and backup.

14. WHAT IS GITOPS AND WHY DOES IT MATTER?

GitOps is a deployment pattern where the desired state of a Kubernetes cluster is declared in a Git repository, and an automated agent continuously reconciles the live cluster state with what is declared in Git. Tools such as **ArgoCD** and **Flux** implement this pattern. The benefits: fully auditable deployment history, easy rollbacks, automatic drift detection, and a clear separation between infrastructure state and application code. GitOps is now considered a production best practice for Kubernetes deployments.

Security and Network

15. WHICH NETWORK SYSTEMS AND ARCHITECTURES ARE COMPATIBLE WITH CONTAINERS?

Containers can act as a useful supplement or alternative to VMs. They can be linked to VM-based systems within a network when required. Kubernetes networking is handled by CNI (Container Network Interface) plugins: popular choices include Calico, Cilium, and Flannel, each with different performance and security characteristics.

Traditionally, Kubernetes used Ingress resources and controllers (such as ingress-nginx) to manage external traffic routing. The community is now standardising on the **Gateway API**, which reached v1.0 General Availability in 2024 and provides a more expressive, role-oriented approach to traffic management. New deployments should evaluate Gateway API rather than Ingress for long-term compatibility.

16. HOW SAFE ARE CONTAINER SOLUTIONS COMPARED TO VIRTUAL MACHINES?

Containers are just as secure as traditional server solutions and VMs: what matters is how the environment is configured and maintained. Key security considerations in Kubernetes environments include:

- **Pod Security Standards (PSS):** Kubernetes v1.25 removed the deprecated PodSecurityPolicy (PSP). The current mechanism is **Pod Security Admission (PSA)**, which enforces Privileged, Baseline, or Restricted security profiles. Additional policy engines like **Kyverno** or **OPA Gatekeeper** provide more fine-grained control.
- **Image scanning:** Scan container images for known vulnerabilities (CVEs) before they reach production. Tools like Trivy, Grype, or integrated registry scanners automate this.
- **Secret management:** Kubernetes Secrets are only base64-encoded by default. For production, use external secret management via the External Secrets Operator, HashiCorp Vault, or a cloud-provider secret store.
- **Network policies:** Restrict pod-to-pod communication using Kubernetes NetworkPolicy resources to implement a zero-trust network model.

- **RBAC:** Role-Based Access Control limits what users and service accounts can do within the cluster.

17. ARE THERE ALTERNATIVES TO DOCKER FOR BUILDING AND RUNNING CONTAINERS?

The container tooling ecosystem has diversified significantly:

- **For building images: Podman** (daemonless, rootless, drop-in Docker CLI replacement), **Buildah** (fine-grained OCI image building), **Kaniko** (builds inside Kubernetes without Docker daemon)
- **For container runtimes in Kubernetes:** Since Kubernetes v1.24, the Docker runtime layer (dockershim) has been removed. The standard runtime is now **containerd**, which powers over 95% of Kubernetes clusters. CRI-O is another OCI-compliant option, commonly used with OpenShift. Existing Docker images run without modification on containerd, as both use the OCI image format.
- **Important note:** Docker Desktop for commercial use requires a paid subscription since 2022. Teams should evaluate alternatives like Podman Desktop, Rancher Desktop, or OrbStack.

Getting Your Team on Board

18. HOW CAN INTERNAL BUY-IN FOR CONTAINERS BE OBTAINED?

Just as with any other innovation or investment, the most compelling arguments for implementing containers are concrete cost savings and an improved time to market. Practical case studies and real-world examples are highly effective: numbers and stories together make the case. Our customers have documented their experience migrating to NKE; these case studies are available on [nine.ch](#).

19. WHAT ARE THE REALISTIC COSTS AND RESOURCES NEEDED FOR A CONTAINER SOLUTION?

Costs vary depending on the size and complexity of the project, the training effort required, and whether the container platform is managed internally or by an external provider. A thorough upfront analysis – ideally with the input of an experienced external partner – helps assess the expected investment and ongoing operational costs accurately. We provide architectural consulting as part of our customer onboarding to help teams make informed decisions.

20. HOW DO I GET MY TEAM ON BOARD WITH CONTAINERS?

Migrating to a container setup comes with an initial investment and a learning curve. The best way to build acceptance within the team is to clearly demonstrate the advantages: time savings through automated processes, agile working methods, simplified scaling, and improved deployment confidence. A hands-on training – such as the [Nine Kubernetes Academy](#) – is the most effective way to bring a team from theory to practice quickly.

21. WHERE CAN I FIND SUPPORT FOR CONTAINER AND KUBERNETES QUESTIONS?

The official Kubernetes documentation (kubernetes.io) is the most authoritative source for technical questions. The CNCF Landscape (landscape.cncf.io) provides an overview of the entire cloud-native ecosystem. For Swiss-specific requirements, architecture advice, or hands-on support, our certified engineering team is available directly, not via an anonymous ticket system.

Talk to one of our engineers about container technology

We are happy to help, whether you are just starting out or looking to migrate an existing workload.

[Get in touch](#)

Nine Internet Solutions AG · Badenerstrasse 47 · 8004 Zurich · Switzerland
info@nine.ch · +41 44 637 40 00 · nine.ch