  
**Fait par :** Kamguia Kouam Eric Stephane

Devoir I:

1. Explication de la problématique des IDS
2. Qu’est-ce qu’un système de détection d’intrusion

Réponse :

Un système de détection d'intrusion (IDS) est un outil logiciel ou matériel utilisé pour surveiller le trafic réseau et les activités suspectes sur des hôtes afin d'identifier les tentatives d'intrusion ou les comportements anormaux.

1. Discutons des approches classiques pour détecter les intrusions (méthodes basées sur les signatures et les anomalies).

Réponse :

Méthodes basées sur les signatures : le principe de cette méthode repose sur le fait que le système fait une comparaison entre les activités en cours et une base de signatures d’attaques connues. Lorsqu’une similitude est trouvée le système génère une alerte. Les principaux avantages de cette méthode est qu’il a un faible taux de faux positif et il est d’une précision élevée. Comme inconvénient, cette méthode nécessite des mises à jour régulière et ne peut détecter des attaques de zéro-Day.

Méthodes basées sur les anomalies : le principe ici est que le système fait un profil de base basé sur l'analyse du trafic réseau, des performances système, le lieu d’activité régulier ou plus généralisatrice des paramètres d’habitudes. Toute écart marqué par rapport à ce modèle est considérée comme suspecte. Cette méthode est très adaptative et peu mieux que la méthode précédente, détecter des attaques zéro-Day. Par contre elle génère beaucoup de faix positif, très complexe et demande beaucoup de ressources (puissance de calcul).

1. Analysons les limites des méthodes traditionnelles, notamment leur incapacité à détecter les nouvelles menaces.

Réponse :

Dans la méthode de détection par signatures seul les signatures d’attaques préalablement correctement enregistre dans l’IDS peuvent être détecter. Une fois que les caractéristiques de cette même attaque est légèrement modifiée sa signature numérique change ce qui la rend indétectable par un IDS utilisant la méthode de signature. Dans ce cas de figure on parle d’évasion de signature qui pourrais inclure des techniques comme l’obscurcissement des malwares, le polymorphisme, la fragmentation des paquets ou requetés à l’instar d’une

reconnaissance effectuée par un cyberattaquant en ne complétant pas le three-way handshake du protocole TCP.

Dans le déploiement de nouveau services ou nouvelle application dans un systèmes en présence d’IDS les attaques zéro-Day représente une très grande menace car cette menace qui se transformerais plus tard en vulnérabilités rend l’outils IDS inefficace. Par conséquent sans des mises à jour régulière IDS serais obsolète d’où sa limite dans le temp. Ce qui traduit la réactivité dans l’utilisation des IDS.

Dans la méthode de détection d’anomalies un comportement dit normale est établi par l’IDS ; Donc tout ce qui n’entre pas dans les habitudes de trafic réseaux ou de comportement d’hôte est anormale. Par contre cette méthode est moins évolutive c’est-à-dire avec l’avènement des nouvelles technologies tel que le cloud computing, les IoT et des SDN, les réseaux sont devenus plus distribués et dynamique. Avec cette méthode elle est très statique et dépend plus de l’humain et de la configuration initiale que de l’adaptativité a l’environnement changeant ; Et même définir un comportement normal est une chose complexe car il faut construire un modelé d’une grande précision et généralisatrice dans le cas contraire les nouvelles attaques dans un environnement diffèrent ou même le même ne seront pas détecte.

1. L'apport du Machine Learning (ML) pour les IDS
2. Expliquez en quoi les algorithmes de machine learning peuvent apporter une solution aux limitations des techniques classiques.

Réponse :

Les algorithmes de machine learning apporte du dynamisme, de l’adaptabilités ce que n’avait pas les techniques classiques. Elle apporte un aspect prédictif dans la détection d’anomalies ou de comportement malveillant. Une fois le modèle génère il est capable de généraliser et distinguer les attaques des non-attaques ; Elle ne nécessite pas des mises a jours réguliers, aussi, peut détecter des attaques de zéros-Day.

1. Décrivez les avantages du ML dans la détection d'intrusions, notamment la capacité à généraliser et à s'adapter à de nouveaux types d'attaques

Réponse :

**L’apprentissage machine** dans la détection d'intrusions présente plusieurs avantages par rapport aux méthodes traditionnelles, notamment :

**- Capacité à généraliser et à détecter les attaques zero-day**

**- Adaptabilité aux environnements changeants**

**- Détection des comportements anormaux**

- Réduction du taux de faux positifs et de faux négatifs

**- Capacité à détecter des attaques multi-étapes et persistantes**

- Automatisation et réduction de l'intervention humaine

3. Présentation du jeu de donnée NSL-KDD :

a) Mon jeu de données est constitue sous forme de matrix ou tableau de [125973, 43] dont 24 attributs de type int, 15 de types float et 4 de type object. Les differents caracteristique qui sont au nombre de 43 sont :

Duration : durée de la connexion en secondes.

Protocol\_type : type de protocole utilisé dans la connexion

Service : service réseau utilisé pour la connexion

flag : indicateur de statut de la connexion

src\_bytes : nombre d'octets envoyés de la source (le client) vers la destination

dst\_bytes : nombre d'octets envoyés de la destination (le serveur) vers la source (le

client).

Land : indicateur du fait que la connexion ai la mêmes adresses IP de source et de destination et les mêmes numéros de port (0 ou 1).

Wrong\_fragment : nombre de fragments de paquets incorrects.

Urgent : nombre de paquets urgents dans la connexion.

Hot : nombre d'indicateurs « chauds » dans une connexion (par exemple, des commandes suspectes).

num\_failed\_logins: nombre d’ouverture de sessions échoué.

logged\_in: ouverture de sessions reussite

num\_compromised: nombre de sessions compromises

root\_shell: Valeur binaire (0 ou 1), où 1 signifie qu'un shell root a été obtenu.

su\_attempted: Valeur binaire (0 ou 1), où 1 signifie qu'une tentative d'utilisation de la commande "su" a eu lieu

num\_root: Nombre de requêtes d'accès root.

num\_file\_creations: Nombre de fichier creer

num\_shells: Nombre de shells lancés dans la connexion.

num\_access\_files: nombre de fichier consulte

num\_outbound\_cmds: Nombre de commandes outbound dans la connexion (utile pour le FTP).

is\_host\_login: ouverture de sessions par hote

is\_guest\_login: ouverture de sessions par l’invite

count: Nombre de connexions à la même adresse de destination dans une fenêtre de 2 secondes.

srv\_count: Nombre de connexions au même service dans une fenêtre de 2 secondes.

serror\_rate: Pourcentage de connexions ayant des erreurs de type SYN.

srv\_serror\_rate: Pourcentage de connexions au même service ayant des erreurs de type SYN.

rerror\_rate: Pourcentage de connexions ayant des erreurs de type REJ

srv\_rerror\_rate: Pourcentage de connexions au même service ayant des erreurs de type REJ.

same\_srv\_rate: Pourcentage de connexions au même service.

diff\_srv\_rate: Pourcentage de connexions à des services différents.

srv\_diff\_host\_rate: Pourcentage de connexions au même service mais à des hôtes différents.

dst\_host\_count : Nombre de connexions à la même adresse de destination.

dst\_host\_srv\_count : nombre de connexions au même service avec la même destination au cours des 2 dernières secondes.

dst\_host\_same\_srv\_rate : proportion de connexions au même service.

dst\_host\_diff\_srv\_rate : proportion de connexions à différents services

dst\_host\_same\_src\_port\_rate : taux de connexions au même port source

dst\_host\_srv\_diff\_host\_rate : taux de connexions à différents hôtes

dst\_host\_serror\_rate : taux de connexions au même service avec des erreurs «SYN»

dst\_host\_srv\_serror\_rate : taux de connexions à l’hôte avec des erreurs « REJ » dans la connexion.

dst\_host\_rerror\_rate : taux de connexions à l’hôte avec des erreurs « REJ » dans la

connexion.

dst\_host\_srv\_rerror\_rate : taux de connexions au même service avec des erreurs « REJ ».

outcome : indique si la connexion est normale ou s’il s’agit d’une attaque.

5. Application des algorithmes

**Régression logistique**

Est un modèle utilisé pour la classification binaire (0 ou 1). Elle prédit la probabilité qu'un exemple appartienne à une classe donnée. Contrairement à la régression linéaire qui produit des valeurs continues, la régression logistique applique une fonction sigmoïde (ou logistique) à une combinaison linéaire des variables d'entrée pour obtenir un résultat compris entre 0 et 1, représentant la probabilité d'une classe. Les paramètres utilisés (**random\_state=0,** penalty='l2', solver='lbfgs', **C=1.0,** max\_iter=100, multi\_class='auto'

### ****Arbre de décision****

Est un modèle de classification ou de régression qui divise les données en segments basés sur des règles décisionnelles. Chaque nœud de l'arbre pose une question concernant une caractéristique des données, et selon la réponse (vraie ou fausse), les données sont divisées en sous-groupes. Les paramètres utilisés sont : criterion='entropy', max\_depth=5, min\_samples\_split=4, random\_state=0

### ****Forêt aléatoire (Random Forest)****

Est un ensemble d'arbres de décision. Elle combine plusieurs arbres indépendants (souvent des centaines ou milliers) pour obtenir un modèle plus robuste et précis. Chaque arbre est construit sur un sous-échantillon aléatoire des données avec des variables sélectionnées aléatoirement. Les paramètres utilisés sont : n\_estimators=200, max\_depth=10, random\_state=0, min\_samples\_split=5, max\_features='sqrt'

### ****Naive Bayes****

Est un algorithme de classification probabiliste basé sur le théorème de Bayes. Il est dit "naïf" parce qu'il suppose que toutes les caractéristiques sont indépendantes les unes des autres (ce qui n'est souvent pas vrai dans la réalité). Les paramètres utilisés sont : 0.6, 0.4], var\_smoothing=1e-5.

**Comparons les algorithmes** : L’algorithme Logique régression a un f1-score = 0.9780 avec dans la matrix de confusion la diagonale à minimiser a un totale de 645. Par contre l’algorithme Arbre de décision a un f1-score = 0.9982 avec un total de FP + FN = 52. Dans l’algorithme foret aléatoire j’ai un f1-score = 09993 et la diagonale a minimisé à un totale 21 ; par contre quand je ne normalise pas mes données, la Foret aléatoire a un f1-score = 0.9997 avec FP + FN = 10. Enfin l’algorithme Naive Bayes a un f1-score = 0.9041 et dans la matrix de confusion, la diagonale a minimisé à un totale de 2810.

Au regard de mes résultats obtenu et des différents prétraitements que j’ai faite sur mes données lors de ma manipulation et du code python que je soumets l’algorithme adapte pour cette tâche est le Foret aléatoire car elle a un score très élevé et dans sa matrice de confusion présente la diagonale à minimiser la plus basse. Cela est même meilleur lorsque mon jeu de données n’est pas normalisé.

Comme amélioration sur les modelés plus précisément dans Logistique de régression je suis quitte d’une force de régulation initialement a 0.5 a c= 1.0. Dans le cas de Foret aléatoire je suis quitte d’une profondeur maximale des arbres de 100 a max\_dept=1000. Dans arbre de décision je suis quitte d’une profondeur de l’arbre de 10 a max\_dept=100. Enfin dans Naive Bayes mon paramètre var\_smoothing avait une valeur de 1e-8 je l’ai manipulé jusqu’à trouver un juste milieu pour la minimalisation de la diagonale des FP et FN dont la valeur trouve adéquate est de 1e-5 .