Sécurité

1. Scénario : Personne qui tente de modifier de l’information pour une transaction 2

2. Scénario : Un utilisateur malveillant essaie de changer la valeur d’une action 8

3. Scénario : Assurer l’intégrité et la confidentialité des données face aux personnes mal intentionnées. 10

**4.** Scénario **:** Sécuriser l’environnement de production 13

5. Scénario : Sécuriser les données de la bourse 16

6. Scénario : Covfefe, un hacker russe employé par Donald Trump, tente d’attaquer le système de manière à détourner des fonds pour la campagne présidentielle de 2020 19

7. Scénario : Prévenir un utilisateur d’effectuer des ventes et achats non autorisés à travers les systèmes 23

8. Scénario : Un administrateur tente de modifier le montant d’une transaction complétée 26

9. Scénario : Interception des données par un pirate lors des transactions. 29

10. Scénario : En dehors des heures de travail, l’attaquant va en profiter pour accéder aux données des clients de l'extérieur de l’organisation. 32

11. Scénario : Déconnecter un utilisateur lors d’une période d’inactivité de 4 minutes 35

12. Scénario 1 (Sécurité): Un développeur tente de se connecter au serveur de production (base de données + application principale) 38

13. Scénario : Un hacker essaye d’ajouter une transaction à son compte. 41

# Scénario : Personne qui tente de modifier de l’information pour une transaction

|  |  |
| --- | --- |
| **objectifs d'affaires** | Ne pas être victime d’une attaque extérieure |
| **Source** | Personne à l’extérieur de l’organisation |
| **Stimulus** | Tentative de modifier de l’information |
| **Artéfact** | Le système du poste de transaction |
| **Environnement** | Opération normale |
| **Réponse** | Aucune accès non autorisé n’est effectué sur le système |
| **Mesure de la réponse** | Nombre d’accès non autorisé |
| **Questions** | 1. Qu’est-ce qui détermine si la transaction est frauduleuse? 2. Notre Authentification est en combien de *Factor* 3. 2 |

## Tactique 1 : Detect Intrusion

**Description**: Le système compare les requêtes avec un pattern d’attaques déjà connues. Il se base entre autres sur la source de la requête.

**Justification**: Le système de transaction boursière ne peut pas se permettre d’effectuer des transactions qui ne sont pas représentative du marché.

## Tactique 2 : Authenticate actors

**Description**: Le système vas vérifier que les personnes ou les systèmes externes ont bien le droit de se connecter au système.

**Justification**: Il est important dans un système de transaction bancaire que celui-ci requiert une bonne authentification afin qu’on sache qui s’est connecté au système pour garder des traces sur les actions effectué par cet acteur.

## Tactique 3 : Limite access

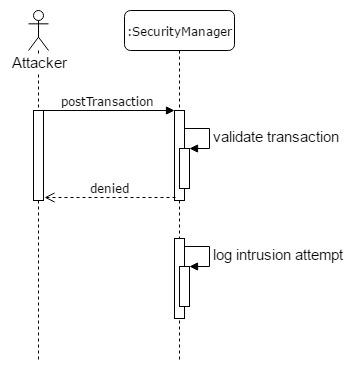
**Description**: Le système vas limiter les accès aux informations confidentielles des client ainsi que limiter l’accès aux opérations pouvant être effectuée sur les comptes.

**Justification**: Il est important dans un système de transactions bancaire de limiter l’accès aux informations bancaires ou personnelles des utilisateurs pour que celles-ci restent confidentielles. Il est aussi important de s’assurer que seul un utilisateur authentifié a accès aux opération affectant son comptes et seulement som compte.

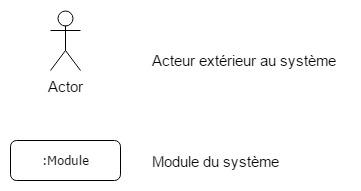
## Vue architecturale (à l’extérieur de la classe)

**Tactique 1:**

### Diagramme



### Légende



\*Les autres éléments sont standards aux diagrammes de séquences

### Texte de description du diagramme

Un utilisateur externe (qui a contourné le système d’authentification d’une quelconque façon) tente d’effectuer une transaction. Le *SecurityManager* tente de valider la transaction, mais il détecte qu’il s’agit d’une attaque. Il log ensuite les confirmations concernant la tentative d’intrusion.

### 

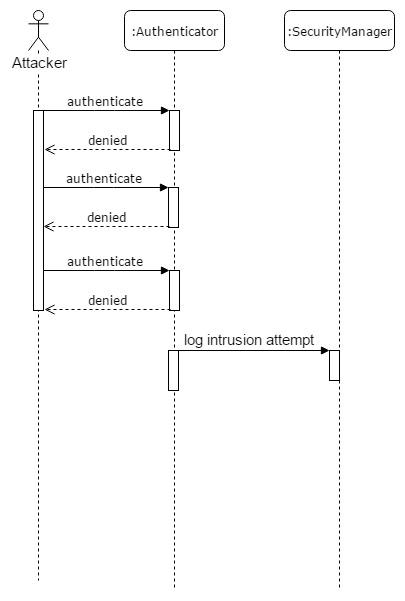
### 

### Table de description des éléments du diagramme

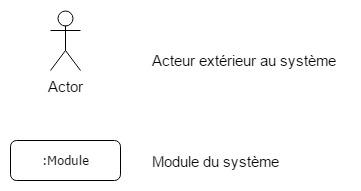
|  |  |
| --- | --- |
| **Élément** | **Description** |
| Attacker | L’utilisateur externe qui tente d’attaquer le système. |
| SecurityManager | Module du système responsable de détecter l’attaque. |
| postTransaction | L’attaque que l’utilisateur tente d’effectuer. |
| validate transaction | Le SecurityManager qui valide la transaction pour détecter s’il s’agit d’une attaque. |
| log intrusion attempt | Le SecurityManager enregistre les informations relatives à l’attaque |

**Tactique 2:**

### Diagramme



### Légende



\*Les autres éléments sont strandards aux diagrammes de séquences

### Texte de description du diagramme

Un utilisateur externe tente de s’authentifier plusieurs fois, mais il est rejeté plusieurs fois pas le systèm d’authentification. Il enregistre ensuite les informations relative à l’intrusion pour les analyses ultérieures.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Élément** | **Description** |
| Attacker | L’utilisateur externe qui tente d’attaquer le système. |
| Authenticator | Module responsable de l’authentification des utilisateurs |
| SecurityManager | Module du système responsable de détecter l’attaque. |
| authenticate | La tentative de connexion de l’utilisateur qui veut effectuer une attaque. |
| log intrusion attempt | Le SecurityManager enregistre les informations relatives à l’attaque |

**Représentation générale:**

### 

### Texte de description du diagramme

Un client qui veut effectuer une transaction doit passer d’abord par le module Security Manager, puis par Permission Manager et enfin par Post Trading.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Élément** | **Description** |
| Client authentifié | Utilisateur externe au système |
| SecurityManager | Module du système responsable de détecter l’attaque. |
| Permission Manager | Module du système responsable de déterminer si l’utilisateur à les permissions nécessaires à la transaction qu’il veut effectuer |
| Post trading | Module du système responsable des transactions. |

### Texte décrivant la relation entre les éléments et les tactiques

Premièrement le client vas envoyer un demande de connection au *Authentificator,* par la suite celui-çi vas vérifier si cette demande correspond avec un utilisateur qui a accès au système. Si les *credentials* sont erroné pour plus de 3 tentatives le client vas être banni du système. Une fois le client authentifié, il vas envoyer ses requêtes au Security manager qui vas effectuer la tactique de *Detect intrusion* en regardant la source de la requête ainsi que son *payload* pour déterminer si celle-ci est frauduleuse ou non. Par la suite une fois la transaction vérifié le *Security Manager* vas envoyer la transaction au *Permission Mamager* qui vas appliquer la tactique *Limit Access.* Le *Permission Manager* vas vérifier que l’utilisateur a les droits pour effectuer la transaction demandé. Une fois la transaction vérifié elle sera finalement envoyé au module de *Post-trading*

# Scénario : Un utilisateur malveillant essaie de changer la valeur d’une action

|  |  |
| --- | --- |
| **objectifs d'affaires** | Empêcher la modification des valeurs sur la base de données |
| **Source** | Utilisateur malveillant provenant d’un emplacement distant |
| **Stimulus** | essaie de modifier la valeur d’une action |
| **Artéfact** | Données dans la base de données |
| **Environnement** | Opération normale |
| **Réponse** | système empêche la modification de données |
| **Mesure de la réponse** | l’utilisateur ne peut accéder aux données |
| **Questions** | 1. D'où proviennent la plupart des accès aux données. 2. Que sont les droits possibles pour les utilisateurs sur les données. |

## Tactique 1 : Detect intrusion

**Description**: Comparer le trafic ou des requêtes aux serveurs avec des métriques connues pour découvrir des anormalités dans le serveur.

**Justification**: Comparer la provenance de l’adresse IP avec une table d’adresse IP de requête courante et lever un drapeau lorsque l'IP n’est pas sur la table.

## Tactique 2 : Encrypt data

**Description**: Crypter les données de la base de données afin d'empêcher la modification de ceux-ci.

**Justification**: Utiliser l’algorithme de chiffrement des données AES256 sur la base de données pour empêcher la modification des valeurs sur les actions emmagasinées sur la base de données

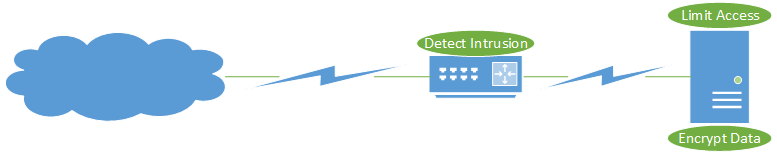
## Tactique 3 : Limit Access

**Description**: Limiter l’utilisation de la base de données

**Justification**: Enlever les droits de modifications sur la base de données (personne ne devrait pouvoir modifier).

## Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende

|  |  |
| --- | --- |
|  | Nuage qui représente les accès faits de l’extérieur de la compagnie |
|  | Appareil de routage permettant de faire l’analyse de paquets IP |
|  | Connecteur entre les appareils |
|  | Serveur de base de données |

### Texte de description du diagramme

Les requêtes aux serveurs de base de données sont analysées par l’appareil d’analyse pour vérifier si une requête ne fait pas partie de la liste d’adresse IP et ensuite lever un drapeau si une adresse IP n’est pas sur la liste. Les données sur le serveur de base de données sont cryptées pour empêcher la lecture de données sur le serveur et les données sont en lecture seule ce qui empêche la modification de données par un utilisateur sur la base de données.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Appareil d’analyse d’adresse IP | Cet appareil est utilisé pour détecter les adresses IP potentiellement malveillantes et lever un drapeau si une de ses adresses est potentiellement dangereuse. |
| Serveur de base de données | Endroit où le cryptage des données est fait pour empêcher un utilisateur d’avoir les données de façon lisible. La base de données est aussi en lecture seulement pour empêcher la modification des données par un utilisateur. |

### Texte décrivant la relation entre les éléments et les tactiques

**Detect Intrusion** est utilisé pour analyser les adresses IP et ensuite valider si une adresse sur la liste est sans risque et peut être ajouté à la liste des adresses IP acceptée ou si cette adresse devait être bloquée dans le futur.

Les tactiques **Encrypt Data** et **Limit Access** sont utilisés conjointement pour empêcher la modification des valeurs dans la base de données et crypter les données pour empêcher le vol d’information en texte claire par un utilisateur malveillant.

# Scénario : Assurer l’intégrité et la confidentialité des données face aux personnes mal intentionnées.

|  |  |
| --- | --- |
| **objectifs d'affaires** | Empêcher de compromettre les données enregistrées sur les transactions des dernières années. |
| **Source** | Une personne externe ou un employé. |
| **Stimulus** | Un accès non autorisé aux données est détecté. |
| **Artéfact** | Base de données de l’historique des transactions (850 TB actuellement) |
| **Environnement** | En ligne et connecté au réseau. |
| **Réponse** | * On interdit l'accès et l’utilisation des données provenant de l’adresse IP ou du code employé. * Sauvegarde des informations liées à la tentative de connexion. * Envoi d’une alerte aux administrateurs du système. |
| **Mesure de la réponse** | * Le temps avant de détecter une intrusion. * Le nombre d’attaques tentées et échouées. * La quantité de données qui ont été compromises. |
| **Questions** | 1. Pour qui les données sur les transactions peuvent-elles être utiles ? 2. Est-ce que les données d’historique sont cryptées ? 3. Combien de personnes légitimes ont accès au système et aux données ? 4. Est-ce que les patrons de requête diffèrent beaucoup entre eux ? |

## Tactique 1 : Detect intrusion

**Description** : Détection des intrusions en filtrant les requêtes aux serveurs.

**Justification** : Nous trouvons les intrusions en observant les patrons de requête. Cela nous permet d’isoler rapidement les requêtes inhabituelles lorsqu’il y a par exemple des entrées non habituelles.

## Tactique 2 : Revoke Access

**Description** : L’adresse IP de la personne mal intentionnée sera bloquée par les routeurs afin de couper l’accès immédiatement.

**Justification** : Ce type de verrouillage permet d’agir plus rapidement que d’intervenir sur tous les systèmes et permet de filtrer seulement l’intrus sans affecter les usagers.

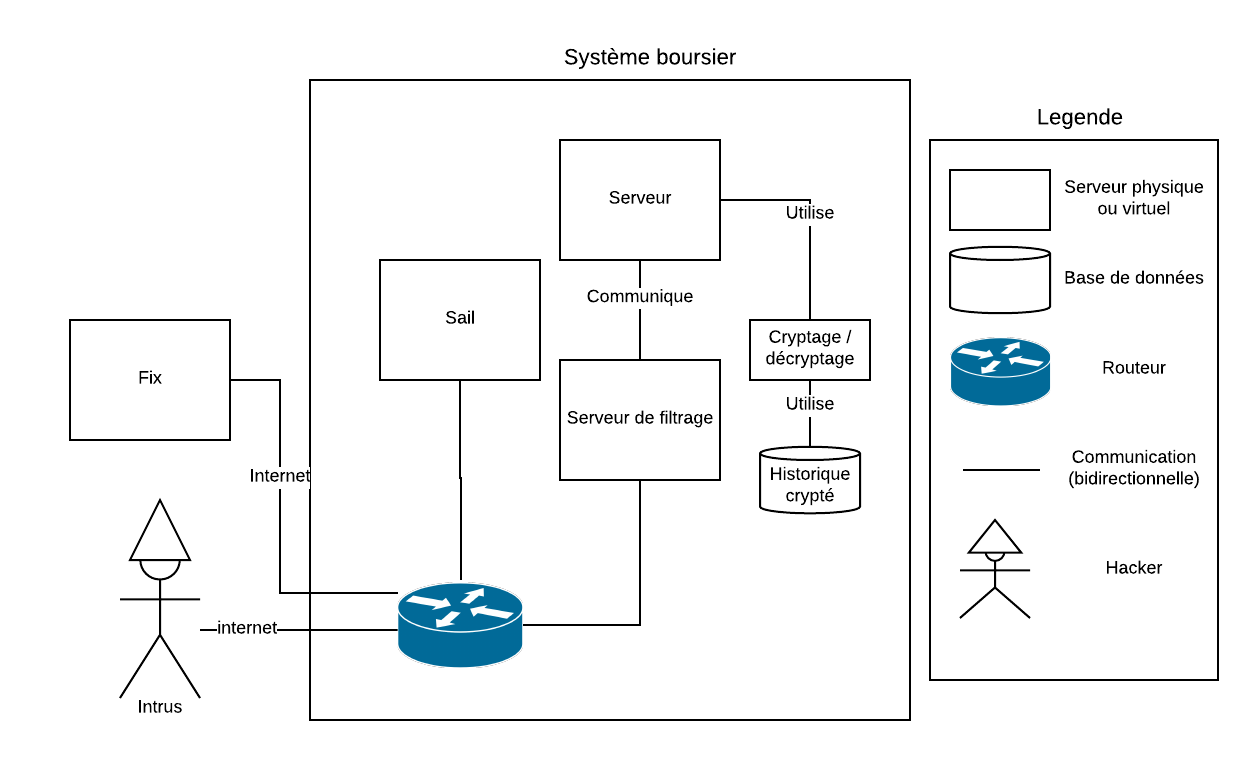
## Tactique 3 : Encrypt Data

**Description** : Cryptage des données avant de les stocker sur la base de données.

**Justification** : Crypter les données sur les transactions avec une clé de cryptage permet de rendre illisibles ces données si on les récupère sans en avoir d’autorisation. Il serait intéressant d’avoir plusieurs clés de cryptage.

## Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Texte de description du diagramme

Ce diagramme de la famille «module» représente la vue d’ensemble de l'accès aux données de l’historique des transactions du point de vue de la sécurité. Le système Fix étant externe au système boursier, il communique par internet. Par contre, Sail est interne au système boursier, il communique alors directement avec le routeur par intranet. Les communications passent d’abord par le routeur qui applique un filtre sur les adresses IP communiquant. Ces requêtes passent ensuite par un serveur de filtrage qui observe les patrons. Si une requête sort de l’ordinaire, il ajoute l’adresse IP à la liste noire du routeur. Si la requête passe le filtre, elle est traitée par le serveur de requête. Le serveur passe par la machine de décryptage afin d'obtenir les données de la base de données.

Dans notre scénario, l’intrus se rendra une première fois au serveur de filtrage et verra sa requête refusée. Le routeur lui bloquera ensuite l'accès au système boursier.

### 

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Intrus | Personne mal intentionnée tentant une intrusion. |
| Fix | Interface usager de la bourse de Montréal (interne). |
| Sail | Interface usager standard de la bourse (externe). |
| Routeur | Routeur par lequel passent les requêtes provenant autant d’internet que de l’interne. |
| Serveur de filtrage | Serveur filtrant les entrées des requêtes pour **détecter les intrusions**. |
| Serveur | Serveur principal du traitement des requêtes du système boursier. |
| Cryptage/décryptage | Serveur de **cryptage** et de décryptage des données sur l’historique des transactions. |
| Base de données d’historique des transactions | Base de données contenant les données cryptées sur l’historique de transactions des dernières années. |

### Texte décrivant la relation entre les éléments et les tactiques

On peut voir que pour la tactique de détection des intrusions, on fait du filtrage sur les requêtes qui passent par le routeur et ensuite dans le serveur de filtrage. On filtre alors les paramètres des requêtes et on compare avec les patrons de requêtes légitimes. Ainsi, lorsqu’une requête n’est pas conforme, on peut déduire qu’on a une intrusion.

Ensuite, pour la tactique de Revoke Access, on la retrouve sur le routeur qui bloque donc les adresses IP qui ont tenté de faire une intrusion dans le cas des requêtes faites de l’externe. Dans le cas des requêtes depuis l’interne avec le système Sail, on bloque l’accès de l’utilisateur qui s’était préalablement authentifié au système.

Finalement, un serveur de cryptage et décryptage est présent pour répondre à notre tactique de cryptage des données. Celui-ci s’occupe de crypter les données entrant dans la base de données et de décrypter les données sortantes à l’aide de clés de cryptage et de calculs. Même si l’intrus obtient les données sur les transactions, il ne pourra pas les lire.

# Scénario **:** Sécuriser l’environnement de production

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Sécuriser l’environnement de production |
| **Source** | Développeur |
| **Stimulus** | Accès non autorisé par un développeur |
| **Artéfact** | Données du système |
| **Environnement** | Complètement fonctionnel |
| **Réponse** | * Les données sont protégées d’accès à un développeur * On garde les demandes d’accès en mémoire |
| **Mesure de la \_ réponse** | * Nombre de demandes d’accès. * Quantité de données accessibles. |
| **Questions** | 1. Quelle est la quantité de ressources à protéger? 2. Combien d’utilisateurs et de rôles le système devra-t-il gérer? |

## Tactique 1 : Limiter l’accès

## Description: Limiter l’accès à des données avec un firewall par exemple.

**Justification:** Restreindre l’accès permet d’avoir plus de contrôle sur la sécurité des données du système.

**Tactique 2 : Authentifier les acteurs**

**Description:** Demander une authentification aux utilisateurs anonymes avant de permettre l’accès

**Justification:** L’authentification des acteurs permet de classer qui accède présentement aux données. Le système peut par la suite prendre des actions selon l’acteur en question.

**Tactique 3 : Enlever le droit d’accès**

**Description:** Dépendant de l’utilisateur, on restreint l’accès à la donnée demandée.

**Justification:** Une fois que nous connaissons les acteurs à l’aide de la tactique 2, il est possible d’enlever le droit d’accès aux utilisateurs qui ne devraient pas pouvoir consulter ces données.

## Vue architecturale (à l’extérieur de la classe)

### Diagramme

Séquence ou cas d’utilisation



### Légende

Les carrés représentent des composants.

Le rectangle arrondi représente un Firewall.

Le bonhomme allumette représente un utilisateur.

Une ligne pleine représente une connexion .

Une ligne pointillée représente une connexion conditionnelle.

### Texte de description du diagramme

Le diagramme représente un utilisateur qui tente de se connecter. Toutes les connexions passent d’abord par un Firewall qui permet d’autoriser et de protéger les composantes Sail et Fix. ACL et LOGS sont directement liés avec le firewall. Par exemple, dès qu’un utilisateur passe par le firewall, celui-ci ajoutera un événement au composant LOGS. Également, le firewall se servira du composant ACL afin de savoir si l’utilisateur a les permissions pour accéder à la composante demandée.

### Table de description des éléments du diagramme

ACL (Access Control List) : Pour contrôler les accès

Logs : Pour analyser les interactions avec les systèmes et restreindre l’accès, au besoin

Firewall : Bloque ou accorde l’accès à l’utilisateur

Utilisateur : n’importe qui utilisant l’un des systèmes

### Texte décrivant la relation entre les éléments et les tactiques

La composante ACL fait référence à l'authentification des acteurs. Pour ce qui est du Firewall, il se base sur la tactique qui limite l’accès aux ressources et aux composantes. De plus, SAIL et FIX sont également protégés derrière le firewall ce qui représente notre troisième tactique : enlever le droit d’accès.

# Scénario : Sécuriser les données de la bourse

|  |  |
| --- | --- |
| **Objectifs d'affaires** | On veut pouvoir éviter les attaques et limiter les dégâts le plus possible en cas de faille. |
| **Source** | Un humain à l’intérieur où à l’extérieur de l’organisation. |
| **Stimulus** | Une attaque est faite afin d’accéder ou de modifier des données. |
| **Artéfact** | Le système lui-même et ses données. |
| **Environnement** | Le système est allumé et connecté à un large réseau. |
| **Réponse** | Les données ne sont pas manipulées par des gens qui ne devraient pas y avoir accès  Les acteurs dans une transaction sont identifiées avec une assurance suffisante. |
| **Mesure de la réponse** | Combien d’attaques le système a pu résister avec succès.  La quantité de données vulnérables pour une attaque donnée. |
| **Questions** |  |

## Tactique 1 : Authentification des acteurs

**Description**: Authentifier les acteurs pour s’assurer de leur identité

**Justification**: En authentifiant les acteurs avec une combinaison de nom d’utilisateur / mot de passe ou de clé SSH, il est beaucoup plus facile de restreindre l’accès au système seulement aux acteurs autorisés. Il y a toujours une possibilité que quelqu’un de malveillant obtienne des informations d’authentification, mais les risques sont relativement bas.

## Tactique 2 : Séparation des entités

**Description**: Séparation du système en plusieurs entités

**Justification**: Séparer le système de cette façon permet de limiter les dégâts en cas de faille de sécurité. Par exemple, si les informations bancaires des clients sont séparées de leur compte client principal, et que c’est le compte client principal qui se fait pirater, les informations bancaires ne sont pas à risque.

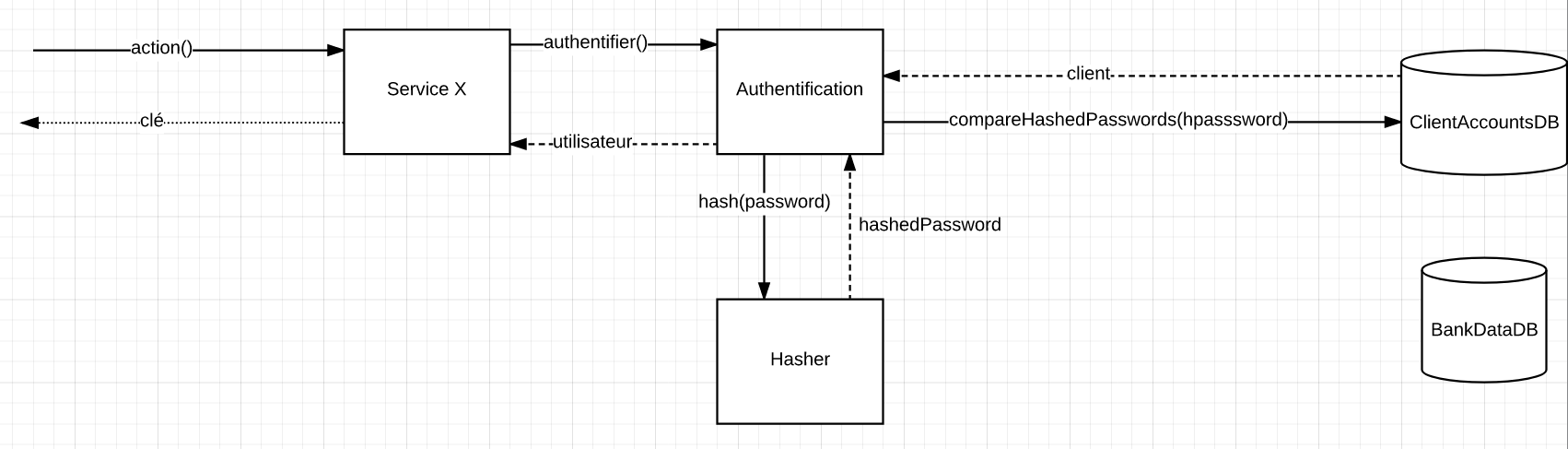
## Tactique 3 : Encryption des données

**Description**: Encrypter les données avec une clé inconnue par les acteurs non autorisés

**Justification**: En encryptant les données, nous nous assurons que même si les données sont dérobées, elles sont illisibles par les acteurs qui ne possèdent pas la clé. Ceci ne prévient pas les intrusions au système, mais limite les dégâts.

## Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende

Carré: Service ou module

Cylindre: Base de donnée

Flèches pleines: action

Flèches non-pleine: retour

### Texte de description du diagramme

Ce diagramme démontre les trois tactiques de notre stratégie. En premier lieu, chaque action faite à un service de l’application aura comme prérequis la présence d’un utilisateur authentifié. Afin d’authentifier un utilisateur, trois possibilités existent:par nom d’utilisateur et mot de passe, clé SSH, ou par clé. Une clé à usage limité est retourné après la première authentification. Ainsi, l’utilisateur ne se fera pas demander de s’authentifier alors qu’il vient de le faire, puisqu’il peut désormais utiliser la clé pour s’authentifier. Ensuite, pendant le processus d’authentification, un hash est utilisé. En effet, toutes les données dans nos bases de données seront hachées, tel que spécifié dans nos tactiques. Ainsi, lorsqu’un utilisateur entre un mot de passe, nous le hashons, puis nous le comparons au mot de passe hashé en base de donnée. Si les deux sont pareils, cela signifie que les version non-hashées de ces mots de passe sont également pareil. Enfin, nous avons extrait toutes les données bancaires afin de les enregistrer dans une autre base de donnée. Ainsi, en séparant les comptes clients des informations bancaires, nous séparons des données à usage fréquent des données transactionnelles. Il sera ainsi plus facile de protéger les données critiques, en les rendant accessibles seulement depuis les systèmes en ayant besoin.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Service X | Un des services ou module de l’application boursière. |
| Authentification | Système d’authentification. Prend un nom d’utilisateur et un mot de passe en entrée, et retourne une clé en cas de réussite. L’authentificateur peut prendre une clé en paramètre afin d’authentifier l’utilisateur, et retourner une nouvelle clé. |
| Hasher | Gère le hashage des données. Génère le seed de clé, et l’enregistre. |
| ClientAccountDB | Contient les informations reliées au login du client. Db consumée par le système d’authentification. |
| BankDataDB | Contient les informations bancaires des clients. DB consumée par le système transactionnel |

### Texte décrivant la relation entre les éléments et les tactiques

Le service X est la base de notre solution de sécurité. En effet, notre module d’authentification vient s’attacher à tous les services déjà présents. L’Authentification est au coeur de notre première tactique, alors que le hasher est au coeur de la troisième. Enfin, nous avons modélisé les bases de données dans l’app afin de démontrer la tactique 2, qui était de séparer les entitées. Ainsi, même si BankDataDB n’est pas consommé dans notre cas d’utilisation, on sait que celle-ci est consommée par le service transactionnel, tel que modélisé dans le document de modificabilité.

# Scénario : Covfefe, un hacker russe employé par Donald Trump, tente d’attaquer le système de manière à détourner des fonds pour la campagne présidentielle de 2020

|  |  |
| --- | --- |
| **objectifs d'affaires** | La bourse souhaite bloquer toute intrusion provenant de l'extérieur du système. |
| **Source** | Un attaqueur humain externe à l’organisation |
| **Stimulus** | Tentative non-autorisée de modification du comportement du système |
| **Artéfact** | Les données du système boursier |
| **Environnement** | Le système est opérationnel |
| **Réponse** | Bloquer l’accès de l’attaquant au système boursier.  Enregistrer toute modification du système ou tentative de modification durant l’attaque |
| **Mesure de la réponse** | Le temps de détecter l’attaque et contrer l’attaque. |
| **Questions** | 1. Comment l’attaquant a-t-il eu accès au système? 2. Quelles parties du système ont été compromises ? 3. Combien de temps s’est écoulé avant que l’attaque ne soit détectée ? |

## Tactique 1 : Verify Message Integrity

**Description**: Vérifie l’intégrité du message, à l’aide de checksum, pour s’assurer qu’il est reçu comme attendu par le système.

**Justification**: Permet de s’assurer que les requêtes n’ont pas été modifiées entre le moment où elles ont été envoyées par le client et le moment où elle ont été reçues par le serveur.

## Tactique 2 : Encrypt data

**Description**: Encrypter les données du système pour que seul le système puisse les lire.

**Justification**: Si les données sont encryptées, ces données seront inutiles pour l’attaquant, puisqu’il ne sera pas en mesure de les lire.

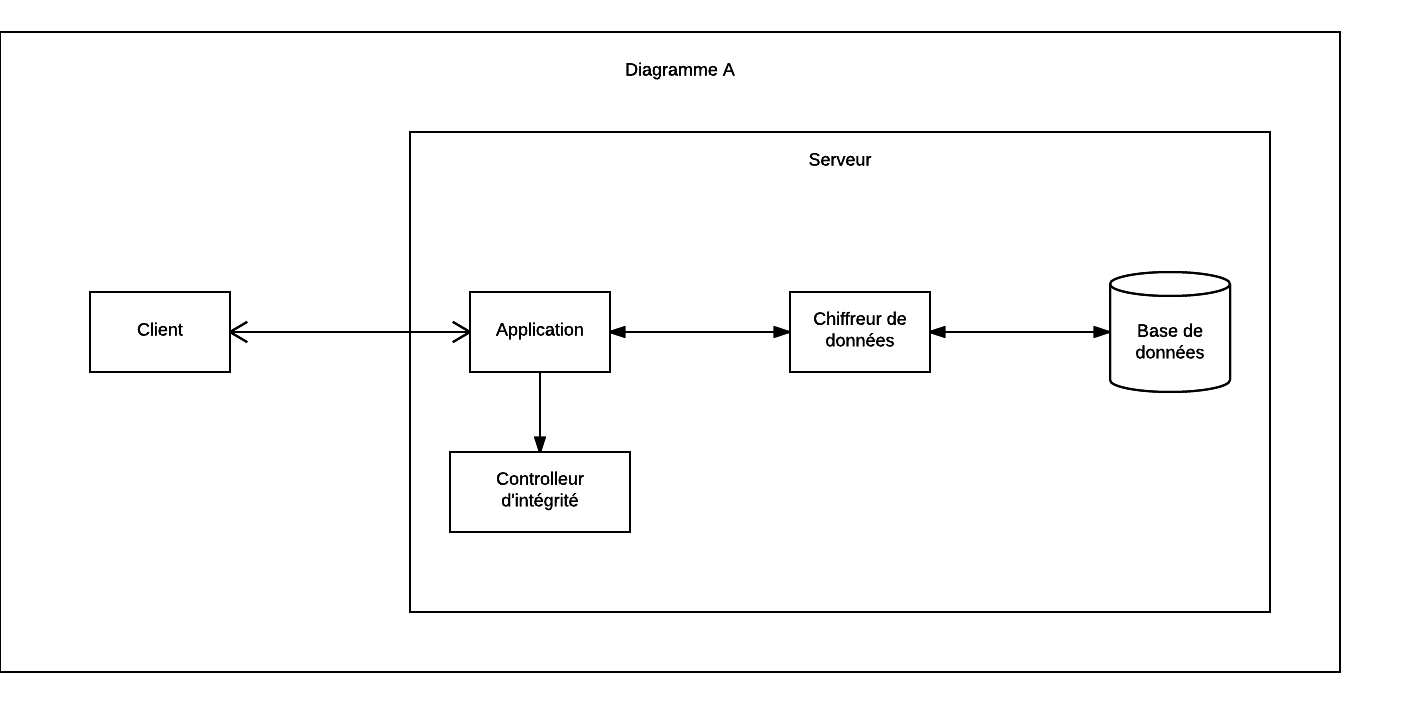
## Tactique 3 : Revoke Access

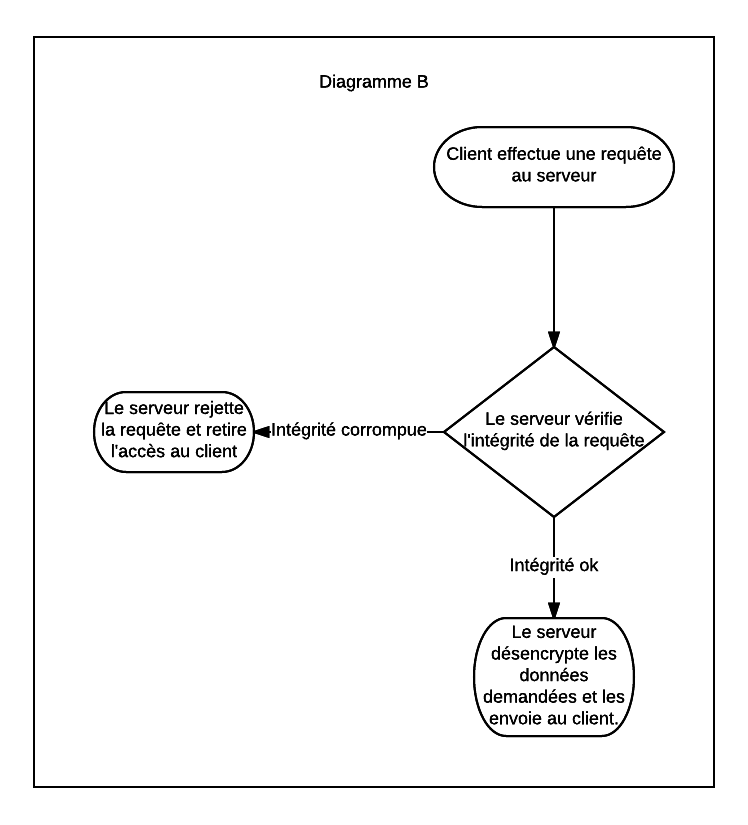
**Description**: Déconnecter tout usager n’ayant pas les accès requis ou retirer les accès à un utilisateur effectuant une opération non-désirée.

**Justification**: Lorsqu’une manoeuvre non-désirée est effectuée, l’utilisateur impliqué aura ses accès retirés afin d’évaluer si l’action est une attaque ou non. Aussi, cette tactique évite d’analyser chaque cas en temps réel, ce qui pourrait laisser assez de temps à un utilisateur frauduleux pour agir.

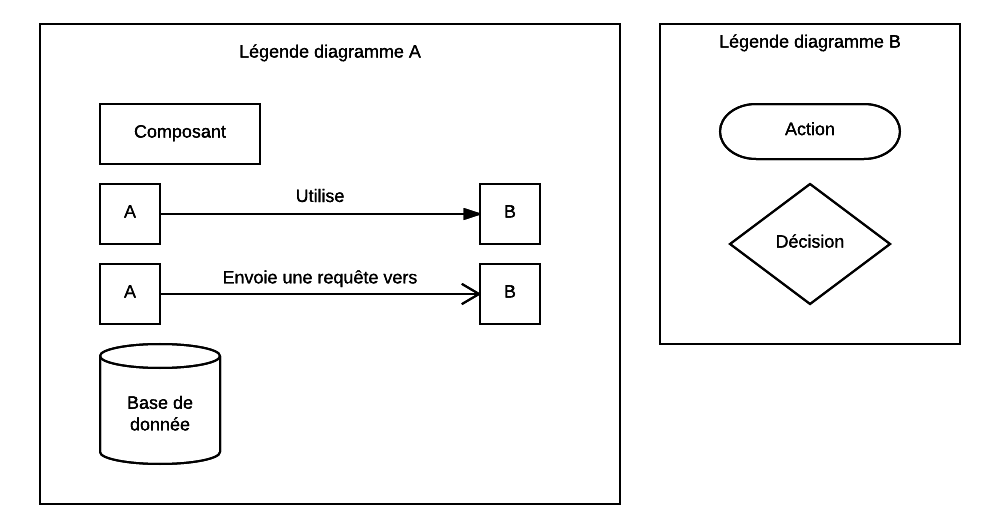
## Vue architecturale (à l’extérieur de la classe)

### Diagramme





### Légende



### Texte de description du diagramme

|  |  |
| --- | --- |
| **Diagramme A** |  |
| Le but du diagramme est de montrer comment une requête est vérifiée et accède aux données de manière sécuritaire. En utilisant un contrôleur d’intégrité, il est possible de vérifier que la requête n’est pas modifiée. De plus, le chiffreur de données permet de sécuriser les données avant qu’elles ne soient insérées dans la base de données. | |

|  |  |
| --- | --- |
| **Diagramme B** |  |
| Le but du diagramme est de montrer de quelle manière le serveur répond quand il reçoit une requête. Si l’intégrité d’une requête n’est pas validée, alors l’accès au serveur sera révoqué pour le client ayant envoyé la requête. En revanche, si la requête est intègre, le serveur répondra normalement à la requête. | |

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Client** | Représente un agent qui effectue une requête |
| **Serveur** | Contient les composants en rapport avec la sécurité de l’application |
| **Application** | Responsable de retourner des réponses à l’utilisateur en fonction du traitement effectué par ses différents composants |
| **Contrôleur d’intégrité** | Composant qui vérifie l’intégrité de la requête |
| **Chiffreur de données** | Composant qui chiffre les données avant leur envoi à la base de données |

### Texte décrivant la relation entre les éléments et les tactiques

Le contrôleur d’intégrité est directement lié à la tactique: Verify Integrity

Le chiffreur de données est directement lié à la tactique: Encrypt data

Le diagramme B démontre la logique associée à la révocation des accès à un client

# Scénario : Prévenir un utilisateur d’effectuer des ventes et achats non autorisés à travers les systèmes

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Empêcher un utilisateur non autorisé à effectuer des opérations non autorisées à travers l’application (systèmes FIX et SAIL). |
| **Source** | Un utilisateur externe |
| **Stimulus** | Effectue des achats et ventes sur l’application de certaines actions d’une compagnie |
| **Artéfact** | Données dans le système |
| **Environnement** | En mode opération normale |
| **Réponse** | L’utilisateur est identifié avec assurance, les données et services sont protégés d’un accès  non autorisé, Enregistrement des accès et modifications, avertissement d’un administrateur lors de l'échec de l’identification. |
| **Mesure de la réponse** | Combien de connexion non légitime a été repoussée avec succès et combien ont réussi à pénétrer le système?  À quel point les données sont vulnérables?  Combien de temps cela prend avant que nous détections un accès aux données non autorisé? |
| **Questions** | 1. Combien de fois ce genre d’attaque survient par intervalle de temps? 2. Quels sont les utilisateurs qui effectuent ce type d’activités non autorisées dans le système? 3. Quelle est la gravité des données corrompues? 4. Combien de temps cela va prendre pour récupérer de cette corruption? 5. Combien de temps se découle avant l'identification d’une attaque? |

## Tactique 1 : Détection de l’intégrité de la transaction

**Description**: Ajout d’une classe qui serait responsable de valider l’intégrité de la transaction.

**Justification**: Afin de bien pouvoir répondre à une attaque, il est important de commencer par détecter les opérations qui ne devraient pas être effectuées dans les systèmes (FIX et SAIL). Il s’agit de la première étape pour répondre à une attaque. Par conséquent, un vérificateur de transaction permettrait de valider si la transaction est parmi l’usuel de l’utilisateur. Si la transaction n’est pas considérée ‘usuelle’ alors celle-ci devrait être approuvée par un niveau supérieur tel qu’un appel téléphonique par exemple.

## Tactique 2 : Enregistrements des opérations effectuées dans les systèmes et restaurer au nécessaire

**Description**: Tactique qui consiste à enregistrer toutes les opérations effectuées dans les systèmes de l’application. L’ajout d’un audit de transaction et d’un enregistreur d’état permettra de faire cela.

**Justification**: Cette tactique est nécessaire afin de pouvoir réparer les attaques. Il est important d’enregistrer ce que les utilisateurs effectuent au cas qu’une intrusion soit détectée et qu’il faut annuler les données corrompues par cette intrusion.

## Tactique 3 : Désactivation des accès

**Description**: Tactique qui consiste à retirer les accès au système lors de la détection d’une attaque.

**Justification**: Suite à la détection d’une attaque, il est important de minimiser les dommages au système. C’est pour cela que dès la détection, l’accès est désactivé immédiatement et peut être réactivé seulement par un administrateur.

## Vue architecturale (à l’extérieur de la classe)

### Diagramme

### 

### Texte de description du diagramme

Dans le diagramme ci-dessus, il est possible de voir un utilisateur qui utilise le frontend. Dans un premier temps, il doit s’identifier, ce qui est géré par l’authentificateur. Une fois identifié, il peut faire des transactions. Chaque transaction est validée et auditée. Dans le cas où une transaction serait non valide, le valideur refuse la transaction et renvoie un message d’erreur aux mains qui lui révoque la connexion via l’authificateur. Dans le cas d'une détection d’attaque tardive, l’audit des transactions permet de revenir en arrière afin de rétablir l’état des données.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Nom** | **Description** |
| Frontend | Package représentent le système SAIL |
| Utilisateur | Personne qui veut faire une transaction |
| Backend | Package représentent le backend du système |
| Application | Package représentant toute l’application de la bourse. |
| Main | Package principal de l’application responsable de communiquer avec les autres packages du backend. Le package du plus bas niveau du backend. |
| Authentificateur | Package responsable de l’authentification et de la gestion des droits d’utilisations. |
| Transaction | Package responsable de gérer les transactions. |
| Audit des transactions | Package contenant tous les autres packages non énumérés et optimisés |
| Autres packages | Package contenant tous les autres packages non énumérés. |
| Valideur de transaction | Package responsable de la validation des transactions. |
| Réponse | Réponse du backend au frontend |
| Relation | Relation et action qui existent entre les composantes du diagramme |

### Texte décrivant la relation entre les éléments et les tactiques

La première tactique est implémentée via le valideur de transaction. Celui-ci valide que la transaction est plausible et qu’elle respecte des paramètres préétablis tels que le montant maximum, quantité maximum par jours, etc. Si elle respecte le tout, alors la transaction peut être procédé. Dans le cas où cette validation ne passe pas, un message d’erreur est retourné au package main. Ce message est lié directement avec la gravité du non-respect. Dépendant du message d’erreur, l'accès peut être révoqué temporairement par la main jusqu’à tant qu’un administrateur confirme la validité du tout et rétablisse l’accès. Ceci est donc notre troisième tactique. Finalement, toute transaction est auditée et enregistrée via le package d’audit de transaction. Si une attaque est détectée en retard, il est possible de faire un ‘rollback’ sur chacune des transactions afin de rétablir les données du système avant l'accès interdit. (Exemple : le maximum de transaction par journée était 5 pour un utilisateur et il vient tout juste d’atteindre le 5. Il est donc refusé par le valideur et son accès est révoqué. Par contre, en allant réactiver l'accès, l’administrateur se rend compte que c’était un fraudeur qui savait connecter au compte de l’utilisateur. Il est possible de revenir à l’état de compte avant les 5 transactions.) Voilà la deuxième tactique qui est l’enregistrement des transactions.

# Scénario : Un administrateur tente de modifier le montant d’une transaction complétée

|  |  |
| --- | --- |
| **Objectif d'affaires** | Empêcher la fraude (préserver l’intégrité des données) |
| **Source** | Administrateur Interne (employé) |
| **Stimulus** | Changement des valeurs des données |
| **Artéfact** | Base de données des transactions (Data Management & Reporting) |
| **Environnement** | Opération normale |
| **Réponse** | Les données ne sont pas manipulée sans autorisation, les données sont disponibles pour usage légitime, les tentatives d’accès aux données sont enregistrées |
| **Mesure de la réponse** | Résistance aux impacts  Combien de données sont vulnérables aux attaques  Temps écoulé avant la détection d’une attaque |
| **Questions** | 1. Quels sont les accès d’un employé normal? 2. Qui peut lire/écrire ces données? 3. Quelle est l’exposition du système pour ces données? 4. Y a-t-il des différents types d’accès (rôles)? |

## Tactique 1 : Limiter l’exposition

**Description**: Limiter la visibilité de la base de données et son accès à un minimum de ressources possible

**Justification**: Pour minimiser les possibles attaques ou modifications non-autorisées des données, il est nécessaire de restreindre la visibilité du système. En ayant un groupe limité de visibilité, nous pouvons déterminer des différents rôles et permissions dans le système pour augmenter la sécurité et l’intégrité des données.

## Tactique 2 : Révoquer les accès

**Description**: Bloquer un employé tentant de modifier frauduleusement les données

**Justification**: Cela permet de réduire le nombre d’attaques en limitant presque complètement un employé jusqu’à la correction de l’attaque. Les employés pour lesquels les droits ont été révoqués ne pourront pas s’adonner à plus d’activités illégales, ce qui augmente la sécurité du système.

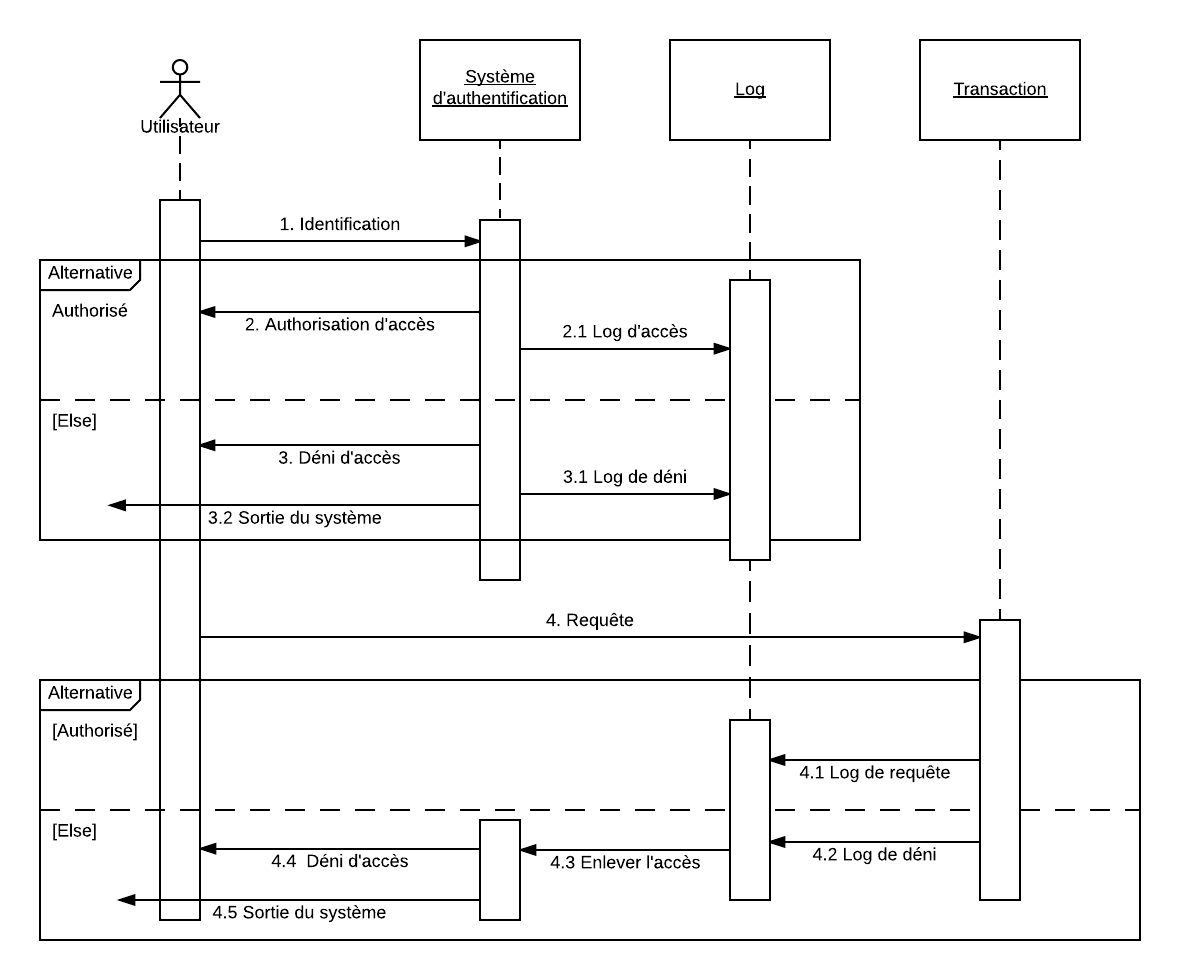
## Tactique 3 : Restauration des données

**Description**: Conserver un historique des transactions complétées

**Justification**: En conservant un historique des transactions complétées et de leurs données, il sera possible d’identifier les modifications illégitimes et de faire un « rollback » sur celles-ci pour récupérer les données véritables associées à ces transactions.

## Vue architecturale

### Diagramme



### Texte de description du diagramme

Le diagramme de séquence ci-haut décrit les actions à prendre par le système lorsqu’un utilisateur tente de se connecter et de faire une requête à la base de données des transactions. Premièrement, il faut que l’utilisateur s’authentifie. Dépendamment du succès de l’authentification, l’utilisateur peut effectuer une requête à la base de données. Peu importe l’action de l’utilisateur, le log sauvegarde toutes ses transactions. Si l’utilisateur ne réussit pas à s’identifier / s’authentifier au moment de faire une requête, le système enlève ses permissions et le déconnecte.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Élément du diagramme** | **Description** |
| Utilisateur (acteur) | Représente la personne ou ressource qui tente de se connecter au système. |
| Système d’authentification | Le système d'authentification est un portail sécurisé qui garde en mémoire (log) toutes les actions faites par les utilisateurs. |
| Log | Journalisation de toutes les actions faites par les utilisateurs à tout moment lors de leurs interactions avec la base de données des transactions et le système d’authentification. |
| Transaction | Représente la base de données des transactions du système boursier. |

### Texte décrivant la relation entre les éléments et les tactiques

Premièrement, “Limiter l’exposition”: En ayant un système d'authentification, nous limitons l’accès aux transactions à un groupe d’usagers possédant un niveau d’accès différents.

Deuxièmement, pour ce qui concerne la tactique “Révoquer les accès” :Si un utilisateur, n’ayant pas le niveau d’accès nécessaire, fait une requête de lecture ou de modification d’une transaction il sera sorti du système automatiquement et ses permissions seront révoquées. Toutes les requêtes seront journalisés.

Troisièmement, la tactique “Restauration des données” sera appliquée dans la possibilité où une transaction a été modifiée sans l'autorisation nécessaire pour le faire. Cette restauration pourra être faite à l’aide des backups de la base de données ainsi que du “log”:. Avec le TimpeStamp dans la journalisation, nous pouvons savoir l’état d’une transaction avant la modification.

# Scénario : Interception des données par un pirate lors des transactions.

|  |  |
| --- | --- |
| Objectifs d'affaires | S’assurer que les achats et ventes se fassent de façon sécuritaire (sans interception). |
| Source | Un pirate informatique |
| Stimulus | ~~Le pirate~~ tente de voler les données des transferts faits à partir d’une intrusion sur le système. |
| Artéfact | Les données utilisées lors de la transaction. |
| Environnement | Le système est en ligne, connecté au réseau et opérationnel. |
| Réponse | Lorsque le système reçoit plusieurs requêtes (de plusieurs utilisateurs) lors d’une même transaction, il faudrait notifier les administrateurs d’une anomalie et le système doit bloquer la transaction. Le système peut, aussi, enregistrer toutes les traces d’une transaction. |
| Mesure de la réponse | Le temps de détecter l’interception des données. |
| Questions | 1. Devrait-on ajouter un mot de passe lors de l’exécution d’une transaction ? 2. Est-ce que des informations confidentielles ont été volées ? 3. Est-ce que l’interception pourrait être évitée ? |

## Tactique 1 : Maintain Audit Trail

Description: Conserver une trace des actions d'achats et de ventes effectuées par les utilisateurs.

Justification: En conservant les audits, les achats et les ventes seront surveillés de plus près et il sera possible de déterminer les faiblesses utilisées par les attaquants.

## Tactique 2 : Identify Actors

Description: La tactique d’identification utilise des informations récupérées pour authentifier l’identité du client.

Justification: L’identification des acteurs du système boursier permettra de vérifier si la demande de transaction (achat/vente) a été réalisé par le compte client en droit.

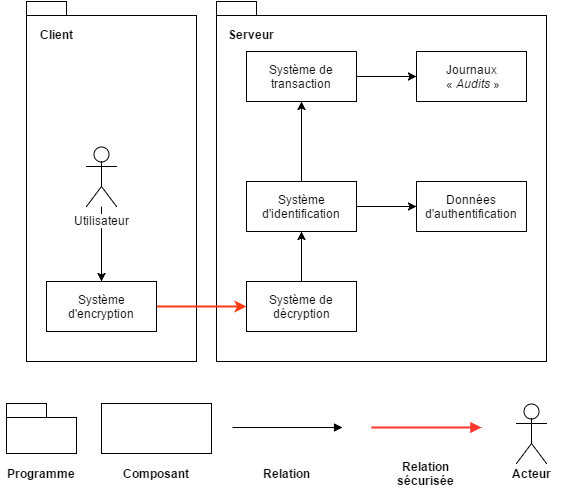
## Tactique 3 : Encrypt data

Description: L’utilisation de clefs privées ou d’encryption renforce l’intégrité des données transmises entre le client et le serveur.

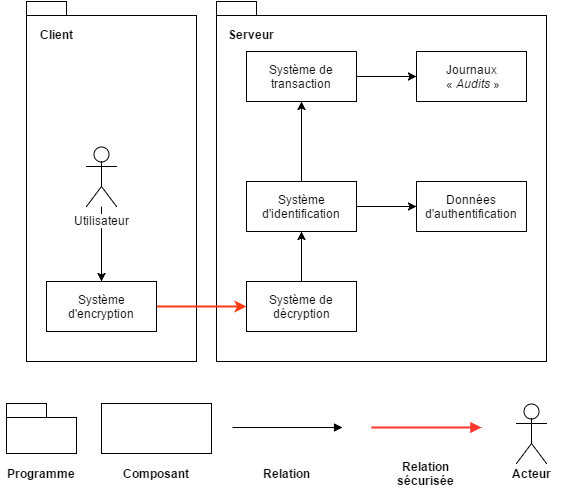
Justification: L’encryption des données, à défaut de performance, diminuera les attaques du type “Man in the middle”. En effet, ne sera pas possible aux attaquants de créer de fausses requêtes vers les clients ni vers les serveurs. De plus, ils ne seront pas capables de décrypter donc il sera impossible pour les attaquants d’interpréter les messages circulant sur le réseau.

## Vue architecturale

### Diagramme



### Légende



### Texte de description du diagramme

Le client utilisera un système d’encryption. Lorsque la transaction sera effectuée, le système du serveur pourra décrypter les données à son tour. Ce nouveau système permettra, lors de l'interception inhabituelle des données, de rendre les données illisibles (indécodable). Le système de décryptage sera différent pour chacun des utilisateurs. Celles-ci seront décryptées à partir des données d’authentification. Finalement, une fois les données décryptées, elles passeront dans le système de transaction; ce qui permettra d’effectuer la transaction voulue par le client.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Éléments** | **Responsabilités** |
| Système d’encryption | Crypte les données du client (au niveau client). |
| Système de décryptage | Décrypte les données lors de leurs arrivées au serveur. |
| Système d’identification | Identifie le client. |
| Système de transaction | Procède à la transaction avec les données. |
| Données d’authentification | Base de données permettant d’authentifier l’utilisateur. |
| Journaux audits | Sauvegarde les informations de la transaction. |

### Texte décrivant la relation entre les éléments et les tactiques

|  |  |
| --- | --- |
| **Tactique** | **Description** |
| Maintain audit trail | Les «journaux audits» permettront de sauvegarder toutes les traces créées par les transactions. |
| Identify actors | Le «système d’identification» récupère les données sur un utilisateur dans la bande de «données d’authentification» pour l’ identification du client pour s’assurer que le compte client est en droit. |
| Encrypt data | Le «Système d’encryption» et «Système de décryptage» sont directement lié à cette tactique. Ils cryptent et décryptent les données pour éviter qu’un utilisateur malveillant se mette entre le serveur et le client et intercepte les données d’une transaction. |

# Scénario : En dehors des heures de travail, l’attaquant va en profiter pour accéder aux données des clients de l'extérieur de l’organisation.

|  |  |
| --- | --- |
| **objectifs d'affaires** | Sécuriser les données de l’organisation |
| **Source** | Un utilisateur malveillant venant de l'extérieur. |
| **Stimulus** | Display, change or remove data |
| **Artéfact** | Data within the system |
| **Environnement** | Online |
| **Réponse** | Recording access or modification  Notifying appropriate entities when an apparent attack is occurring |
| **Mesure de la réponse** | 1 minutes après la brèche, le système doit être revenue dans un état opérable |
| **Questions** | 1. Est-ce la première attaque? 2. Une équipe d’urgence peut-elle être appelée dans les plus brefs délais? 3. Quelle données ont été touchées? 4. Doit on informer les clients de la brèche? 5. Pendant combien de temps le hacker a infiltré le réseau? |

## Tactique 1 :Inform actors

**Description**: Permet d’informer le personnel ou les opérateurs du système

**Justification**: Lors d’une attaque potentielle, le personnel ou les opérateurs du système seront notifiés lorsque le système aura détecter une attaque.

## Tactique 2 : Limit exposition

**Description**: Minimise les risques d’expositions du système.

**Justification**: Séparer les bases de données utilisateurs et transactions pour protéger les données sensibles comme les informations sur les clients.

## Tactique 3 : Detect Intrusion

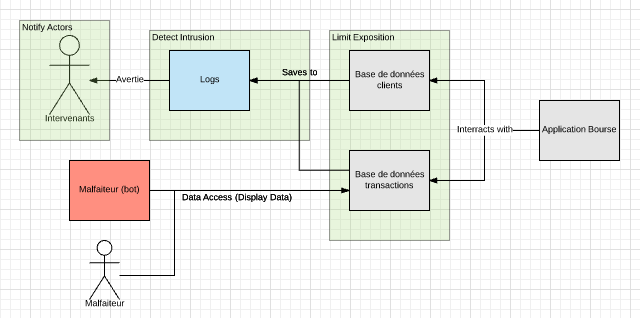
**Description**: Détecte les personnes qui ont été enregistrés au système

**Justification**: Permet de vérifier que les personnes enregistré au système sont bel et bien le personnel de l’organisation

## Vue architecturale (à l’extérieur de la classe)

### Diagramme

### 



### Légende

-Les flèches représentent les liens entre les éléments

-Le malfaiteur et le Malfaiteur bot, en rouge, sont des entités qui ont réussi à accéder aux données

-Les éléments gris sont des éléments physique (Une machine roulant l’instance de l’application les bases de données)

-Les encadrés vert représente les éléments qui sont sous une tactique

### Texte de description du diagramme

Les données dans les bases de données sont placées dans des endroits différents, soit dans une base de données qui contient les informations de l’utilisateur et une qui contient les données de transactions. Les utilisateurs réguliers font des transactions à l’aide de l’application. Les malfaiteurs accède à la base de données des transactions. Toutes les accès aux bases de données sont enregistrées dans les logs. Lorsqu’un accès non-autorisé est détecté, les acteurs principaux sont averties.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Tactique/Élément** | **Description** |
| Detect Intrusion | Tactique de détection d’intrusion |
| Notify Actors | Tactique de Notify Actors |
| Intervenants | Les personnes à avertir à l’aide de la tactique notify actor |
| Malfaiteur / Malfaiteur Bot | Des entités malfaisante qui ont accédé aux données |
| Logs | Historique de tous les accès aux données de la base de données |
| Base de données client | La base de données qui contient tous les infos des clients |
| Base de données transactions | Base de données qui contient les informations des transactions |
| Limit Exposition | La tactique de limitation d’exposition, effectué en ayant deux bases de données physique différentes |
| Application Bourse | Machine roulant l’application qui peut faire des transactions boursière |

### Texte décrivant la relation entre les éléments et les tactiques

Un malfaiteur réussit à accéder aux données de la base de données de l’application. Par contre, le fait d’avoir deux bases de données limite les expositions des données sensibles au malfaiteur. Tous les accès aux serveurs sont enregistrées dans les logs, alors une intrusion peut être détecté. Lorsque l’intrusion est détectée, les principaux intervenants sont appelé afin de juger des actions nécessaire à effectuer (si l’intrusion est retraçable, poursuite ou bien arrestation du malfaiteur)

# Scénario : Déconnecter un utilisateur lors d’une période d’inactivité de 4 minutes

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Empêcher un intrus d’accéder à un poste déjà connecté sur l’application lorsque la personne authentifier n’est pas à son bureau |
| **Source** | L’utilisateur est authentifié |
| **Stimulus** | Un temps de 4 minutes s’écoule |
| **Artéfact** | Session courante (implique les données et services du système) |
| **Environnement** | Le système est en ligne et sert le client authentifié |
| **Réponse** | Terminer la session |
| **Mesure de la réponse** | Combien de temps s’est écoulé avant la déconnexion  Combien de temps s’est écoulé avant la requête avec un token expiré |
| **Questions** | 1. Quel est le temps idéal avant la déconnexion ? 2. Quel est l’importance des données? 3. On redemande juste le mot de passe ou les informations de login au complet? 4. Que fait-on avec les utilisateurs ayant essayé de faire une requête avec un token expiré depuis longtemps? |

## Tactique 1 : Déconnexion automatique

**Description** : Après un certain délai d'inactivité de l’utilisateur, le système devra déconnecté ce dernier afin d’éviter qu’un autre humain puisse avoir accès à des données sensibles.

**Justification** : Puisqu’on est dans un système bancaire, les données de chaque utilisateurs et de l’entreprise sont sensible. L’utilisation d’un système de déconnection automatique permet à l’utilisateur d’être assuré que ses informations ne seront pas disponibles si un utilisateur a accès à son poste physique (après un certain temps d’inactivité).

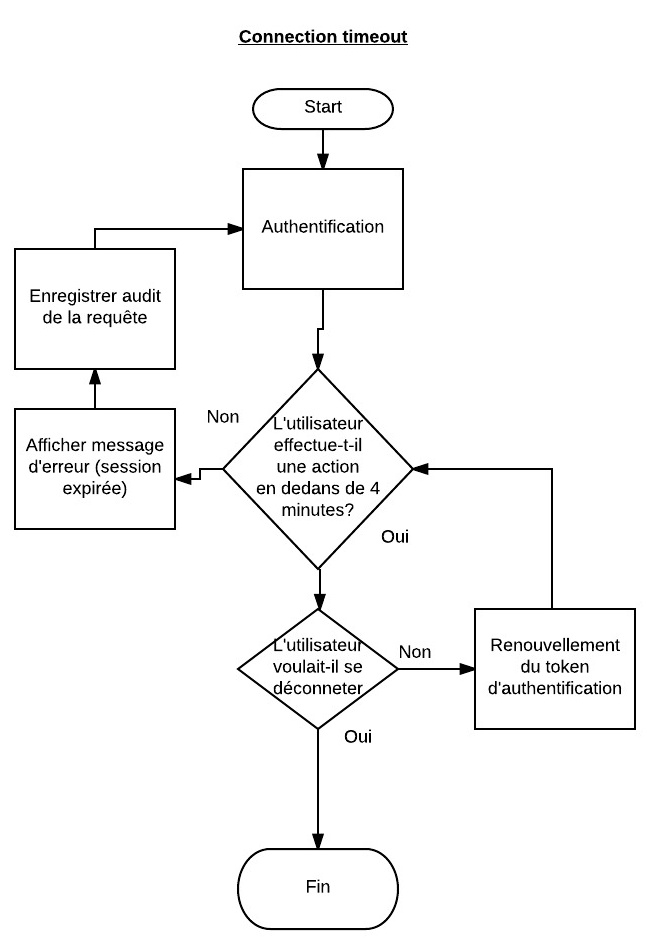
## Tactique 2 : Maintenir des traces de l’audit

**Description** : Chaque requête envoyée au serveur avec un token expiré est enregistrée.

**Justification** : Chaque enregistrement va aussi contenir le temps écoulé depuis l’expiration du token. Ainsi, il va être possible d’identifier qui laisse souvent sa session ouverte lorsqu’il quitte l’ordinateur.

Vue architecturale

## Diagramme



## Description du diagramme

Le ci-dessus démontre le déroulement de la vérification de l’expiration d’une session. Au départ, l’utilisateur s’authentifie. S’il n’effectue pas d’action au cours des 4 prochaines minutes, la session expire et il doit s’authentifier à nouveau. Lorsque l’utilisateur est déconnecté à cause de l’expiration de la session, l’événement est enregistré dans le système pour maintenir un historique. Ces informations peuvent être intéressantes au cours d’un audit de sécurité.

## Table d’éléments

|  |  |
| --- | --- |
| **Start/Fin** | Ce sont les points d’entrées du diagramme d’état. Ici, nous débutons lorsqu’un utilisateur essaie de s’authentifier et on termine lorsqu’il se déconnecte. |
| **Décision** | Plusieurs décisions sont prises afin d’appliquer des processus intelligents en fonction de ces décisions. Une décision se sépare généralement en 2 chemins (Oui/Non). |
| **Processus** | Ces processus sont actionnés en fonction des décisions prises. Plusieurs processus peuvent être appliqué l’un à la suite de l’autre. Dans notre cas, ce sont des processus permettant d’offrir une plus grande sécurité pour les utilisateurs. |

## Liens entre les vues et les tactiques

Après avoir été authentifié par le système, l’utilisateur sera déconnecté après une période d’inactivité de 4 minutes. Après une déconnexion, l’utilisateur doit saisir à nouveau toutes les informations nécessaires à sa connexion.

# Scénario 1 (Sécurité): Un développeur tente de se connecter au serveur de production (base de données + application principale)

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Aucun développeur n’a accès aux données ou à l’environnement de production. |
| **Source** | Développeur |
| **Stimulus** | Tente de se connecter au serveur de production |
| **Artéfact** | Serveur de production |
| **Environnement** | Online |
| **Réponse** | * Enregistrer les accès et les modifications * Enregistrer les tentatives d’accès à la ressource et aux services * Données et services sont protégés * Données et services ne peuvent être modifiés sans autorisation |
| **Mesure de la réponse** | * Portée d’accès du développeur (scope) * Temps de détection de la violation |
| **Questions** | 1. Qui a accès au serveur de production? 2. Qui a droit de modifier le serveur de production? |

**Tactique 1: Autoriser les acteurs**

**Description:**

Lorsque le développeur tente d’accéder à des informations sur le serveur de production, le système vérifie s’il possède les droits nécessaires et l'empêche d'accéder aux données dont il n’a pas accès.

**Justification**:

En permettant seulement les acteurs autorisés, on peut empêcher les acteurs non autorisés d’accéder au serveur de production.

**Tactique 2: Identifier les acteurs**

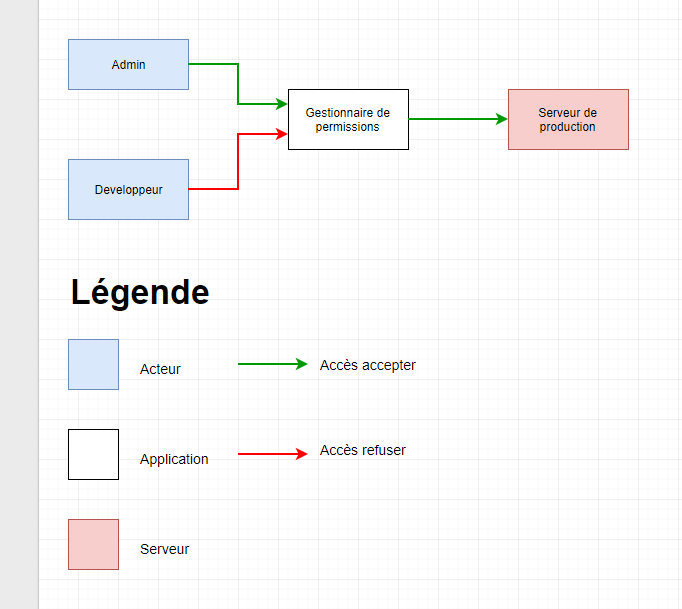
**Description:**

Le système identifie le rôle de l’utilisateur et lui applique les droits appropriés.

**Justification**:

Cette technique évite d’avoir des utilisateurs inconnus, ce qui permet de mieux gérer leurs permissions.

### Diagramme et légende



### Texte de description du diagramme

Le gestionnaire de permissions reçoit les demandes d’authentification des clients et leur assigne le rôle correspondant. Il utilise ces rôles pour gérer les permissions de chaque utilisateur lorsque ceux-ci tentent d’accéder aux données sur le serveur de production.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Admin | L’utilisateur qui est assigné le rôle d’admin. |
| Gestionnaire de permissions | La machine qui authentifie les utilisateurs et qui leur accorde les permissions appropriées. |
| Serveur de production | Le serveur de production . |
| Développeur | L’utilisateur qui est assigné au rôle de développeur. |
| Relation accès acceptée | Lorsque le gestionnaire de permission identifie et approuve l’acteur, celui-ci va lui donner accès aux serveurs de production. |
| Relation accès refusée | Lorsque le gestionnaire de permission identifie et désapprouve l'acteur, celui-ci ne va pas donner accès aux serveurs de production. |

### Texte décrivant la relation entre les éléments et les tactiques

Comme on peut voir dans le diagramme, le gestionnaire de permissions utilise les deux tactiques que nous avons choisies. En effet, le gestionnaire va identifier les acteurs et trouver leurs rôles afin d'autoriser seulement les acteurs qui possèdent les permissions requises pour avoir accès au serveur de production.

# Scénario : Un hacker essaye d’ajouter une transaction à son compte.

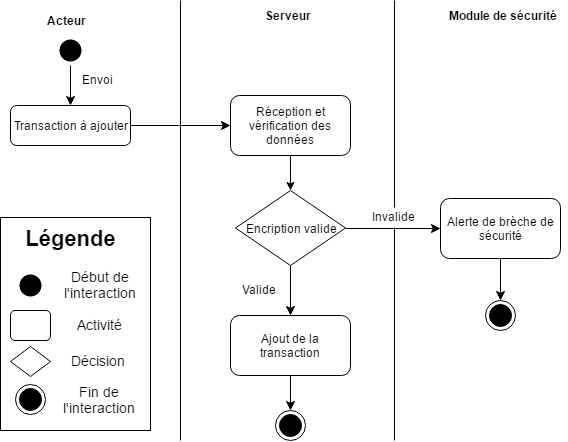
|  |  |
| --- | --- |
| **Objectifs d’affaires** | S’assurer que les transactions passent par le système officiel de création de transactions. |
| **Source** | Hacker |
| **Stimulus** | Tentatives non-autorisées d’ajout de transactions |
| **Artéfact** | Système de création de transactions |
| **Environnement** | Le système fonctionne normalement. |
| **Réponse** | Les transactions sont entièrement gérées par le système de création de transactions. |
| **Mesure de la réponse** | * D’autres parties du système pourraient être accessible par le hacker. * Si une transaction est ajoutée sans avoir passé via le système officiel, une alerte doit être lancée. |
| **Questions** | * Est-ce que le système peut reconnaître une attaque? * Combien de tentatives de connection permet le système de bourses? * Comment différencier une transaction légitime d’une transaction frauduleuse? |

## Tactique 1 : Encrypt Data

**Description**: Le système est le seul qui peut sécuriser correctement des données pour garder l'intégrité de celles-ci.

**Justification**: Cette tactique nous permet de confirmer que les communications sont bel et bien créées et lues par le système uniquement. Donc, le hacker ne peut pas ajouter une transaction sans passer par le système.

**Vue Architecturale**



### Texte de description du diagramme

Un acteur envoi une transaction à ajouter au serveur qui lui vérifie les données reçues. Ensuite, le serveur valide le chiffrement. Si celui-ci est valide, la transaction est ajoutée et on atteint la fin du parcours. Sinon, la transaction n’est pas valide et une alerte de brèche de sécurité est lancée dans le module de sécurité.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Acteur** | Entité voulant ajouter une transaction. |
| **Serveur** | Reçoit la demande et les données et valide le chiffrement. |
| **Module de sécurité** | Lance une alerte de brèche de sécurité si jamais l’encryption est invalide. |

### Texte décrivant la relation entre les éléments et les tactiques

La tactique choisie est celle du chiffrement des données pour s’assurer de garder l’intégrité de l’information reçue en plus de valider sa provenance. L’acteur est celui qui ajoute une transaction mais celui-ci n’a pas nécessairement la bonne façon de chiffrer les données. Le serveur s’assure que le chiffrement a correctement été fait. Donc, si le chiffrement n’est pas celui attendu, le système est certain que les données ne sont pas légitimes.