Container security, in plain terms.

Containers and Kubernetes power modern infrastructure, and a single weak setting can turn one compromised pod into a cluster takeover. This section breaks the runtime risks (privileged containers, the Docker socket, host namespaces, host mounts, capabilities) and the Kubernetes surface (kubelet, RBAC, service account tokens, etcd, privileged pods) into plain-language explainers, each ending with how a penetration test surfaces the path in your environment.

HOW IT WORKS

01 Key terms explained

Plain-language definitions of the misconfigurations and components behind container and Kubernetes attacks. Each page covers what it is, the attack, the payload, and how to defend.

Docker and runtime

- What is a privileged container?
- What is the Docker socket?
- What is host namespace sharing?
- What is a host-path mount?
- What is Docker image security?
- Container vs virtual machine
- What is CAP_SYS_ADMIN?

Kubernetes

- What is an exposed kubelet?
- What is Kubernetes RBAC?
- What is a service account token?
- What is etcd?
- What is a privileged pod?

02 How to read this section

The pages follow how an attacker moves through a containerized environment.

- Foundations first: container security, the container escape, and Kubernetes security.

SOURCES

[1] MITRE ATT&CK: Containers Matrix

[2] NIST SP 800-190 Application Container Security Guide

- Docker and runtime: the run-time misconfigurations (privileged mode, the Docker socket, host namespaces and mounts, capabilities) that enable an escape, plus image hygiene and how a container differs from a VM.
- Kubernetes: the cluster attack surface, the kubelet, RBAC, service account tokens, etcd, and privileged pods, that turns one pod into cluster control.

Each explainer ends with how a penetration test confirms the path in your own clusters.

Find the container escape paths before an attacker does.

securelayer7.net/learn/containers

Open online