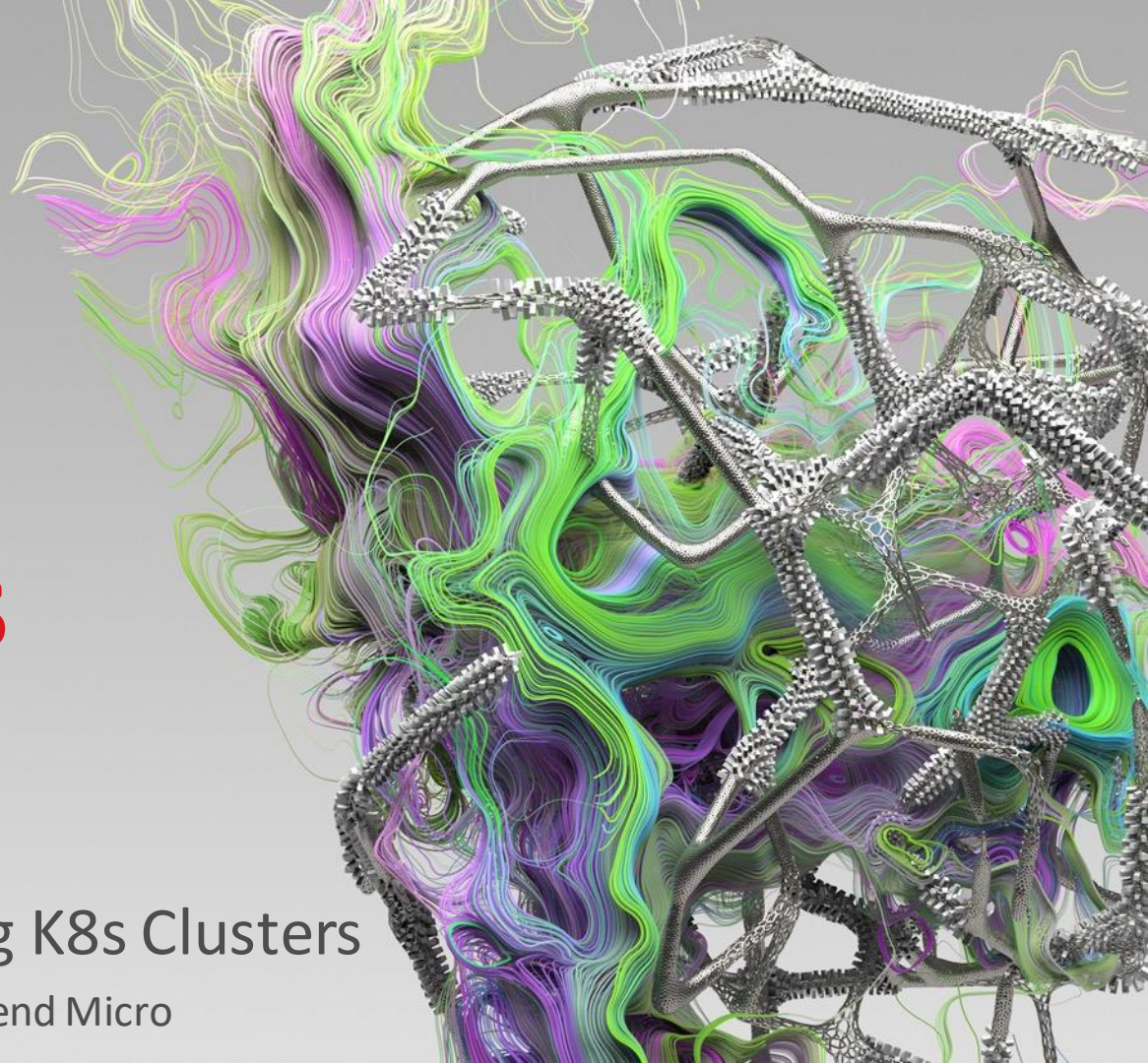


Kubernetes Security

Attacking and Defending K8s Clusters

Magno Logan – Project Nebula @ Trend Micro

@magnologan

- Information Security Specialist & Senior Threat Researcher @ Trend Micro
- Cloud and Container Security Research Team
- Member of the CNCF SIG-Security Team
- Not a K8s Security Expert (yet) =)
- Personal blog: katanasec.com

Agenda

- What is Kubernetes?
- K8s Architecture
- MITRE ATT&CK
  - K8s Threat Matrix
  - MITRE ATT&CK for Containers
  - K8s ATT&CK Scenario & Flow
- Attacking K8s
  - Recon / Initial Access
  - Exploitation / Execution
  - Post-Exploitation / Persistence
- Defending K8s
  - API Server
  - CIS Benchmark
  - Image Scanning
  - Runtime Protection
  - Network Policy
  - ~~Pod Security Policy (PSP)~~
  - PSP Alternatives
  - Audit Logs

Awesome K8s Security List

Awesome Kubernetes (K8s) Security awesome

A curated list for Kubernetes (K8s) Security resources such as articles, books, tools, talks and videos.

Disclaimer

Most of the resources are in English, the ones that aren't will be flagged as such. Most of the contents of this list are public free, please use them for educational purposes only!

Not all the tools have been tested or reviewed, use them at your own risk! Also, I don't consider myself a K8s Security expert, I'm just learning and helping others learn along with me. Thanks!

Contents

These are the main contents of this awesome list. Everything related to the security of Kubernetes, either breaking or improving it, will be added down below. If you have any other good recommendations, feel free to submit a PR!

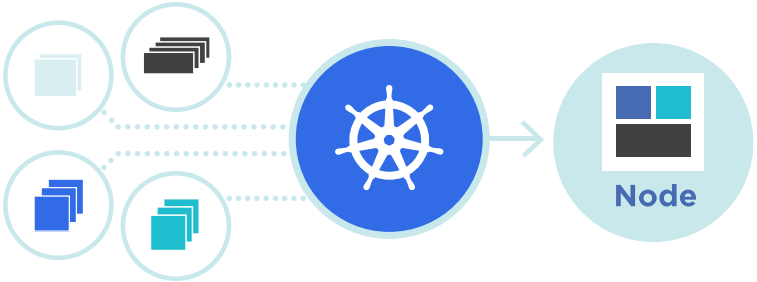
- The Basics
- Official Pages
- Talks and Videos
- Blogs and Articles
- Books
- Certifications
- CVEs
- Slides
- Trainings
- Repositories
- Papers
- Podcasts
- Jobs
- Community

https://github.com/magnologan/awesome-k8s-security

TREND MICRO

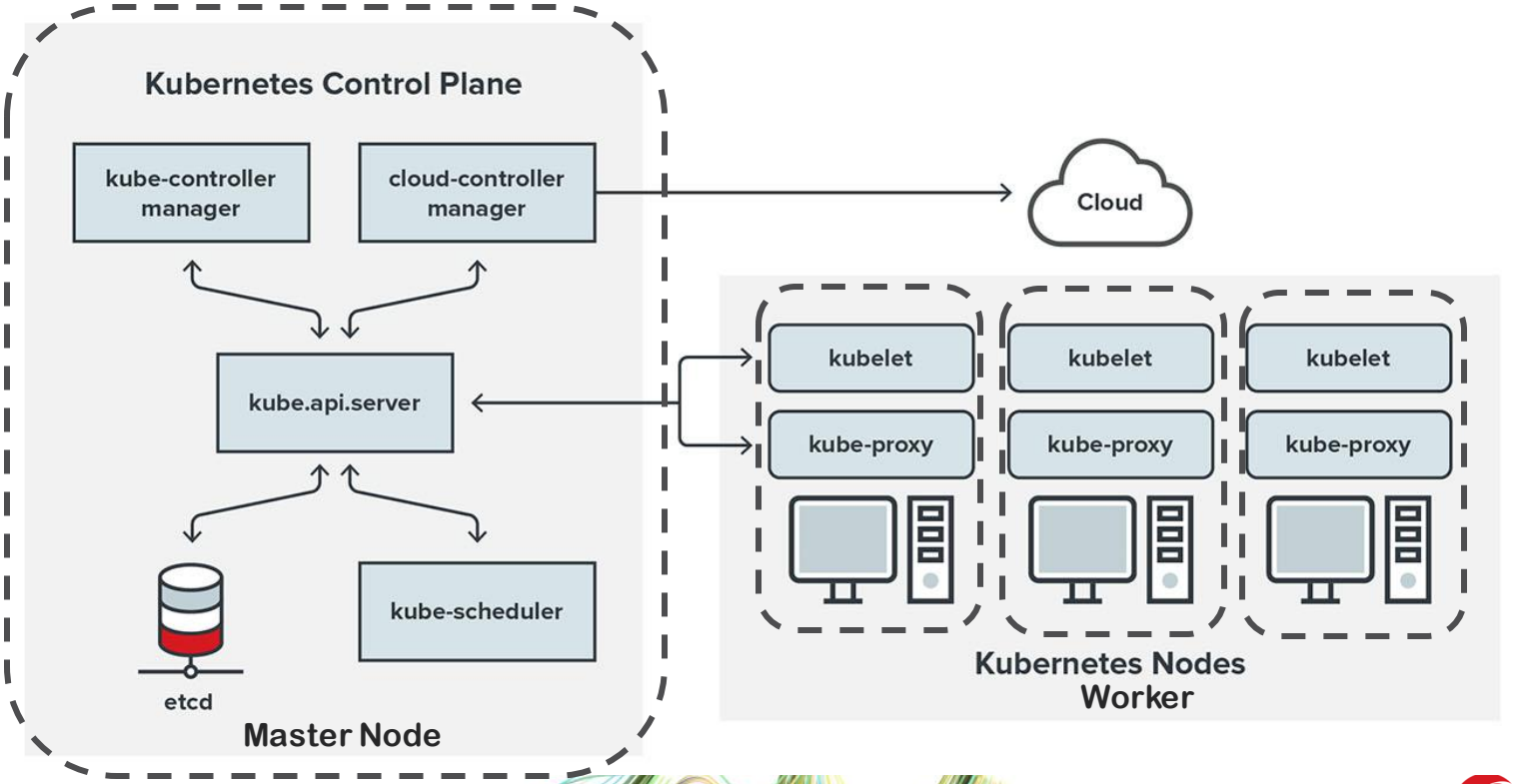
What is Kubernetes?

- Kubernetes (or K8s) is an open-source system for automating deployment, scaling, and management of containerized applications



kubernetes.io

K8s Architecture



MITRE ATT&CK Framework

ATT&CK®

- Adversarial Tactics, Techniques & Common Knowledge - ATT&CK
- Globally accessible KB of opposing tactics and techniques based on real-world scenarios
- Used as a basis for the development of specific threat models and methodologies in many different sectors.

https://attack.mitre.org/

TREND MICRO™

K8s Threat Matrix by Microsoft

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Impact
Using Cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access the K8S API server	Access cloud resources	Data Destruction
Compromised images in registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Resource Hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Access container service account	Network mapping	Cluster internal networking	Denial of service
Application vulnerability	Application exploit (RCE)		Access cloud resources	Connect from Proxy server	Applications credentials in configuration files	Access Kubernetes dashboard	Applications credentials in configuration files	
Exposed Dashboard	SSH server running inside container					Instance Metadata API	Writable volume mounts on the host	
							Access Kubernetes dashboard	
							Access tiller endpoint	

https://www.microsoft.com/security/blog/2020/04/02/attack-matrix-kubernetes

MITRE ATT&CK for Containers (and K8s)

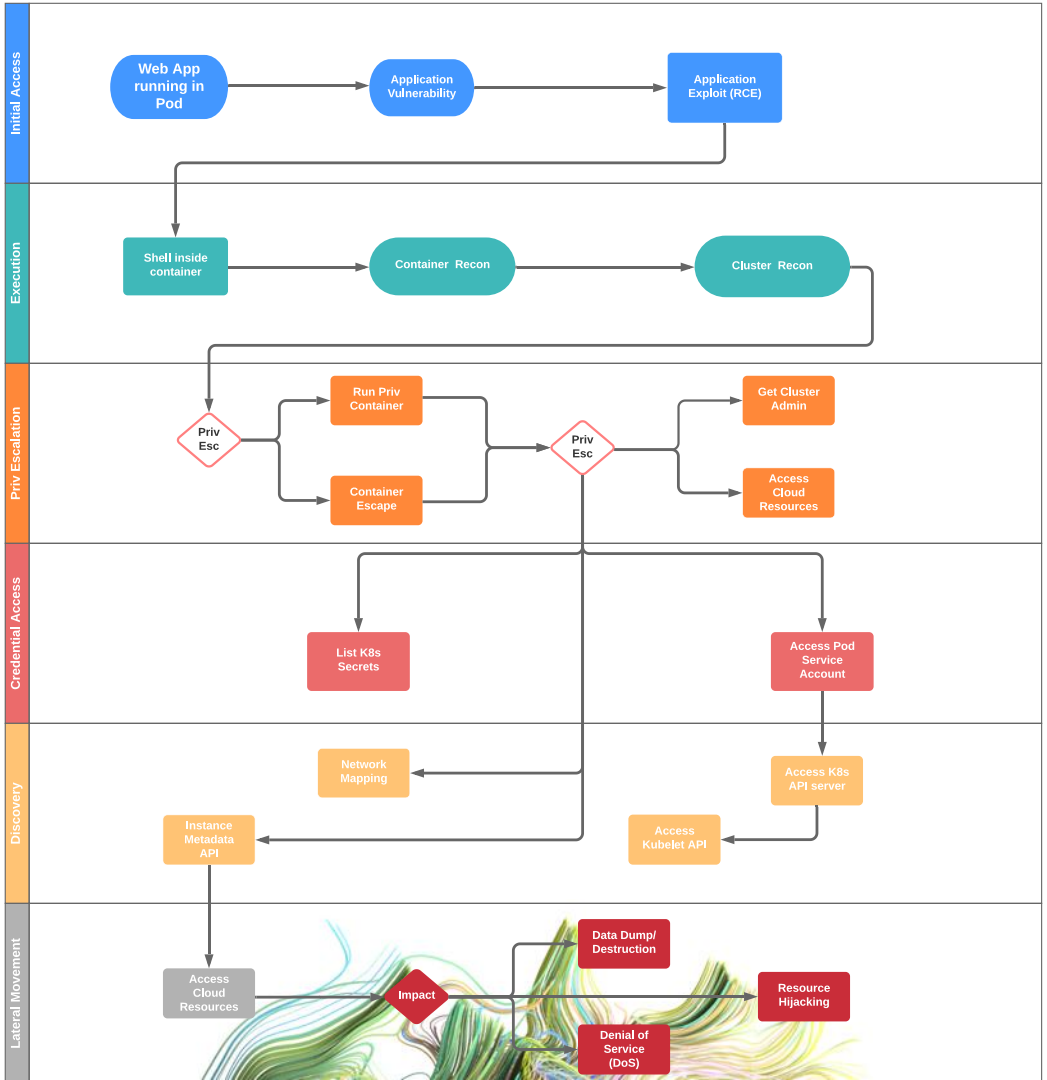
Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Impact
Exploit Public-Facing Application	Container Service	Implant Internal Image (NAME CHANGE)	Escape to Host	Build Image on Host	Brute Force	Container Resource Discovery	Endpoint Denial of Service
External Remote Services	Deploy Container	Scheduled Task/Job	Scheduled Task/Job	Deploy Container	Brute Force: Password Guessing		Network Denial of Service
Valid Accounts	Scheduled Task/Job	Scheduled Task/Job: Container Orchestration Job	Scheduled Task/Job: Container Orchestration Job	Masquerading	Brute Force: Password Spraying		Resource Hijacking
Valid Accounts: Local Accounts	Scheduled Task/Job: Container Orchestration Job	Valid Accounts	Valid Accounts	Masquerading: Match Legitimate Name or Location	Brute Force: Credential Stuffing		
	User Execution	Valid Accounts: Local Accounts	Valid Accounts: Local Accounts	Valid Accounts	Unsecured Credentials		
	User Execution: Malicious Image			Valid Accounts: Local Accounts	Unsecured Credentials: Credentials in Files		
					Unsecured Credentials: Container API		

Proposed new techniques and sub-techniques

https://medium.com/mitre-engenuity/update-help-shape-att-ck-for-containers-bfcd24515df5

TREND MICRO

K8s ATT&CK Scenario



Attacking K8s

Initial Access

- Web Application Vulnerability / Exploit (RCE)
  - In this scenario we are exploiting CVE-2018-7600
    - https://github.com/dreadlocked/Drupalgeddon2
- Exposed Dashboard or Kube API Server
  - The Kube API server endpoint is public by default in some managed services (EKS)!

Exploitation / Execution

- Reach the API endpoint externally

```
curl -k https://3.96.191.147:6443
```

```
$ curl -k https://[redacted].yl4.ca-central-1.eks.amazonaws.com
```

- Get a shell inside one of pods from the cluster
  - Exposed dashboard
  - App vuln RCE

kube-hunter

- Hunts for security weaknesses in Kubernetes clusters
- Developed to increase awareness and visibility for security issues in Kubernetes environments
- Currently detects 37 vulnerabilities on K8s

https://aquasecurity.github.io/kube-hunter/

Internal Recon – Inspect the K8s env

- Environment variables: env | grep -i kube
- Service Account token:
  /var/run/secrets/kubernetes.io/serviceaccount
- amicontained
  - Container introspection tool.
  - Find out what container runtime is being used as well as features available.

Post-Exploitation / Persistence

- And privilege escalation!
- Pod/Container Escape via privileged pod:

https://twitter.com/mauilion/status/1129468485480751104

```
"spec": {
    "hostPID": true,
    "containers": [
        {
            "name": "1",
            "image": "alpine",
            "command": [
                "nsenter",
                "--mount=/proc/1/ns/mnt",
                "--",
                "/bin/bash"
            ],
            "stdin": true,
            "tty": true,
            "securityContext": {
                "privileged": true
            }
```

Defending K8s

- How can I protect my cluster from attackers?
- Isn't K8s secure by default?
- Where do I start?

TREND MICRO

The Kube API Server

```
curl -k https://3.96.191.147:6443
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {

  },
  "status": "Failure",
  "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
  "reason": "Forbidden",
  "details": {

  },
  "code": 403
}%
```

CIS Kubernetes Benchmark

- Prescriptive guidance for establishing a secure configuration posture for Kubernetes
- +120 security checks for your K8s cluster
- Created by Rory Mccune and Liz Rice and many other contributors
- There are specific ones for EKS and GKE

kube-bench

- Checks whether Kubernetes is deployed securely
- Validate it against the CIS K8s Benchmark
- Developed in Go

TREND MICRO

kube-bench Results

```
[INFO] 1 Master Node Security Configuration
[INFO] 1.1 Master Node Configuration Files
[FAIL] 1.1.1 Ensure that the API server pod specification file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.2 Ensure that the API server pod specification file ownership is set to root:root (Automated)
[FAIL] 1.1.3 Ensure that the controller manager pod specification file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.4 Ensure that the controller manager pod specification file ownership is set to root:root (Automated)
[FAIL] 1.1.5 Ensure that the scheduler pod specification file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.6 Ensure that the scheduler pod specification file ownership is set to root:root (Automated)
[FAIL] 1.1.7 Ensure that the etcd pod specification file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.8 Ensure that the etcd pod specification file ownership is set to root:root (Automated)
[WARN] 1.1.9 Ensure that the Container Network Interface file permissions are set to 644 or more restrictive (Manual)
[WARN] 1.1.10 Ensure that the Container Network Interface file ownership is set to root:root (Manual)
[FAIL] 1.1.11 Ensure that the etcd data directory permissions are set to 700 or more restrictive (Automated)
[FAIL] 1.1.12 Ensure that the etcd data directory ownership is set to etcd:etcd (Automated)
[FAIL] 1.1.13 Ensure that the admin.conf file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.14 Ensure that the admin.conf file ownership is set to root:root (Automated)
[FAIL] 1.1.15 Ensure that the scheduler.conf file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.16 Ensure that the scheduler.conf file ownership is set to root:root (Automated)
[FAIL] 1.1.17 Ensure that the controller-manager.conf file permissions are set to 644 or more restrictive (Automated)
[FAIL] 1.1.18 Ensure that the controller-manager.conf file ownership is set to root:root (Automated)
[FAIL] 1.1.19 Ensure that the Kubernetes PKI directory and file ownership is set to root:root (Automated)
[WARN] 1.1.20 Ensure that the Kubernetes PKI certificate file permissions are set to 644 or more restrictive (Manual)
[WARN] 1.1.21 Ensure that the Kubernetes PKI key file permissions are set to 600 (Manual)
[INFO] 1.2 API Server
[WARN] 1.2.1 Ensure that the --anonymous-auth argument is set to false (Manual)
[FAIL] 1.2.2 Ensure that the --basic-auth-file argument is not set (Automated)
[FAIL] 1.2.3 Ensure that the --token-auth-file parameter is not set (Automated)
[FAIL] 1.2.4 Ensure that the --kubelet-https argument is set to true (Automated)
[FAIL] 1.2.5 Ensure that the --kubelet-client-certificate and --kubelet-client-key arguments are set as appropriate (Automated)
[FAIL] 1.2.6 Ensure that the --kubelet-certificate-authority argument is set as appropriate (Automated)
[FAIL] 1.2.7 Ensure that the --authorization-mode argument is not set to AlwaysAllow (Automated)
[FAIL] 1.2.8 Ensure that the --authorization-mode argument includes Node (Automated)
[FAIL] 1.2.9 Ensure that the --authorization-mode argument includes RBAC (Automated)
[WARN] 1.2.10 Ensure that the admission control plugin EventRateLimit is set (Manual)
```

TREND MICRO

Image Scanning

- Clair
- docker scan
- SmartCheck
- Snyk
- Trivy

Cloud-Native Runtime Protection

- Falco
  - Parses Linux kernel sys calls at runtime
  - Detects unexpected behavior on your cluster
  - Generates alerts based on threats detected
  - Uses an easy and powerful rules engine

TREND MICRO

The Pods

- Limit Resources
  - CPU & Memory

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: pods-low
spec:
  hard:
    cpu: "5"
    memory: 10Gi
    pods: "10"
```

- Create and apply a Security Context
  - AllowPrivilegeEscalation = false
  - ReadOnlyRootFileSystem = true
  - RunAsNonRoot = true

The Pods

- Use Seccomp, AppArmor and SELinux
  - Seccomp – filters a process's system calls
  - AppArmor – uses program profiles to restrict the capabilities of individual programs
  - SELinux - applies security labels to objects and evaluates all security-relevant interactions via the security policy.

~~Pod Security Policy (PSP)~~

Control Aspect	Field Names
Running of privileged containers	`privileged`
Usage of host namespaces	`hostPID`, `hostIPC`
Usage of host networking and ports	`hostNetwork`, `hostPorts`
Usage of volume types	`volumes`
Usage of the host filesystem	`allowedHostPaths`
Allow specific FlexVolume drivers	`allowedFlexVolumes`
Allocating an FSGroup that owns the pod's volumes	`fsGroup`
Requiring the use of a read only root file system	`readOnlyRootFilesystem`
The user and group IDs of the container	`runAsUser`, `runAsGroup`, `supplementalGroups`
Restricting escalation to root privileges	`allowPrivilegeEscalation`, `defaultAllowPrivilegeEscalation`
Linux capabilities	`defaultAddCapabilities`, `requiredDropCapabilities`, `allowedCapabilities`
The SELinux context of the container	`seLinux`
The Allowed Proc Mount types for the container	`allowedProcMountTypes`
The AppArmor profile used by containers	annotations
The seccomp profile used by containers	annotations
The sysctl profile used by containers	`forbiddenSysctls`, `allowedUnsafeSysctls`

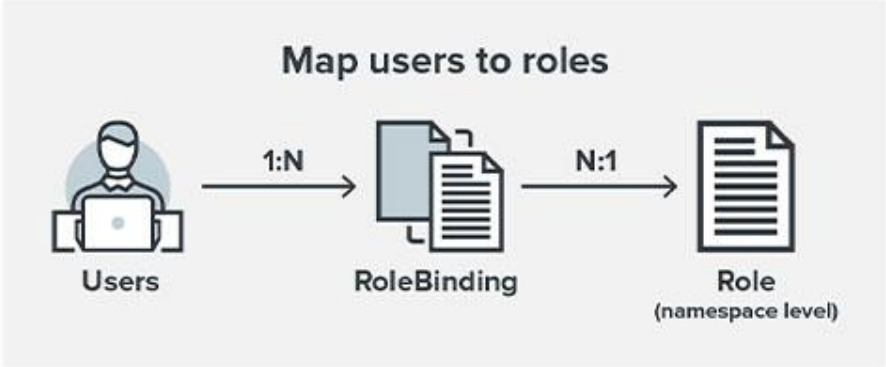
PSP Replacement Alternatives

- OPA / Gatekeeper
- Kyverno
- Still in discussion by K8s SIG-Auth:
  - PSP++ / ContainerBoundary Policy
  - Bare Minimum PSP

RBAC (Role Based Access Control)

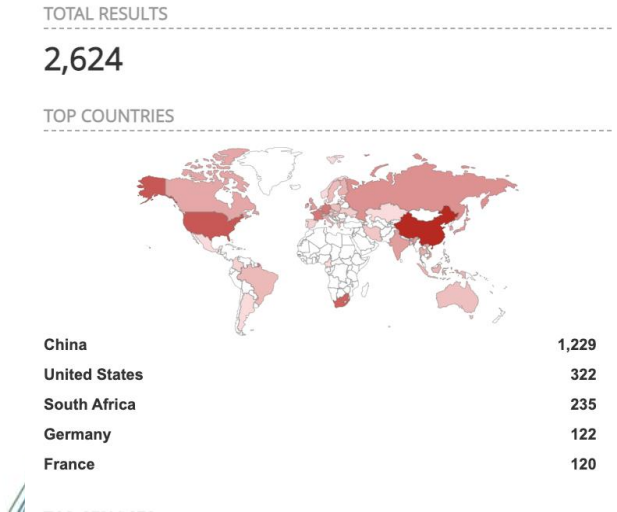
- Allows to configure who can access what in the cluster
- Enabled by default on the latest K8s: *--authorization-mode=Node,RBAC*
- Has 4 different objects: Role, RoleBinding, ClusterRole and ClusterRoleBinding

RBAC



The etcd

- Main data storage location for your cluster
- All the cluster objects are saved here!
- Therea are +2,600 exposed etcd on Shodan



The etcd

- Encryption in transit – default

  ps –ef | grep etcd

  --cert-file = /etc/kubernetes/pkietcd/server.crt

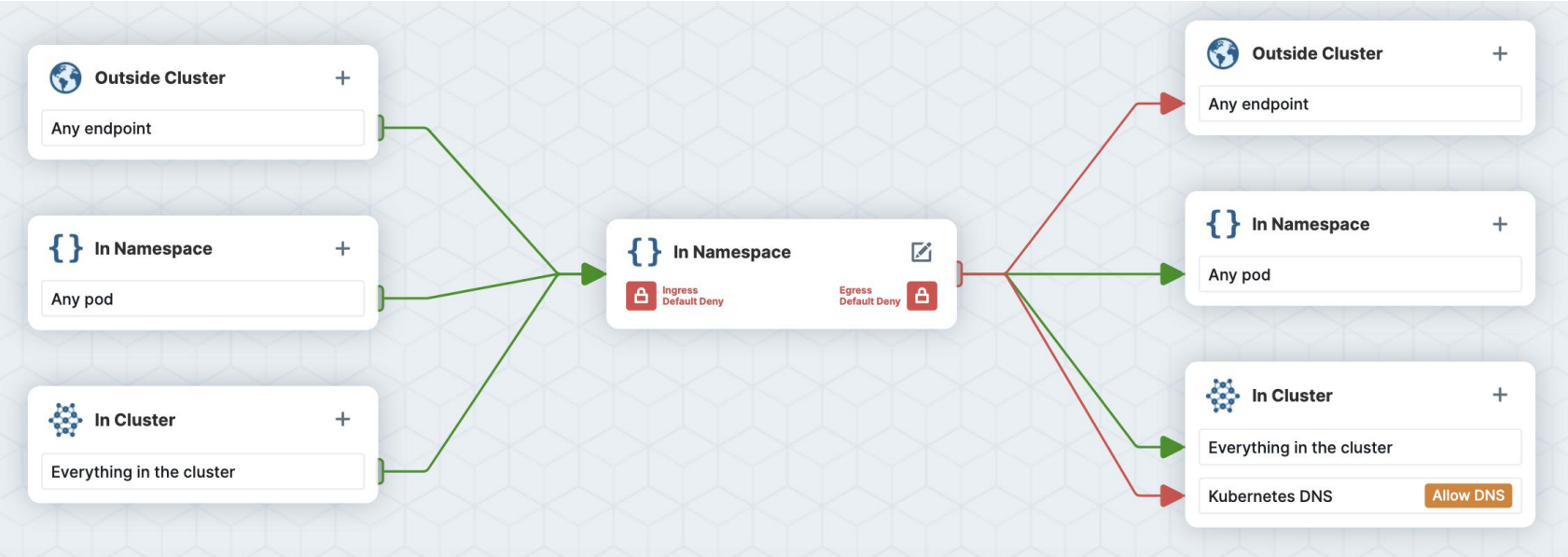
  --key-file = /etc/kubernetes/pki/etcd/server.key

- Encryption at rest – not default!
  - By default, it's all stored in plain text
  - Create an EncryptionConfiguration object

The Network Policy

- By default, all pods can communicate with any other pod in the cluster
- Make sure you create a proper network policy for your cluster
- Does the front-end pod really need to talk to the DB pod?
- What if an attacker can access the pods on the kube-system namespace?

Network Policy Editor by Cilium



https://editor.cilium.io/

The Audit Logs

- Audit logs are not enabled by default
- Highly recommended to enable them for security and troubleshooting
- Need at least two things: a log path and a policy file
- Set those up on the kube-apiserver configuration

The Basics

- Update your Kubernetes environment version early and often, latest version is v1.20
- Don’t use the cluster admin user for your daily work, treat it like root!
- If you can, use a managed K8s service (AKS, EKS, GKE)
- Check out the CIS Kubernetes Benchmark document for more security best practices

References

- https://securekubernetes.com
- https://github.com/magnologan/awesome-k8s-security
- https://www.oreilly.com/library/view/hacking-kubernetes
- https://info.aquasec.com/kubernetes-security
- https://www.microsoft.com/security/blog/2020/04/02/attack-matrix-kubernetes

- https://www.trendmicro.com/vinfo/us/security/news/security-technology/the-basics-of-keeping-your-kubernetes-cluster-secure-part-1

TREND MICRO™

THE ART OF CYBERSECURITY

Unknown threats detected and stopped over time by Trend Micro. **Created with real data by artist Brendan Dawes.**