

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

- ▶ **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://killercode.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/](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
- ▶ ~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
- ▶ Constraint 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
 - kubesecc
 - 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
 - kubesecc
 - 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?