Détection d'Anomalies de Connexion par Graphes Neuronaux (GAE)

Documentation User

EL GOTE Ismail-Ayoub

20 juin 2025

Résumé

Ce projet utilise un **Graph Auto-Encoder (GAE)**, un modèle de graphe neuronal, pour détecter des comportements de connexion anormaux et potentiellement malveillants à partir de logs d'événements. L'objectif est d'identifier des schémas complexes qui échappent aux règles de détection classiques, comme les attaques par *credential stuffing*, la force brute, l'utilisation de comptes volés ou la création de comptes frauduleux.

Table des matières

1 Fonctionnalités **2**

2 Comment ça marche ? **2**

2.1 `pretraitement.ipynb` 2

2.2 `construcion_graphe.ipynb` 2

2.3 `gae_model.ipynb` 2

3 Comment l'utiliser ? **3**

1 Fonctionnalités

- **Analyse Comportementale** : Le modèle ne se contente pas de regarder les échecs de connexion. Il analyse des caractéristiques comportementales fines :
  - **Fréquence de connexion** : Détecte les actions trop rapides pour un humain.
  - **Diversité des appareils** : Repère les comptes utilisés depuis un nombre anormal d'appareils ou de navigateurs.
  - **Comportement géographique** : Identifie les connexions depuis des lieux inhabituels pour un utilisateur.
- **Détection non supervisée** : Le modèle apprend la « normalité » à partir des données elles-mêmes, sans avoir besoin d'un jeu de données préalablement étiqueté comme « malveillant ».
- **Priorisation des menaces** : Chaque anomalie se voit attribuer un score, permettant aux analystes de se concentrer sur les menaces les plus critiques.

2 Comment ça marche ?

Le projet est divisé en trois notebooks Jupyter qui doivent être exécutés dans l'ordre.

2.1 `pretraitement.ipynb`

Ce notebook prépare les données brutes.

- **Input** : `logs/user_events_with_geoip_25k.csv` (logs bruts à remplacer avec le dataset entier. Ici, nous n'avons utilisé que 25000 lignes par souci de puissance de calcul).
- **Actions** :
  1. Nettoie et formate les logs.
  2. Filtre les événements pour ne garder que les tentatives de connexion.
  3. **Crée les caractéristiques comportementales** : Calcule la fréquence de connexion et la diversité des « user-agents » pour chaque IP et chaque utilisateur.
- **Output** : `logs/logs_events_clean.csv` (un fichier de logs propre et enrichi).

2.2 `construcion_graphe.ipynb`

Ce notebook transforme les logs en une structure de graphe que le modèle peut comprendre.

- **Input** : `logs/logs_events_clean.csv`.
- **Actions** :
  1. Crée un graphe bipartite où les nœuds sont des **IPs** et des **utilisateurs**.
  2. Une arête est créée entre une IP et un utilisateur s'il y a eu une tentative de connexion.
  3. Associe à chaque nœud les caractéristiques calculées à l'étape précédente (fréquence, diversité, infos géographiques, etc.).
- **Output** :
  - `construction/credential_stuffing_graph_v4.pt` : Le graphe avec toutes ses caractéristiques, prêt pour le modèle.
  - `construction/node_mapping_v4.pt` : Un dictionnaire pour faire le lien entre les nœuds du graphe et leur véritable identifiant (IP ou utilisateur).

2.3 `gae_model.ipynb`

C'est le cœur du projet : l'entraînement du modèle et la détection d'anomalies.

- **Input** : Les fichiers de graphe créés à l'étape 2.

— **Actions** :

1. Charge le graphe de connexions.
2. Entraîne le modèle GAE à « reconstruire » le graphe. Le modèle apprend ainsi à quoi ressemble une connexion « normale ».
3. Calcule un **score d'anomalie** pour chaque nœud en se basant sur l'erreur de reconstruction. Une erreur élevée signifie que le nœud se comporte de manière inattendue.
4. Visualise les résultats et sauvegarde les menaces les plus importantes.

— **Output** :

— `results_v4_behavioral/` : Un dossier contenant :

— `best_model_gae_improved.pt` : Le modèle entraîné.

— `model_comparison.csv` : Les performances du modèle.

— `top_anomalies.csv` : La liste des 100 IPs les plus suspectes, prêtes à être analysées.

3 Comment l'utiliser ?

1. **Installation** : Assurez-vous que toutes les bibliothèques listées dans `requirements.txt` sont installées.

```
pip install -r requirements.txt

```

Terminal

2. **Exécution** : Ouvrez et exécutez les notebooks dans l'ordre suivant :
   (a) `pretraitement.ipynb`
   (b) `construcion_graphe.ipynb`
   (c) `gae_model.ipynb`
3. **Analyse** : Consultez le fichier `results_v4_behavioral/top_anomalies.csv` pour voir les adresses IP les plus suspectes identifiées par le modèle.