Kubernetes

Security in cloud-native environments

LINUS ÖSTBERG

linus@scilifelab.uu.se

SciLifeLab Data Centre

Who am I?

Linus Östberg

SciLifeLab Data Centre

(soon Conoa AB)

talavis

linus@oestberg.dev

Where to start?

- Risk assessment
- Threat modelling
- Laws, regulations

CIA

- Confidentiality
- Integrity
- Availability

Fundamentals

- Defence in depth
- Least privilege
- Zero trust

Cloud-native Security

- Code
- Containers
- Clusters
- Cloud (on-premise)

Code

- Security starts in the application
- Secure software development
- Usable by default, not secure

Supply Chain Security

- Trusting third-party libraries
  - Code evaluation?
  - Integrity check?
- Up-to-date?

Third-party security vulnerabilities

- Vulnerability scanning
- Snyk
- Trivy
- Dependabot

TASK: SCANNING WITH TRIVY

- `https://github.com/ScilifelabDataCentre/lunch-menu`
- Are there any known vulnerabilities?
- Use `trivy`, `snyk`, or any other scanner
- Hints:
  - `requirements.txt`
  - `yarn.lock`
  - `trivy fs`

What is a container?

Open Container Initiative

- OCI
  - Image
  - Runtime
  - Distribution

Container Runtimes

- Docker
- Containerd
- CRI-O
- Gvisor
- Kata
- Firecracker

What are the differences between a container and a virtual machine?

- Containers share the kernel
- Very sensitive data should be in different vms/clusters

ROOT == ROOT != ROOT

- Container user == host user
  - User namespaces exist, but have limited support
- Capabilities
- hostUsers
- Privileged container == danger
- Do not run your containers as root

ROOT == ROOT != ROOT

- Container user == host user
  - User namespaces exist, but have limited support
- Capabilities
- hostUsers
- Privileged container == danger
- Do not run your containers as root

Hardened container runtimes

- Run containers using hardened runtimes
- Kata containers (virtualisation)
- gVisor (sandbox)

TASK: USE GVISOR

- RuntimeClass
- https://killercoda.com/killer-shell-cks/scenario/sandbox-gvisor

Container Drift

- Containers are often intended to do only one thing
- Containers that do other things are ”drifting”
- Preventing drift may improve security

SECCOMP, APPARMOR, SELINUX

- Security frameworks to add extra security
- Ubuntu: seccomp, apparmor
- Red Hat: seccomp, selinux
- Define a security profile to be used with a container
- Optimal security: define a specialised profile for each container
- Using the ”runtime default” is better than nothing

TASK: USING APPARMOR

- `container.apparmor.security.beta.kubernetes.io/<container_name>: <profile_ref>`
- `runtime/default`
- `localhost/k8s-apparmor-deny-everything`
- Create profile, make sure it exists on all nodes where the container may run
- https://killercoda.com/killer-shell-cks/scenario/apparmor

Supply Chain Security

- Security vulnerabilities in third-party libraries
- Insecure/malicious images
- Container signing
- Image minimisation

Security Vulnerabilities

- Scan the container images
- Discover issues in the application and helper files
- E.g. Trivy

TASK: IMAGE AND CODE SCANNING

- `https://github.com/ScilifelabDataCentre/lunch-menu`
- Do the latest container images contain any known vulnerabilities?
- Does the containers with tag 23.7.2 contain any known vulnerabilities?
- Use Trivy or any other scanner
- `trivy image container:label`
- Does this make sense?

Malicious Compliance

- https://www.youtube.com/watch?v=9weGi0csBZM
- Possible to trick the vulnerability scanners
  - Remove package management files
  - Symlinks
  - Multi-stage builds

Building containers

- Minimise
  - Minimal base
  - Remove unused binaries
  - Squash layers
- Never include secrets during the build steps
- Automate
- Do not run as root
- Immutable

Running With Scissors

https://www.youtube.com/watch?v=ltrV-Qmh3oY

KUBERNETES

- Container orchestration
- Designed to be flexible
- Declare wanted state
  - Reconciliation loop

OWASP Kubernetes Top Ten

https://owasp.org/www-project-kubernetes-top-ten/

K01 Insecure Workload Configurations
K02 Supply Chain Vulnerabilities
K03 Overly Permissive RBAC Configurations
K04 Lack of Centralized Policy Enforcement
K05 Inadequate Logging and Monitoring
K06 Broken Authentication Mechanisms
K07 Missing Network Segmentation Controls
K08 Secrets Management Failures
K09 Misconfigured Cluster Components
K10 Outdated and Vulnerable Kubernetes Components

K10: Outdated and Vulnerable Kubernetes Components

- Keep Kubernetes updated
- Currently supported releases:
  - 1.28
  - 1.27
  - 1.26
  - 1.25
- $\sim$1 year support
- Distributions may be supported longer

K09: Misconfigured Cluster Components

- CIS Benchmarks (kube-bench)

TASK: USING KUBE-BENCH

- Using kube-bench
- https://killercoda.com/killer-shell-cks/scenario/cis-benchmarks-kube-bench-fix-controlplane

K08: Secrets Management Failures

- Encrypt secrets
- Vault
- Sealed secrets etc

K07: Missing Network Segmentation Controls

- Zero trust
- Network policies

Network Policies

- https://editor.networkpolicy.io/
- Default: deny-all for namespace
- Minimise access

TASK: CREATING NETWORK POLICIES

- Practice creating network policies
- https://killercoda.com/killer-shell-cks/scenario/networkpolicy-namespace-communication
- https://editor.networkpolicy.io/

K06: Broken Authentication Mechanisms

- Certificates last until expiration
- Service account tokens
- Use MFA if possible
- Be careful with service account tokens

K05: Inadequate Logging and Monitoring

- Save logs in an external system
  - Kubernetes Audit logs
  - Application/container logs
  - Event logs
  - Operating system logs
  - Network logs
- **Monitor the logs**

K04: Lack of Centralized Policy Enforcement

- Policies about what may run on the cluster
- Policies as Code — PaC
- Pod Security Standards
- OPA Gatekeeper
- Kyverno

Pod Security Standards

- https://kubernetes.io/docs/concepts/security/pod-security-standards/
- Apply to a namespace
- `pod-security.kubernetes.io/<MODE>: <LEVEL>`
- `pod-security.kubernetes.io/<MODE>-version: <VERSION>`
- Enforce, audit, warn

OPA Gatekeeper

- Open Policy Agent
- Rego
- OPA Gatekeeper
- Contstraint templates
- Constraints
- https://killercoda.com/opa/scenario/intro

Kyverno

- YAML
- Kubernetes-native https://killercoda.com/kyverno/scenario/intro

K03: Overly Permissive RBAC Configurations

- Least Privilege
- Service account and user RBAC permissions
- Limit use of ClusterRoleBinding
- Not everyone needs admin permissions
- Be careful with service account tokens

K02: Supply Chain Vulnerabilities

- Security vulnerabilities in third-party libraries
- Insecure/malicious images
  - Do not use untrusted images
- Secure the CI/CD pipelines
- Software bill of materials (SBOM)
- Container signing

K01: Insecure Workload Configurations

- A container in Kubernetes is by default less secure than in Docker
- Need to improve the configuration
- Tools:
  - kube-score
  - kubesec
  - snyk

No root

- securityContext
  - runAsUser
  - runAsGroup
  - allowPrivilegeEscalation
  - privileged
  - runAsNonRoot
  - capabilities

SERVICE ACCOUNT TOKENS

- Never mount service account tokens unless they are needed
- `automountServiceAccountToken: false`

AppArmor, Seccomp, SELinux

- Use Seccomp, SELinux, and Apparmor
  - Must be supported by the hosts

SECCOMP AND SELINUX

```
securityContext:
 seccompProfile:
   type: RuntimeDefault
 seLinuxOptions:
   level: "s0:c123,c456"
```

AppArmor

```
spec:
  template:
    metadata:
      annotations:
        container.apparmor.security.beta.kubernetes
```

Immutability

- A container should be immutable
- `readOnlyRootFilesystem: true`
- Binaries can be run without being saved to disk
- Add emptyDirs if specific folders need writing

Enforcement

- Pre-deployment:
  - kube-score
  - kubesec
  - kubeaudit
  - snyk
- Admission:
  - Pod security standards
  - Kyverno
  - OPA Gatekeeper

TASK: CREATE A HARDENED DEPLOYMENT

- Make a deployment of `ghcr.io/scilifelabdatacentre/menu-backend:latest`
- Create a new namespace and apply the pod security standard restricted to it
- Update your deployment to allow deployment in the created namespace
- Optional: add network policies or any other relevant protections

Host security

- Hardening
- CIS Benchmark
- Minimisation
- Firewalls

GitOps

- Using a Git repository as the source of truth
- Any changes are committed to Git
- Current cluster state always saved to Git
- Argo, Flux
- Analysis tools can be run as part of CI
- Adds non-repudiation to any changes

Compliance

- Not just a checklist
- Aid to make your systems more secure

- Least Privilege
- Defence in depth (layered security)
- Zero Trust

Keep on learning!

Thank you for listening!

Questions?