Hardening K8s Principes

06-02-2019

Atos

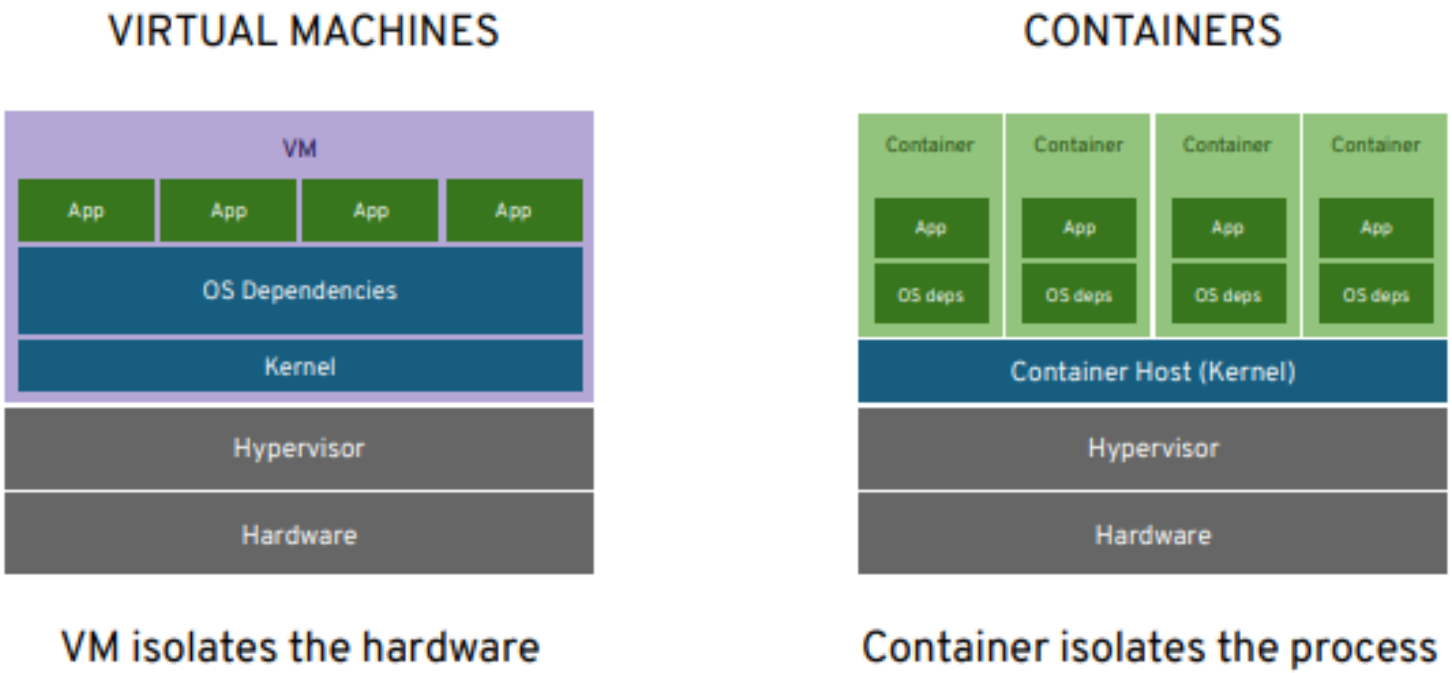
sommaire

- Introduction
- sécuriser le host
- sécuriser le Traffic
- sécuriser le control-plane
- Sécuriser les composants internes

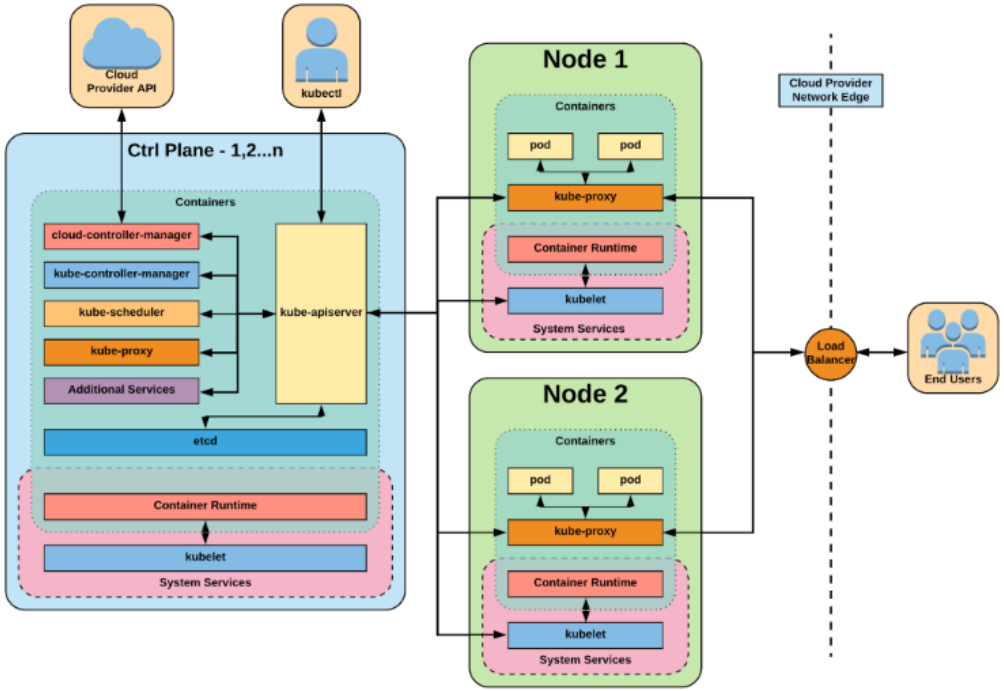
- Ces règles sont basées sur les principes du NIS et CIS.

Introduction

VMs & Containers



Architecture K8s



Les risques ?

- perdre le contrôle total du cluster
- perte de données
- usurpation d’identité
- accès privilégié sur les hosts (déstruction du système)
- accès non autorisé à des projets au sein du cluster
- cryptocurrency mining

Sécuriser le host

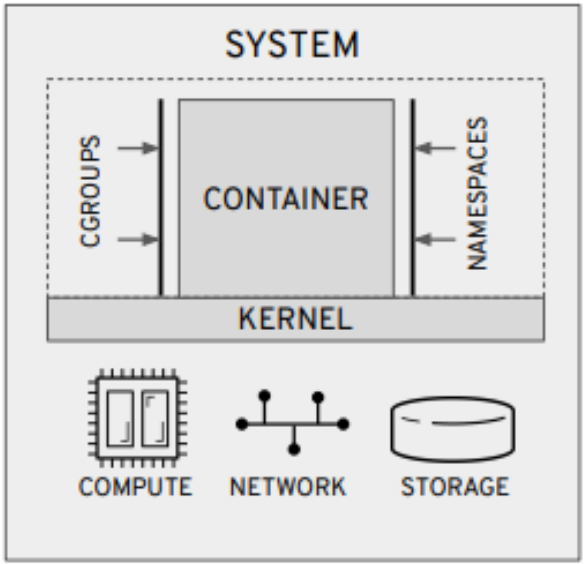
Sécurisation de la machine Host

Risques

- accès aux configurations (kubelets, Scheduler ..)
- passer des requêtes non autorisés au serveur API
- accéder au données d'autres conteneurs
- Détruire la totalité du système

Remédiation

- Read-Only mount point sur les dossiers systémes (/sys, /proc/sys,…)
- renforcer l'utilisation des Cgroups (limiter les ressources systémes, temps CPU …)
- mise en place contexte de sécurité SELinux
- limiter les appels systèmes avec SecComp



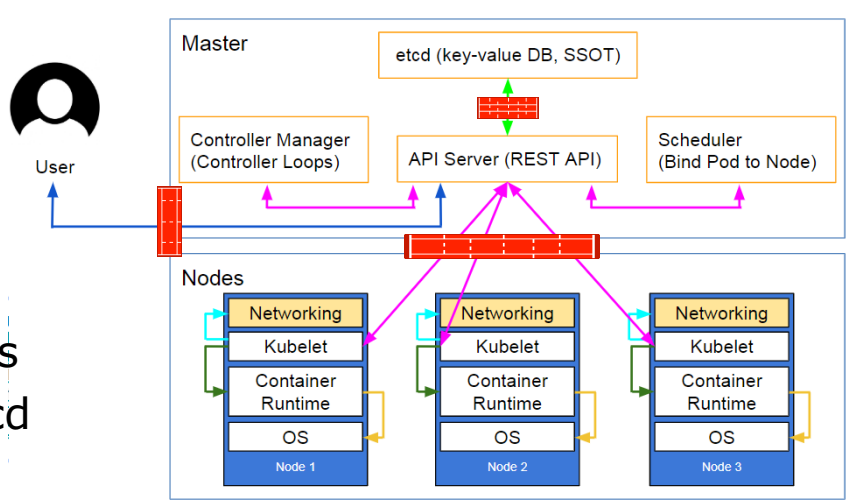
3

Sécuriser le traffic

Sécuriser le traffic: Transport Security

Risques

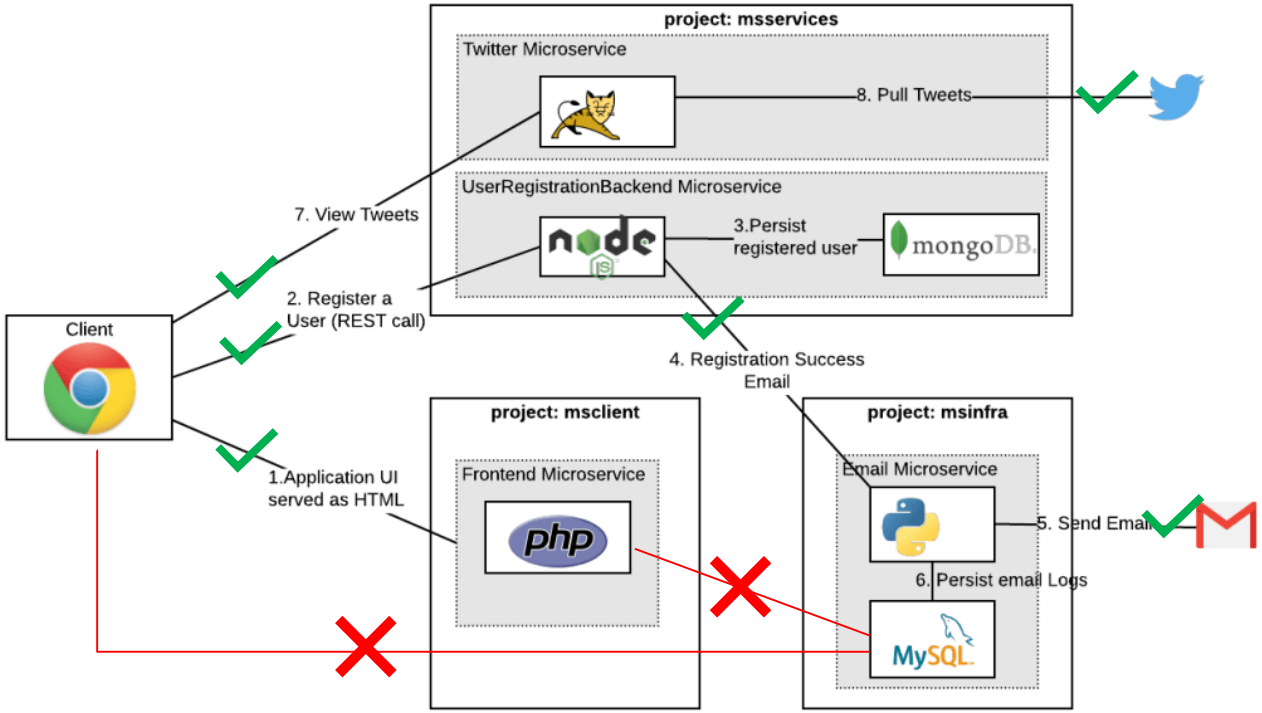
- Sniffing de données
- vols d'identités / Request forgery
- connexions non autorisés entre conteneurs



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

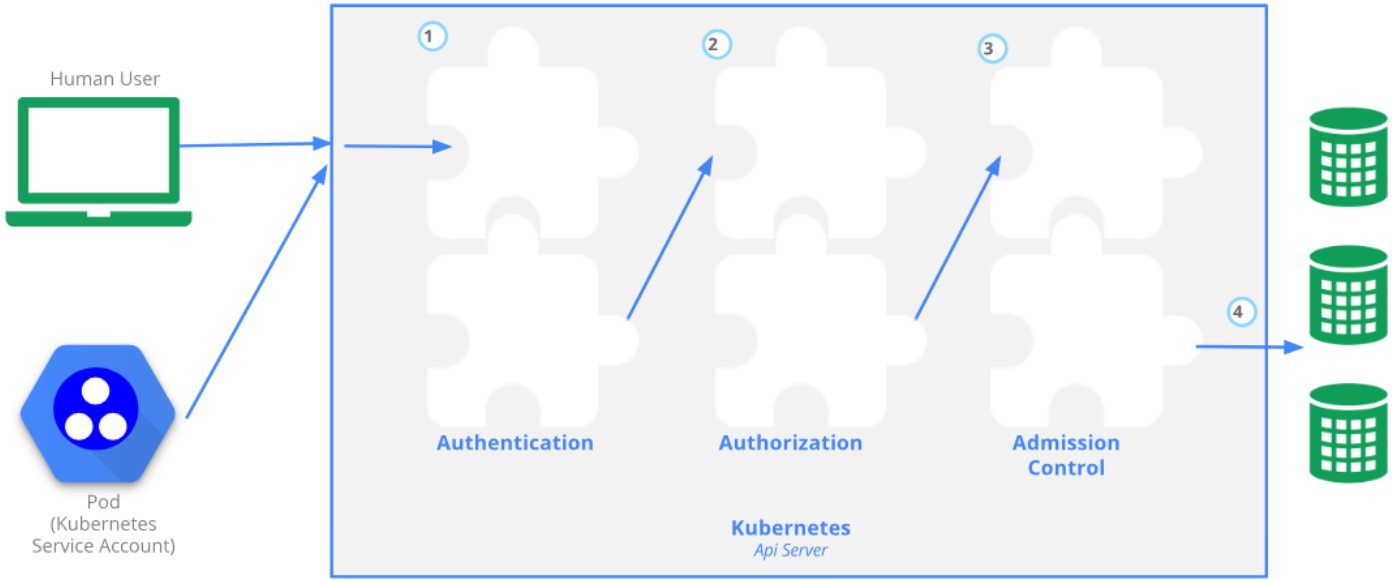


Exemples de Policies :

- deny all
- autoriser le traffic à l'intérieur du projet msinfra
- autoriser twitter Microservice a accéder l'API twitter
- …

Securiser 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'accés anonyme au cluster
- désactiver l'accés 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 a lire que les objets dont ils ont besoin
  - Création de teams, users et les affecter à des namespaces spécifique…
- désactiver toutes les autorisations attribués au 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'evenements 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*
- *NamespaceLifecycle*
- PodSecurityPolicy
- *NodeRestriction*

Securiser les composants internes

sécurisation des composants internes

Composants à sécuriser :

- etcd
- pods
- kubelet
- Containers
- Secrets

> IT DOESN'T MATTER HOW MANY LOCKS ARE ON YOUR DOOR IF YOUR WINDOW IS OPEN

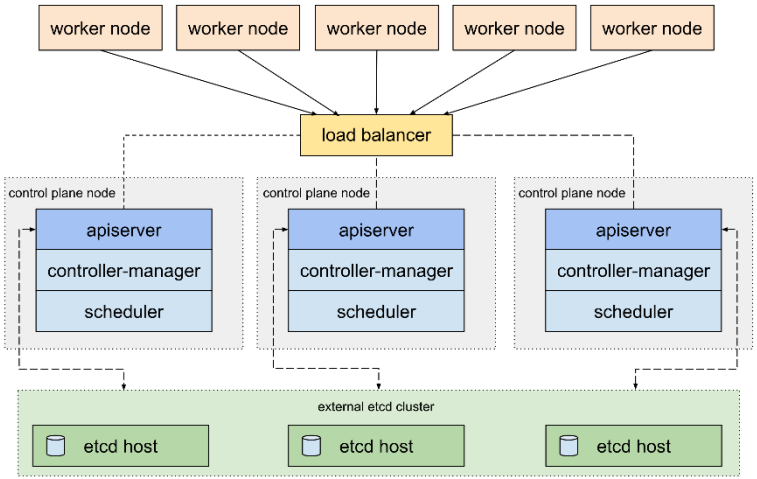
sécuriser le cluster Etcd

Risques

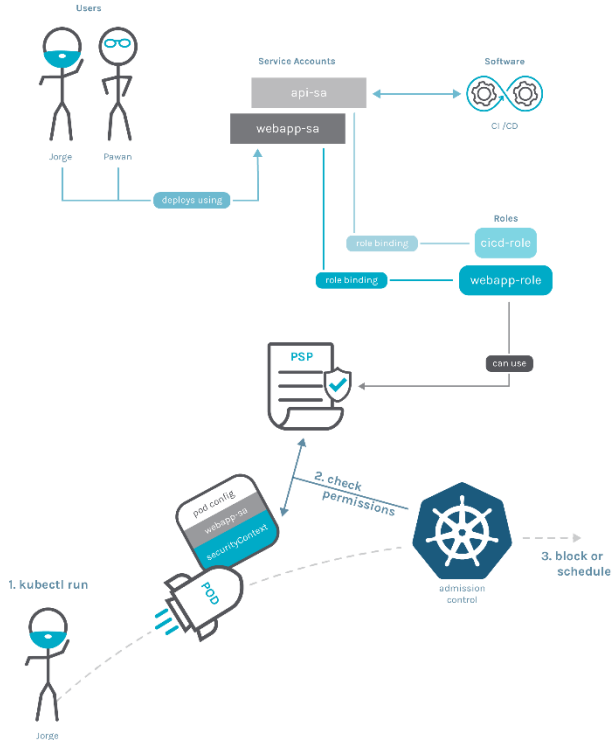
- un accès au cluster etcd c'est avoir un accès root sur tout le cluster
- ajouter / supprimer des pods
- modifier la configuration du cluster

Remédiation

- Restreindre les accées au cluster Etcd
- chiffrer les données sur l'Etcd (Encryption at REST)
- rotation de la clé de déchiffrement



sécuriser les pods: Podsecurity Policies

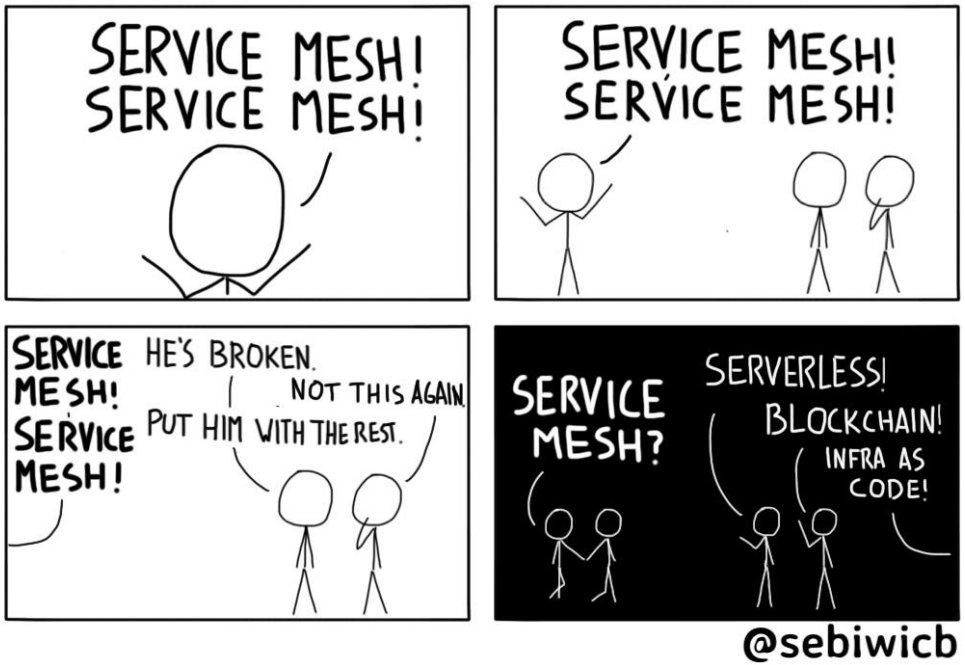


Thank you

For more information please contact:
M+ 33 6 18479546
mohamed.elajroud@atos.net

Atos

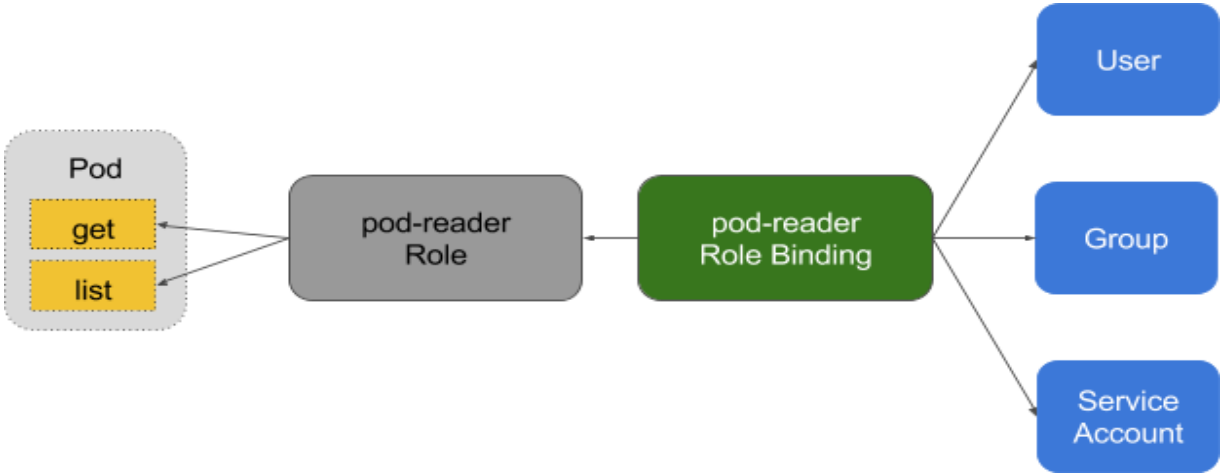
Perspectives



Appendix

RBAC

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

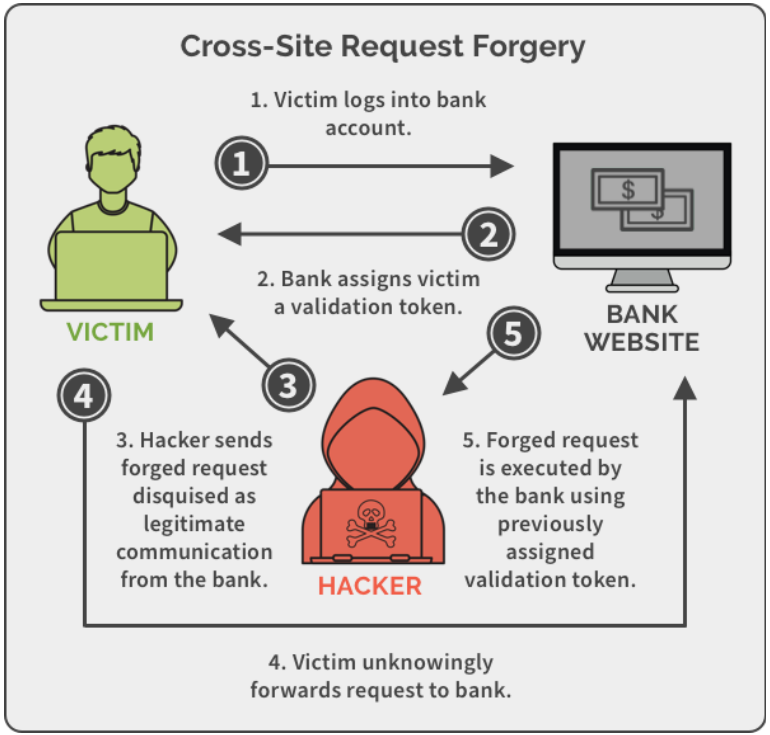


Strong Cryptographic Ciphers

- 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

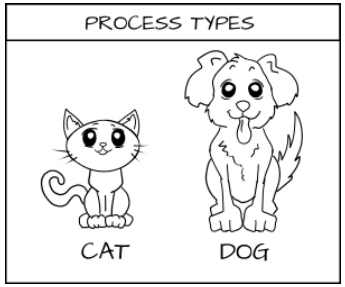
- Connu sous le nom de XSRF, Sea Surf et Session Riding
- Cross-Site Request Forgery est une attaque qui force l'utilisateur final a exécuter des actions sur un site ou ils sont déjà authentifié sans leur consentement .
- Netflix, ING direct banking, Youtube et McAfee secure avaient des vulnérabilités XSRF détecter sur leurs sites

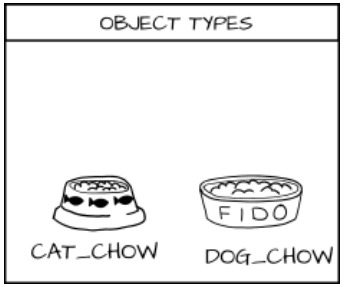


SELinux 1/2

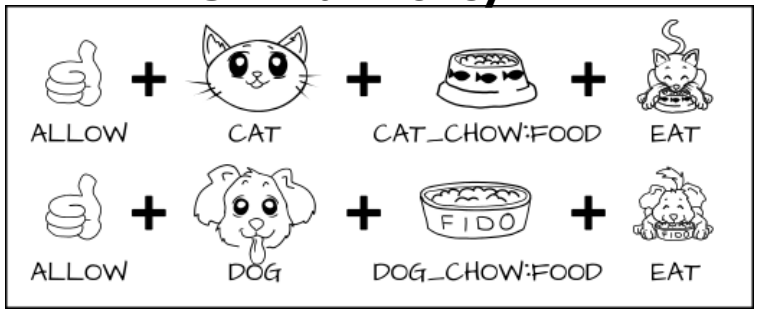
- Selinux (Security-Enhanced Linux) est un module de sécurité développé initialement par la NSA et repris par redhat
- intégrer de base sur le kernel de Centos, RHEL, Fedora
- il permet d'implémenter des régles (security control policy) qui limite les accés sur les fichiers et surveiller l'execution des processus en cours d'execution
- 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 ecrire dans le file system du conteneur

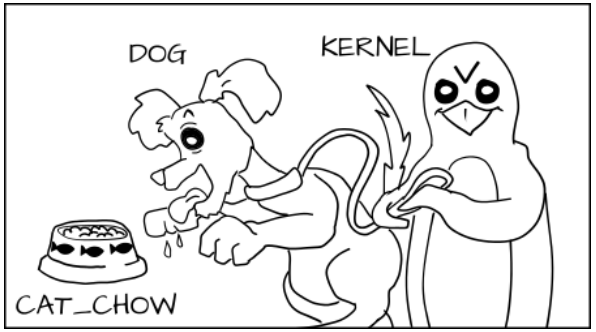
SELinux 2/2





SELinux Policy:





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