

Hardening K8s_{1.15+}

19-02-2019
V2.0

Trusted partner for your Digital Journey

© Atos - For internal use

Atos

sommaire

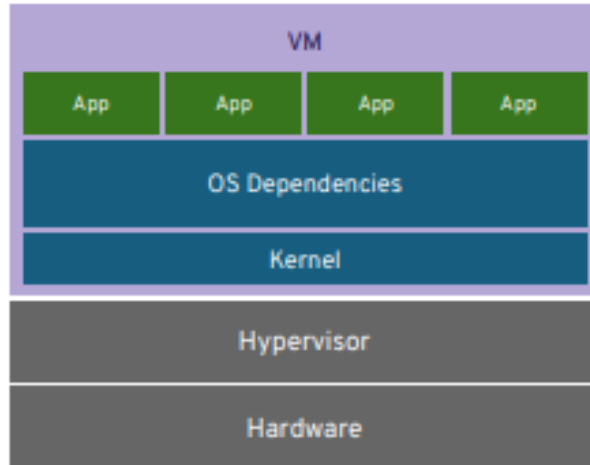
- ▶ Introduction
- ▶ Sécuriser la machine Host
- ▶ Sécuriser le Traffic
- ▶ Sécuriser le Control-Plane
- ▶ Sécuriser les composants internes
- ▶ Monitoring & Logging
- ▶ Perspectives
- ▶ Appendix

1

Introduction

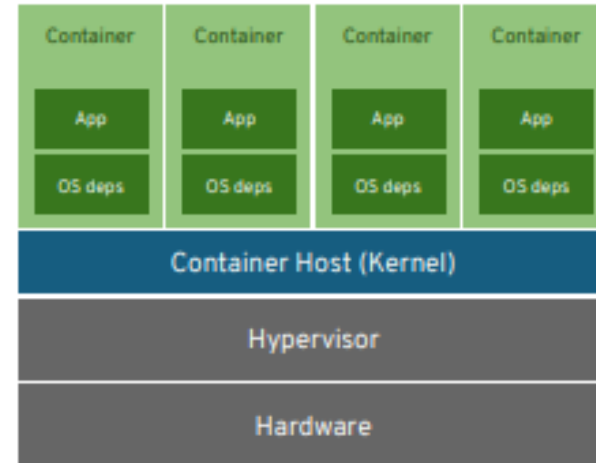
VMs & Containers

VIRTUAL MACHINES



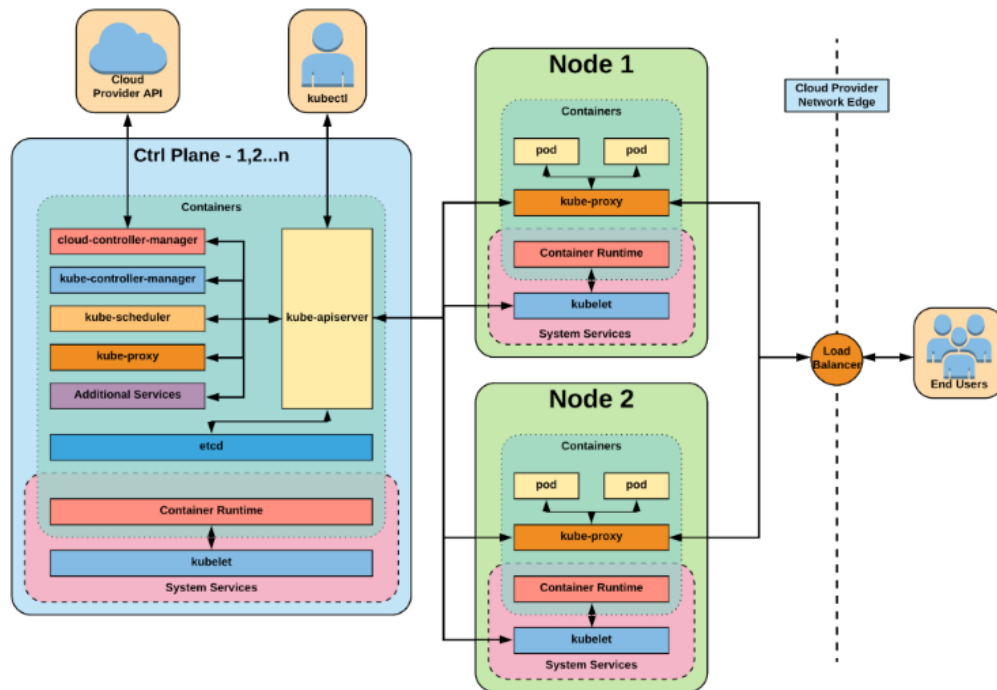
VM isolates the hardware

CONTAINERS

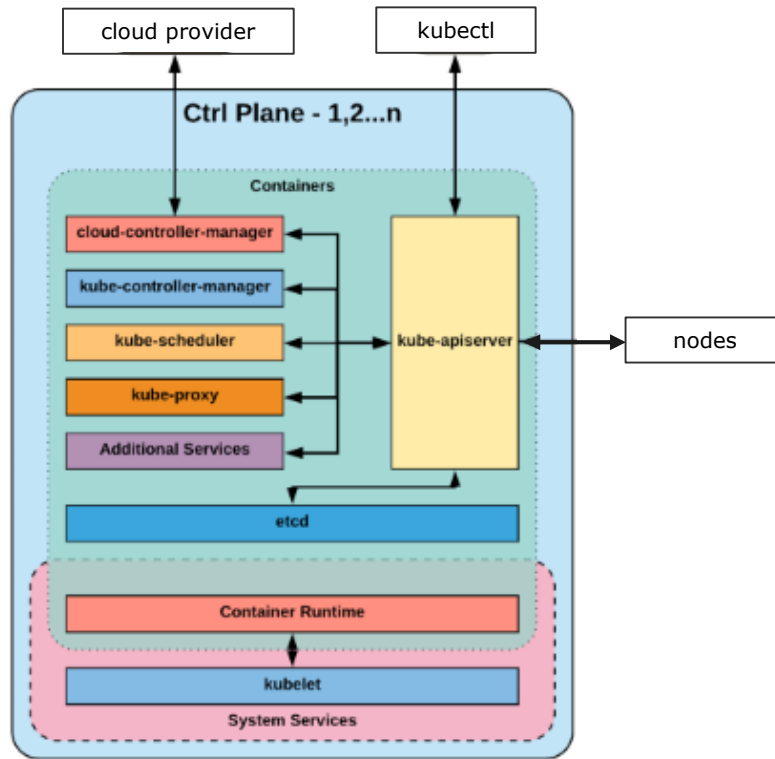


Container isolates the process

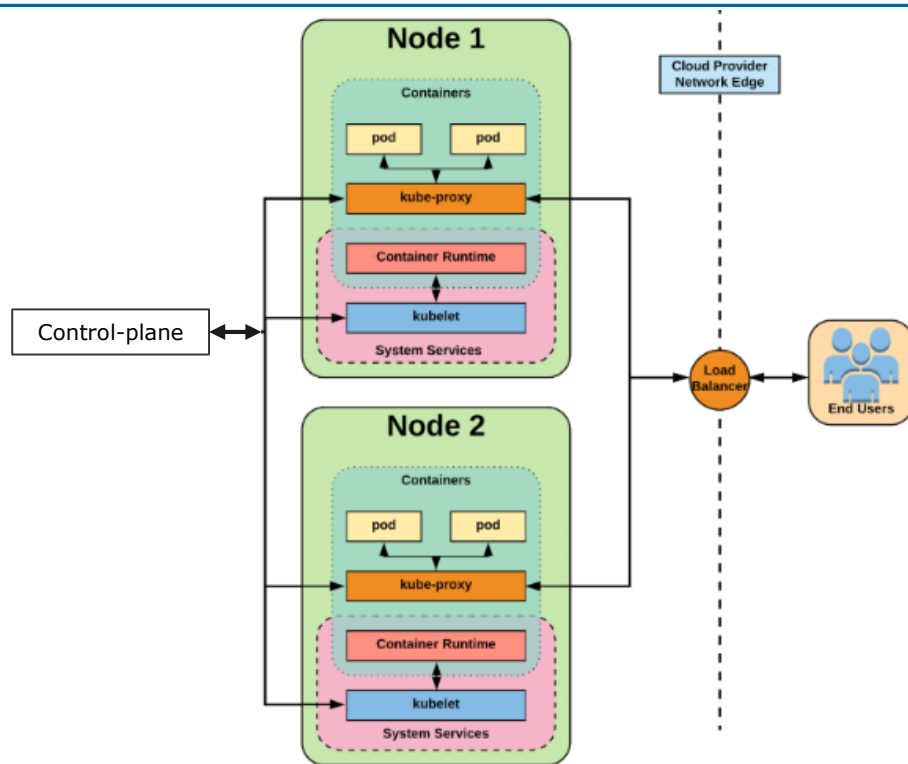
Architecture K8s



Architecture K8s : Control Plane



Architecture K8s : Components



Les risques ?

- ▶ Perdre le contrôle total du cluster
- ▶ Perte de données
- ▶ Usurpation d'identité
- ▶ Accès privilégié sur les hosts (destruction du système)
- ▶ Accès non autorisé à des projets au sein du cluster
- ▶ Cryptocurrency mining

2

Sécuriser la machine Host

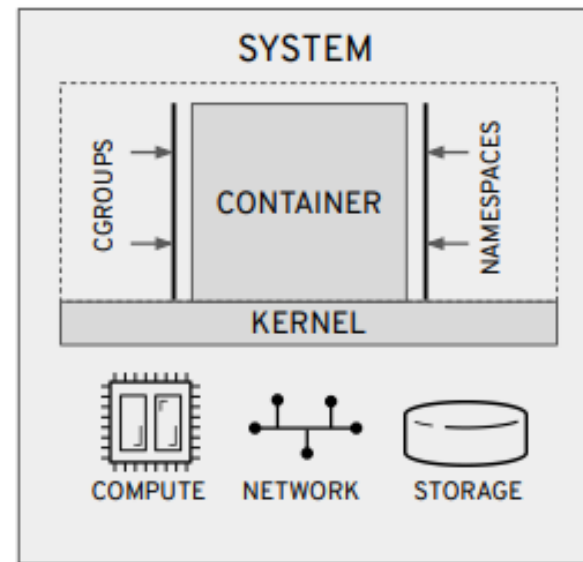
Sécurisation de la machine Host

Risques

- ▶ Accès aux configurations (kubelets, scheduler ..)
- ▶ Passer des requêtes non autorisés au serveur API
- ▶ Accéder aux données d'autres conteneurs
- ▶ **Détruire la totalité du système**

Remédiation

- ▶ Read-Only mount point sur les dossiers systèmes (/sys, /proc/sys,...)
- ▶ Renforcer l'utilisation des cgroups (limiter les ressources systèmes, temps CPU ...)
- ▶ Mise en place contexte de sécurité SELinux
- ▶ Limiter les appels systèmes avec Seccomp



3

Sécuriser le Traffic

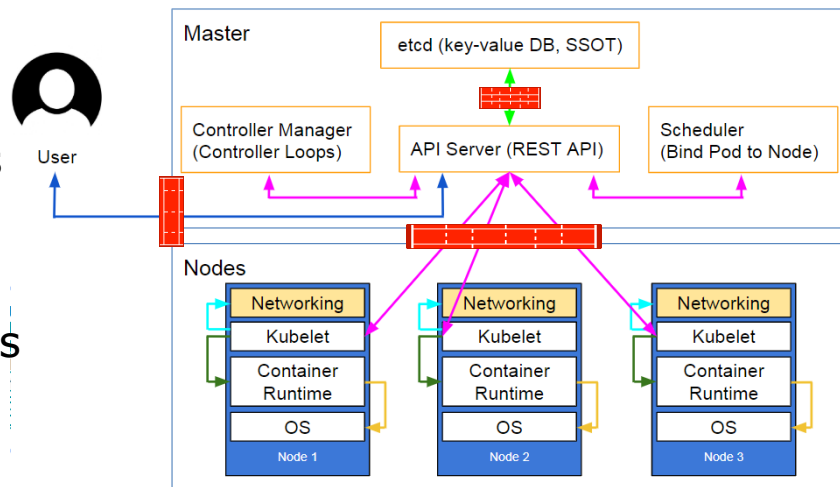
Sécuriser le Traffic: Transport Security

Risques

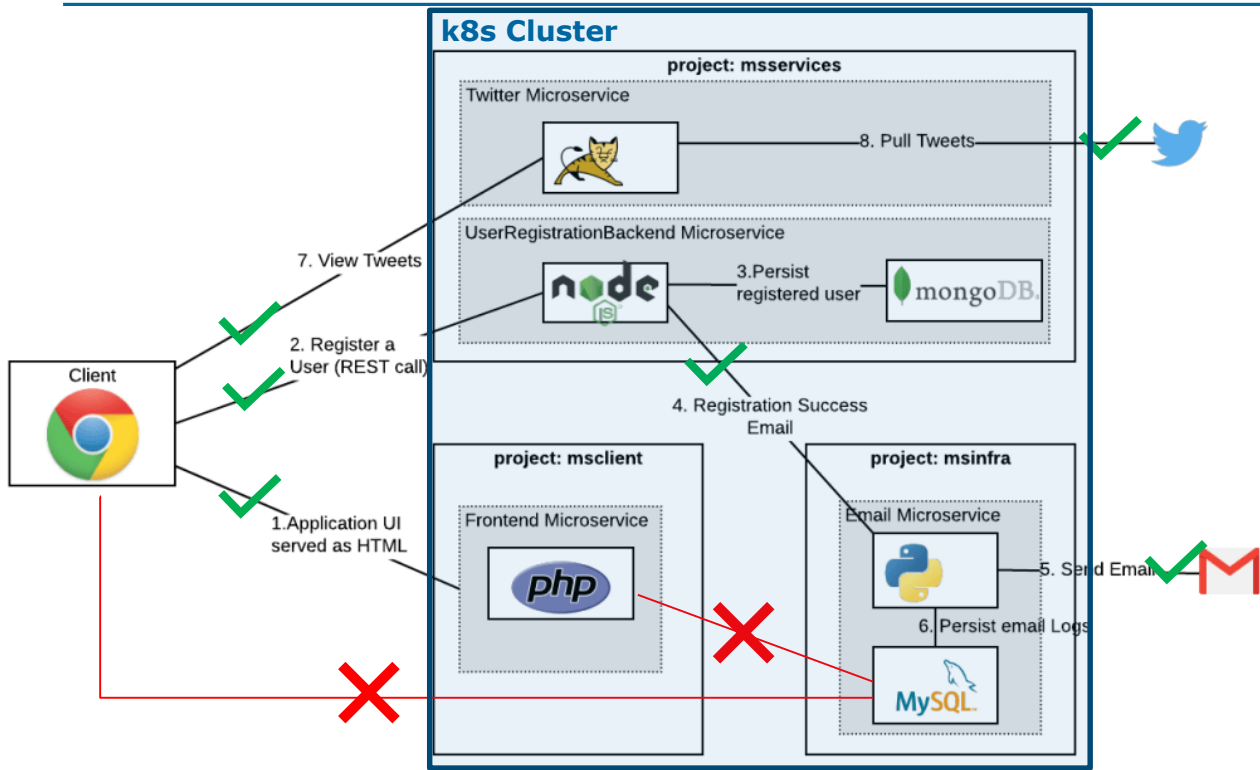
- ▶ Sniffing de données
- ▶ Vols d'identités / request forgery
- ▶ Connexions non autorisés entre conteneurs

Remédiation

- ▶ Toutes les communications doivent être protégées par des certificats TLS mutualisés
- ▶ Séparer le cluster kubernetes du cluster etcd et appliquer des règles de firewall
- ▶ Firewall interne et externe pour limiter les requêtes au serveur d'api
- ▶ Appliquer des politiques de networking (**NetworkPolicy**)



Sécuriser le Traffic: NetworkPolicy



Exemples de politiques :

- ▶ deny all
- ▶ Autoriser le trafic à l'intérieur du projet msinfra
- ▶ Autoriser twitter microservice a accéder l'api twitter
- ▶ ...

Sécuriser le Traffic: CNI

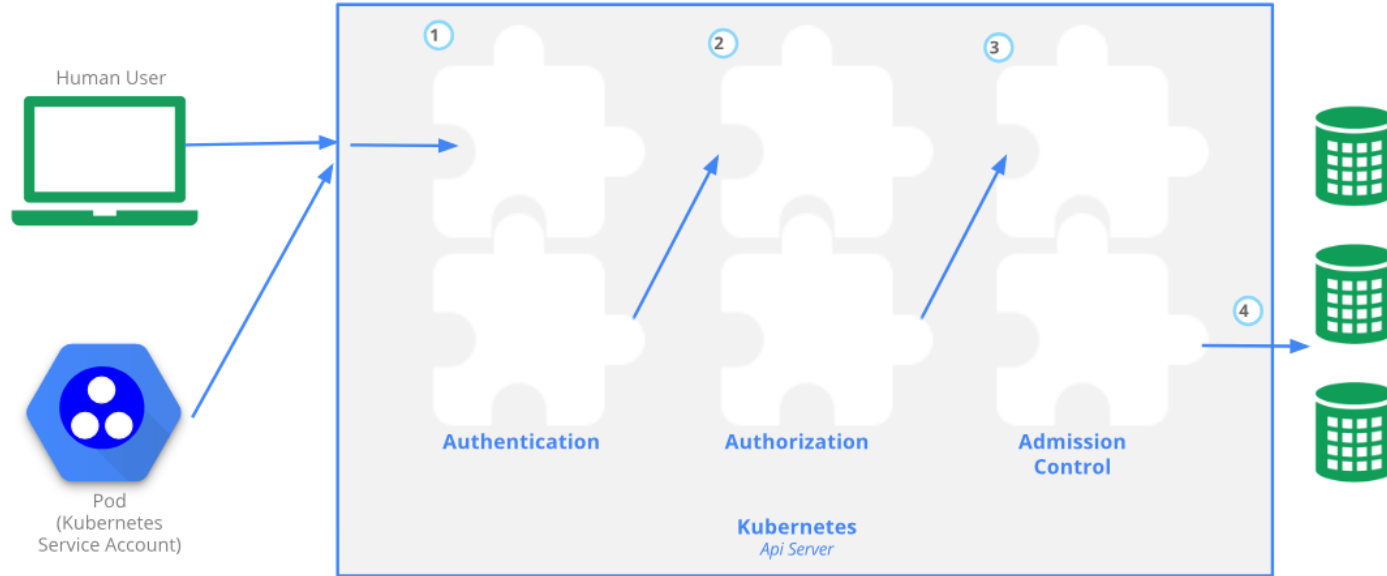
Recommandations:

- ▶ Choisir un CNI qui permet de :
 - Chiffrer les communications (selon le besoin)
 - Implémenter des networkpolicies
- ▶ Choisir la bonne taille des packets (MTU size)
- ▶ Prendre en considération la consommation de ressources (CPU & RAM) lors du choix d'un CNI



Sécuriser le Control-Plane

Sécuriser le Control-Plane : Mécanismes



Sécuriser le control-plane : Authentification

Risques

- ▶ Récupérer des Credentials (Tokens) et les utiliser pour attaquer le serveur d'api
- ▶ Intercepter un certificat et l'utiliser pour s'authentifier sur d'autres services (Etcd)

Remédiation

- ▶ Activer au moins deux modes d'authentification (certificats x509 et Tokens openID)
- ▶ Désactiver l'accès anonyme au cluster
- ▶ Désactiver l'accès non authentifié à l'api :
 - Au niveau de la communication avec les kubelets
 - Au niveau de la communication avec le cluster Etcd
- ▶ Rotation des certificats + Strong Cryptographic Ciphers
- ▶ Utiliser un unique Certificate Authority pour l'Etcd (différente que celle de k8s)

Sécuriser le Control-Plane : Autorisation

Risques

- ▶ Accéder non autorisé sur des services/namespaces
- ▶ Accès privilégiés sur des ressources confidentiels
- ▶ Perte de données accidentelles

Remédiation

- ▶ **Mettre en place des politiques RBAC :**
 - Restreindre les kubelets à lire que les objets dont ils ont besoin
 - Création de teams, users et les affecter à des namespaces spécifique...
- ▶ Désactiver toutes les autorisations attribués aux requêtes (activer par défaut)
- ▶ Utiliser le role *cluster-admin* qu'en cas de besoin

Sécuriser le Control-Plane : Admission Control

- ▶ L'admission control permet d'avoir un filtrage plus granulaire sur les requêtes après l'authentification et l'autorisation

Risques:

- ▶ Attaques dos
- ▶ Accès non autorisé a des images privés

Remédiation:

- ▶ Exemple de plugins d'admission control:
 - *Eventratelimit* : limite le nombre d'événements que peut accepter le serveur d'api en un lapse de temps
 - *Alwayspullimages*: force les nouveaux pods a faire le pull de l'image a chaque fois

Sécuriser le Control-Plane : Admission Control

2/2

- *Serviceaccount* : permet l'automatisation de la gestion des service accounts ([voir section: sécuriser le control-plane : service account](#))
- *Namespacelifecycle* : garantit que les objets nouvellement créés ne peuvent pas être dans un/des namespaces inexistants ou en cours de terminaison .
- *Podsecuritypolicy* : permet de faire un contrôle granulaire sur les actions/les données qu'un pod peut exécuter/accéder([voir section:securiser podsecurity policies](#))
- *Noderestriction* : restreint l'impact du kubelet sur les nodes et les pods ([voir section:sécuriser le kubelet](#))

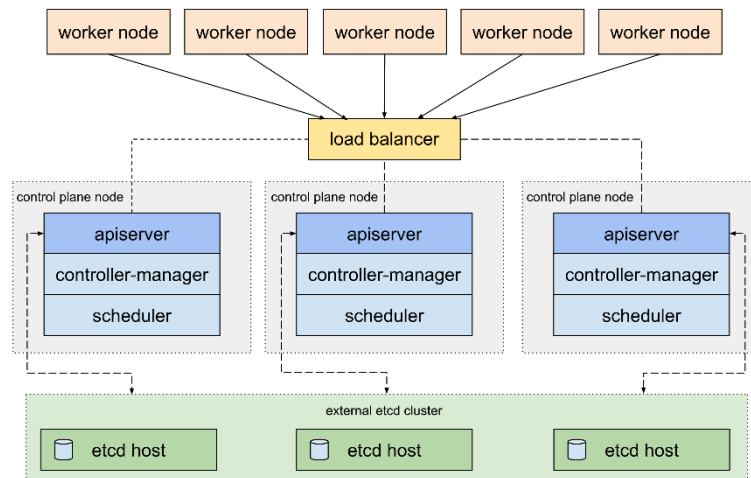
Sécuriser le Control-Plane : le cluster Etcd

Risques

- ▶ Un accès au cluster etcd c'est avoir un **accès root** sur tout le cluster
- ▶ Ajouter / supprimer des pods
- ▶ Modifier la configuration du cluster

Remédiation

- ▶ Restreindre les accès au cluster etcd
- ▶ Chiffrer les données sur l'etcd (encryption at REST)
- ▶ Rotation de la clé de déchiffrement



Sécuriser le control-plane : Service Account

- ▶ Les service account sont utilisés pour que les conteneurs d'un Pod puissent communiquer avec le serveur d'api directement

Recommandations:

- ▶ Utiliser un compte individuel pour chaque Controller (couplé avec RBAC)
- ▶ Ne pas utiliser le service account par défaut
- ▶ Au niveau du serveur API: vérifier le service account Token avant de valider l'authentification Token
- ▶ Automatisation de la gestion des service account

5

Sécuriser les composants
internes

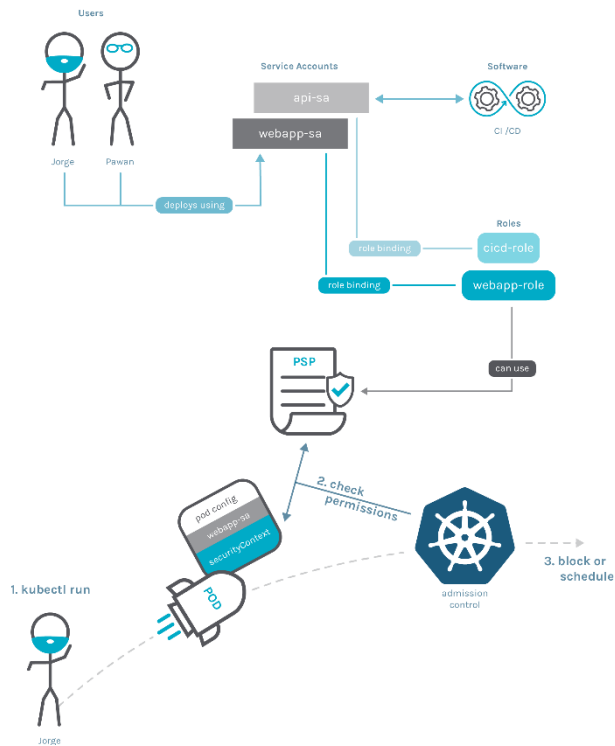
Sécurisation des composants internes

Composants à sécuriser :

- ▶ Images des conteneurs
- ▶ Pods
- ▶ Kubelets
- ▶ Secrets



Sécuriser les Pods: PodSecurity Policies



Risques

- Privilege escalation (accès root sur le node)
- Ecoute de trafic circulant dans le node
- Accès à des données d'autres conteneurs

Remédiation

- Lancer les pods **en mode non-root** avec un read-only root filesystem
- Désactiver le permissions de modifier les security contexts .
- Restreindre le partage de namespaces (hostpid, hostipc, hostnetwork...)
- Restreindre les capabilities du pod

Sécuriser les kubelets

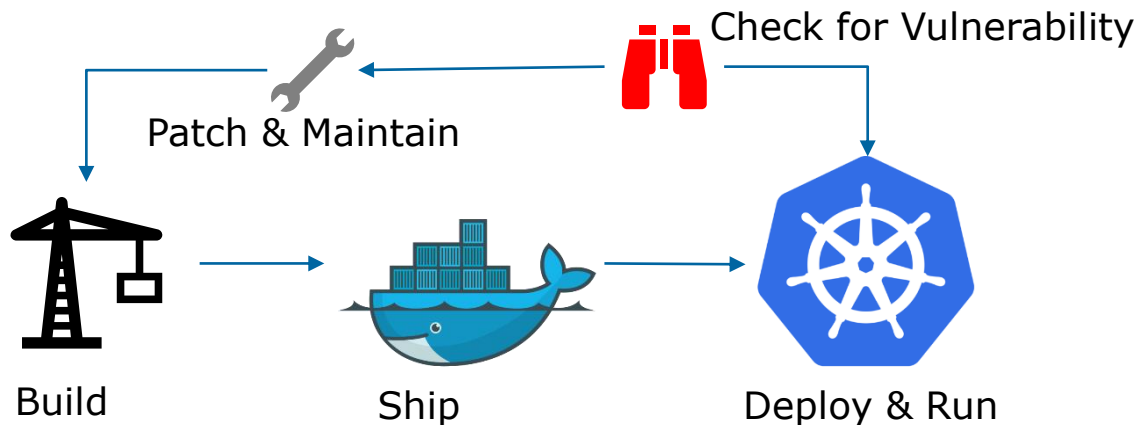
Risques

- ▶ Leak de données depuis n'importe quel pod du cluster
- ▶ Privileges escalation

Remédiation

- ▶ Désactiver les requêtes anonyme
- ▶ Activer l'autorisation explicite sur toutes les requêtes (RBAC, node)
- ▶ Désactiver le read-only port
- ▶ Rotation des certificats

Sécuriser les images 1/2



Recommandations

- ▶ S'assurer que l'image de base est sécurisé
- ▶ Scan de vulnérabilités à la fin du build
- ▶ Adopter les bonnes pratiques de développement d'une image de conteneur

Sécuriser les images 2/2

- ▶ Refuser les images de sources inconnues
- ▶ Vérifier l'intégrité de l'image avant le déploiement
- ▶ Check régulier de vulnérabilités sur les images utilisés
- ▶ Eviter l'update directe dans l'image du conteneur
- ▶ Adopter une approche devsecops avec une automatisation des tâches via des pipelines

Sécuriser les secrets

Risques (utilisation du built-in secret mechanism)

- ▶ Les secrets sont stockés en clair (pas chiffré)
- ▶ Un utilisateur qui consomme un secret peut le voir
- ▶ Dans le cas d'un root exploit , l'attaquant peut avoir tout les secrets
- ▶ Pas de tracking sur les changements des secrets
- ▶ Pas de contrôle granulaire sur les secrets

Recommandations

- ▶ L'utilisation du module secrets pré-intégré à kubernetes **n'est pas recommandé**
- ▶ L'utilisation d'une tierce partie est fortement recommandée (AWS secrets manager, google cloud platform KMS and azure key vault, hashicorp vault ..)

6

Monitoring & Logging

System monitoring

Monitoring:

- ▶ Création de politiques d'audit
- ▶ Health check
- ▶ Liveness/readiness/startup probes

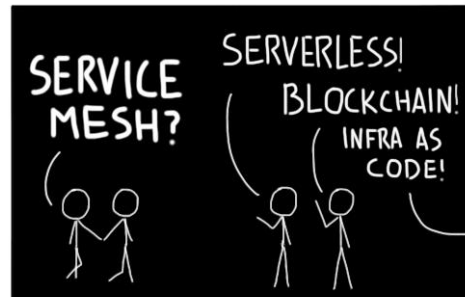
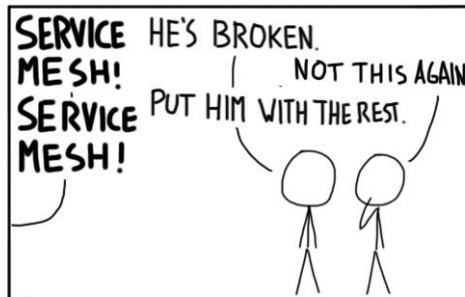
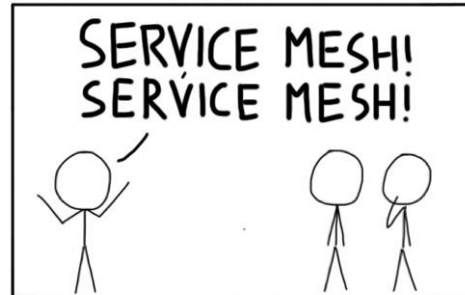
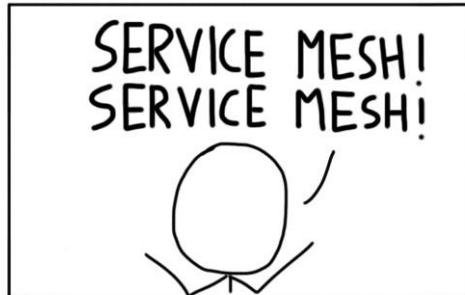
Logging :

- ▶ Mise en place d'une solution de centralisation de logs



Perspectives

Perspectives



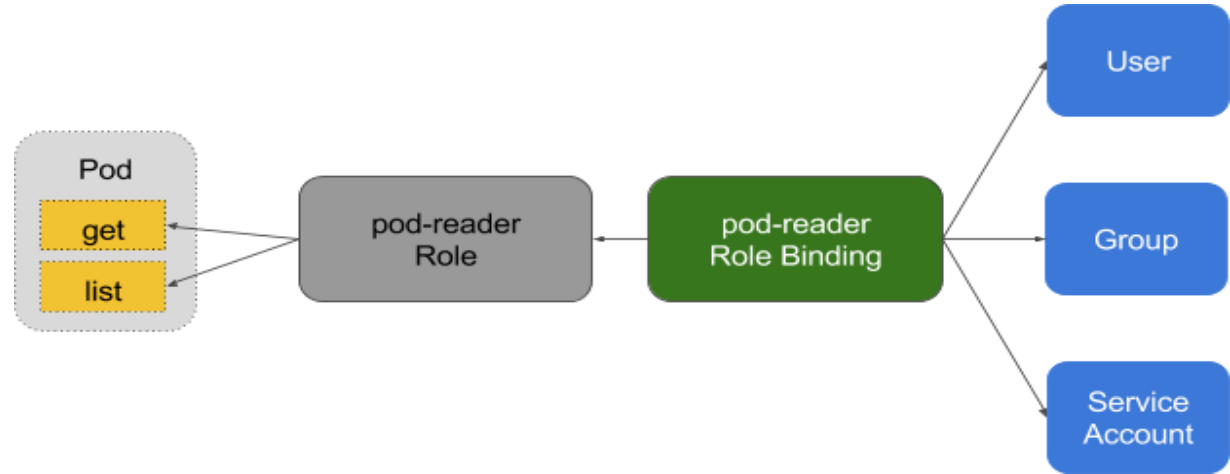
@sebiwicb



Appendix

RBAC

- ▶ Role-based access control provides fine-grained policy management for user access to resources, such as access to namespaces.



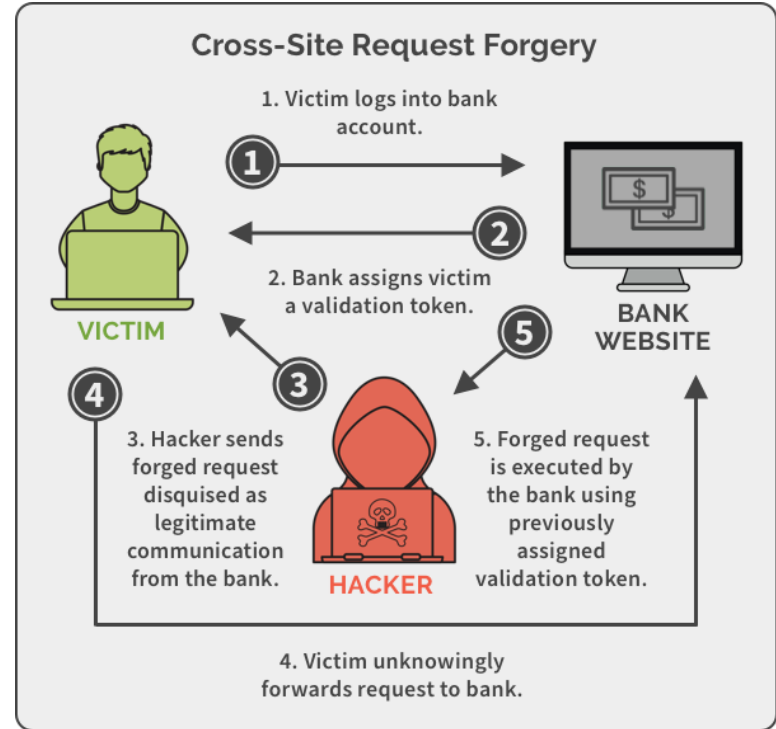
Strong Cryptographic Ciphers

The set of cryptographic ciphers currently considered secure is the following:

- ▶ TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- ▶ TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- ▶ TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305
- ▶ TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- ▶ TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305
- ▶ TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- ▶ TLS_RSA_WITH_AES_256_GCM_SHA384
- ▶ TLS_RSA_WITH_AES_128_GCM_SHA256

Cross-site Request forgery

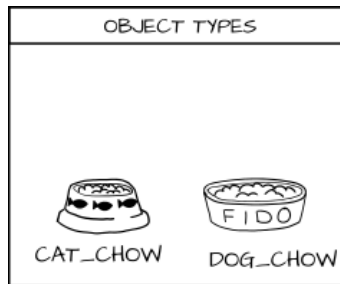
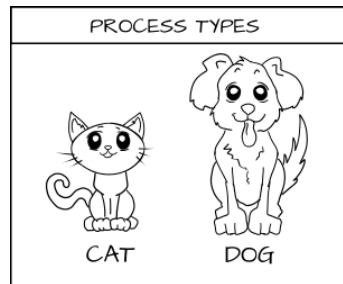
- ▶ Connu sous le nom de XSRF, sea surf et session riding
- ▶ Cross-site request forgery est une attaque qui force l'utilisateur final a exécuter des actions sur un site ou ils sont déjà authentifié sans leur consentement .
- ▶ Netflix, ING direct banking, youtube et mcafee secure avaient des vulnérabilités XSRF détecter sur leurs sites



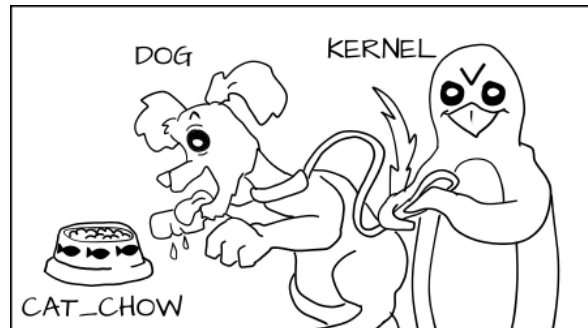
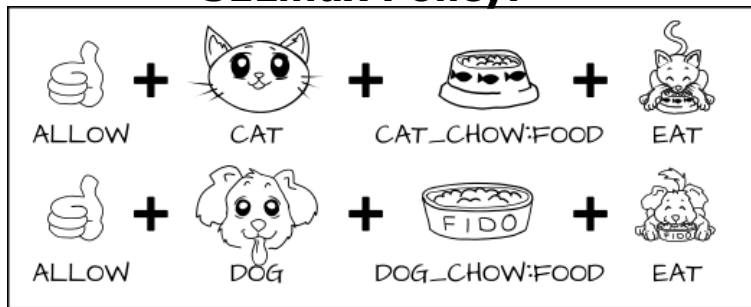
SELinux 1/2

- ▶ Selinux (security-enhanced linux) est un module de sécurité développé initialement par la NSA et repris par redhat
- ▶ Intégrer de base sur le kernel de centos, RHEL, fedora
- ▶ Il permet d'implémenter des règles (security control policy) qui limite les accès sur les fichiers et surveiller l'exécution des processus en cours d'exécution
- ▶ Les policies sont sous forme de labels de type : user::role::type::level
- ▶ Selinux en mode strict refuse tout par défaut
- ▶ **Dans le contexte k8s :**
 - Selinux va assurer qu'un conteneur peut que lire et exécuter de /usr
 - Le(s) process du conteneur peut seulement écrire dans le file system du conteneur

SELinux 2/2

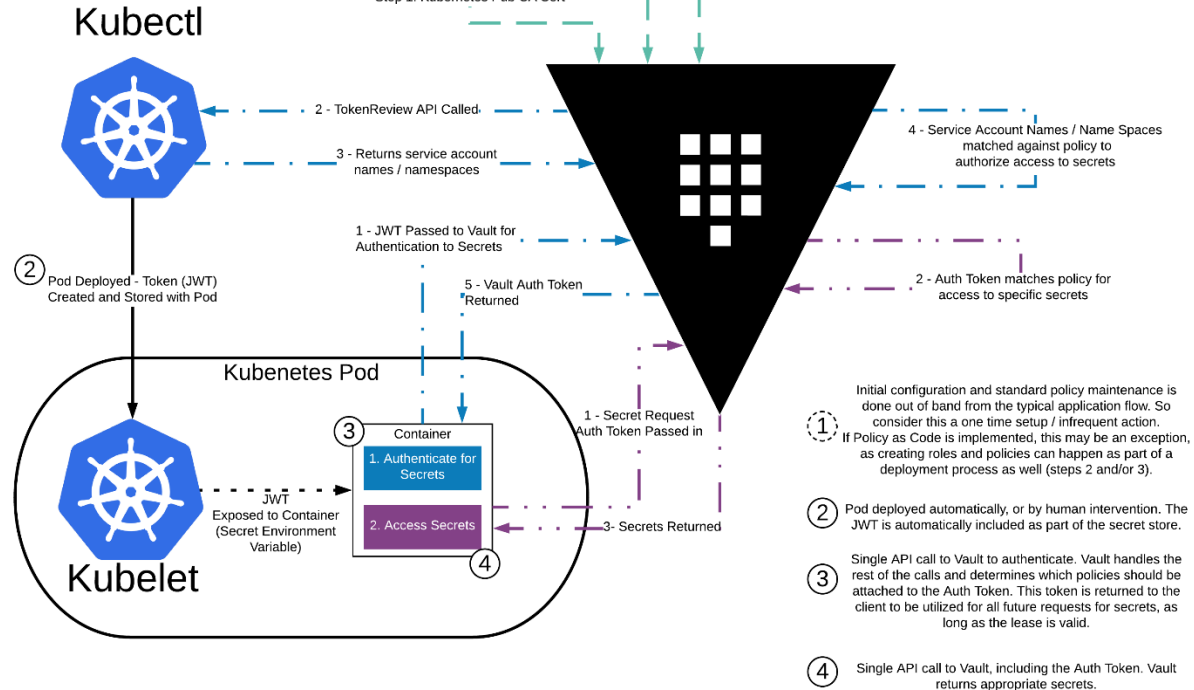


SELinux Policy:



SELinux assure que chaque process a le droit qu'a ce qui lui a été autorisé par les Security policies

Kubernetes Secret Authentication and Access with Vault



Thank you

For more information please contact:

M+ 33 6 18 47 95 46

mohamed.elajroud@atos.net

Atos, the Atos logo, Atos Syntel, Unify, and Worldline are registered trademarks of the Atos group. February 2020. © 2020 Atos. Confidential information owned by Atos, to be used by the recipient only. This document, or any part of it, may not be reproduced, copied, circulated and/or distributed nor quoted without prior written approval from Atos.

The Atos logo, featuring the word "Atos" in a bold, white, sans-serif font. The letter 'o' is stylized with a circular cutout in the center. The logo is positioned in the bottom right corner of the slide.