Kubernetes

Security and stuff

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

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

Container drift

- Preventing containers from doing unintended actions

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

- https://killercoda.com/killer-shell-cks/scenario/apparmor

Security Vulnerabilities

- Scan the packaged software in container images
- E.g. Trivy

TASK: IMAGE AND CODE SCANNING

- `https://github.com/ScilifelabDataCentre/lunch-menu`
- Do the latest container images contain any known vulnerabilities?
- Use Trivy or any other scanner
- `trivy image container:label`

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

KUBERNETES

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

Running With Scissors

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

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
- Three most recent minor releases:
  - 1.28
  - 1.27
  - 1.26
- $\sim$1 year support
- Distributions may be supported longer

K09: Misconfigured Cluster Components

- CIS Benchmarks (kube-bench)

TASK: 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

TASK: ENCRYPTING ETCD

- https://killercoda.com/killer-shell-cks/scenario/secret-etcd-encryption

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

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

K06: Broken Authentication Mechanisms

- Certificates last until expiration
- No certificate revocation list
- Service account tokens
- use MFA if possible

K05: Inadequate Logging and Monitoring

- Audit logs, logs in external system
- Falco etc for monitoring
- Fluentbit etc
- Loki

K04: Lack of Centralized Policy Enforcement

- OPA Gatekeeper, Kyverno
- “Policies in git”

TASK: USING KYVERNO

https://killercoda.com/kyverno/scenario/intro

- GitOps — Argo
- Infrastructure as code
- Ansible
- “Replace one node every month”

K03: Overly Permissive RBAC Configurations

- Least Privilege
- Service account and user RBAC permissions
- Limit use of ClusterRoleBinding
- Not everyone needs admin permissions

K02: Supply Chain Vulnerabilities

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

Supply Chain Vulnerabilities

- Signing
- Do not use untrusted images

Signing

K01: Insecure Workload Configurations

- Scanners: kube-score, kubesec, kubeaudit, snyk ...
- Pod Security Standards
- Pod admission: kyverno, Open Policy Agent Gatekeeper

No root

SERVICE ACCOUNT TOKENS

TASK: ATTACK TOKEN

-

AppArmor/seccomp

-

Immutability

-

Enforcement

GitOps

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

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

Compliance

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

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