

Simulation et préparation de traces d'un réseau domestique

Auteur : Henri BIKOURI

<https://henribikouri.github.io/>

Examineur : Pr Kevin JIOKENG

<https://kjiokeng.github.io/>

INTRODUCTION

Problématique et Objectifs

Dans le contexte actuel de prolifération des objets connectés, il est impératif de pouvoir caractériser et gérer les flux de données hétérogènes qui transitent par le Point d'Accès Wi-Fi. La coexistence de trafics aussi variés que le streaming vidéo régulier et les communications de capteurs à faible débit impose une pression significative sur les ressources du canal 802.11ac.

Conformément au sujet proposer, « **Simulation et préparation de traces d'un réseau domestique** », mes **objectifs** étaient doubles, à savoir :

1. **Modéliser et Simuler** un scénario de maison connectée en 802.11ac avec N équipements de K types différents.
2. **Préparer un jeu de données (dataset)** à partir des traces collectées pour l'entraînement d'un modèle de Machine Learning de classification des applications au niveau du point d'accès.

Mon approche combine l'ingénierie réseau fine avec la science des données, démontrant ma maîtrise de la chaîne allant de la conception du protocole à l'analyse avancée.

Outils et Justification

- 👉 **Outil de Simulation : NS-3 (Network Simulator 3).** Exiger dans les consignes du travail. Implémenter avec le standard wifi 802.11ac (que j'ai respecté) avec précision, ce qui est essentiel pour capturer les caractéristiques fines du trafic (comme le Délai d'Inter-Arrivée des paquets - IAT) nécessaires à une classification ML efficace.
- 👉 **Outils d'Analyse :** J'ai implémenté la chaîne d'analyse en Python, utilisant **Scapy** pour la manipulation des traces PCAP et d'autres bibliothèques pythons Pandas pour la structuration du jeu de données.

I. Modélisation et Conception du Scénario (NS-3)

J'ai conçu le scénario pour répondre aux exigences de l'énoncé, notamment en maximisant la variabilité introduite dans les données et dans les analyses.

I.1 Justification de mes Choix N et K

L'énoncé stipulait que "Plus N et K sont grands, plus il y a de la variabilité, et mieux c'est". Ma stratégie a été de maximiser la complexité du scénario pour produire un jeu de données riche et réaliste, captivant ainsi l'intérêt par la profondeur de la modélisation.

- **Mon Choix du Nombre de Types d'Applications (K=10) :** J'ai choisi **K=10** types d'applications distinctes. Cette diversité est l'élément clé qui garantit que le futur modèle ML devra identifier des signatures très fines (régularité, sporadicité, agressivité) au-delà d'une simple distinction UDP/TCP, augmentant ainsi la valeur de l'analyse.
- **Mon Choix du Nombre Total d'Équipements (N=32) :** J'ai défini **32 équipements** sources actifs. Ce nombre génère une forte concurrence et de la congestion au niveau du Point d'Accès

(AP), saturant le canal Wi-Fi 802.11ac. Ceci garantit que les traces PCAP contiennent des événements réels (collisions, backoffs, délais) essentiels pour l'extraction de caractéristiques ML robustes.

Tableau 1: tableau récapitulatif des choix

Type (k)	Application	Nbre de Nœuds	Plage d'Adresses IP	Protocole (Port)
1	Caméra Vidéo	5	10.1.1.2 à 10.1.1.6	UDP (Port 9001)
2	Capteur Temp.	10	10.1.1.7 à 10.1.1.16	TCP (Port 9002)
3	Assistant Vocal	3	10.1.1.17 à 10.1.1.19	TCP (Port 9003)
4	Téléchargement	2	10.1.1.20 à 10.1.1.21	TCP (Port 9004)
5	VoIP	4	10.1.1.22 à 10.1.1.25	UDP (Port 9005)
				UDP (Port 9006)
6	Domotique	4	10.1.1.26 à 10.1.1.29	UDP (Port 9007)
7	Streaming Mus.	1	10.1.1.30	TCP (Port 9008)
8	Sonnette	1	10.1.1.31	TCP (Port 9009)
9	MàJ Firmware	1	10.1.1.32	TCP (Port 9010)
10	Monitoring	1	10.1.1.33	UDP (Port 9011)
		Total N = 32		

La tableau 1 suivant détail mieux mes choix :

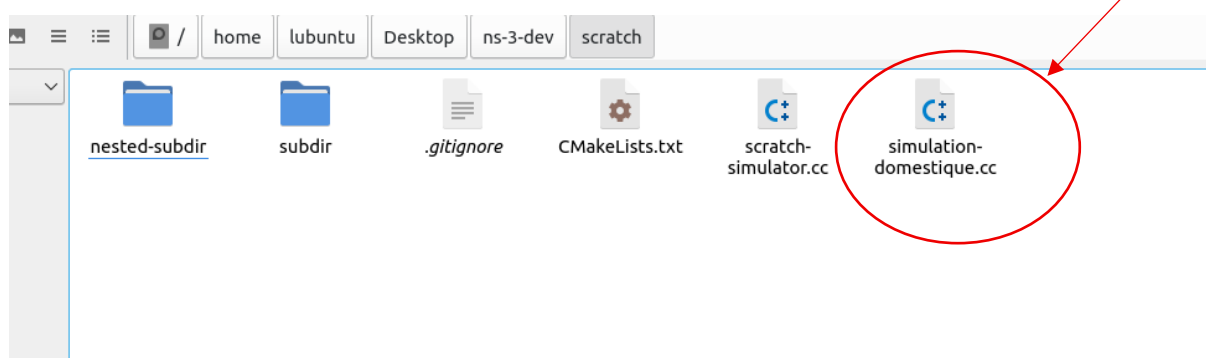
Et chaque application est associée à un serveur distant, soit 10 **Serveurs** d'adresses : **10.2.x.2** (x allant de 1 à 10 pour les 10 serveurs)

NB : D'après le tableau précédent vous pouvez voir que j'ai géré les deux cas de trafic VoIP (Montant et Descendant)

I.2 Simulation du réseau

Après avoir installé ns3 sur le site officiel : <https://www.nsnam.org/> et le configurer notamment avec la commande : `./ns3 configure --enable-examples --enable-tests` et `./ns3 build [1]`

Je me dirige vers le dossier `scratch` du dépôt ns3 puis j'ai créé mon fichier `simulation-domestique.cc`



C'est le fichier au cœur de mon travail (une explication en profondeur dans la demo), permet la simulation, ce qui a été configuré :

- 1- Met en œuvre la topologie 802.11ac avec **N=32 et K=10**.
- 2- Génère des traces **PCAP** (pour le Machine Learning).
- 3- Calcule des métriques de **QoS/Performance** (Débit, Délai, Perte) via FlowMonitor.

J'ai également configuré quelques options applicables au lancement de la topologie, par exemple :

- 1 **--duration=<seconds>** : durée de la simulation (par défaut 600 soit 10min)
- 2 **--enableFlowMonitor=<true/false>** : activer FlowMonitor (par défaut désactivé), pour recoller les métriques
- 3 **--enablePcap=<true/false>** : activer/désactiver la capture PCAP (désactivée par défaut), car groupant en ressource et exige de l'espace, même si je l'ai optimisé plusieurs fois.

Aperçu après lancement : pour cette durée Test de 300s. (mais dure en réalité environs 45min)

```

Lubuntu@Lubuntu22-virtualbox:~/Desktop/ns-3-dev$ ./ns3 run scratch/simulation-domestique -- --duration=300 --enableFlowMonitor=true --enablePcap=true --enableCsv=true --csvOutput=metrics_test.csv --flowOutput=traces de simulation
test fm.xml
J'impose le standard Wi-Fi : 802.11ac
Type de WifiNetDevice AP : ns3::WifiNetDevice
Type Phy AP : ns3::YansWifiPhy
Standard Phy AP configuré : 802.11ac
Bande Phy AP : 5GHz
RemoteStationManager AP : ns3::MinstrelHtWifiManager
Type de WifiNetDevice STA : ns3::WifiNetDevice
Type Phy STA : ns3::YansWifiPhy
Standard Phy STA configuré : 802.11ac
Bande Phy STA : 5GHz
RemoteStationManager STA : ns3::MinstrelHtWifiManager
Adresses assignées pour les serveurs et clients :
Serveur 0 -> 10.2.1.2
Serveur 1 -> 10.2.2.2
Serveur 2 -> 10.2.3.2
Serveur 3 -> 10.2.4.2
Serveur 4 -> 10.2.5.2
Serveur 5 -> 10.2.6.2
Serveur 6 -> 10.2.7.2
Serveur 7 -> 10.2.8.2
Serveur 8 -> 10.2.9.2
Serveur 9 -> 10.2.10.2
Client 0 -> 10.1.1.2
Client 1 -> 10.1.1.3
Client 2 -> 10.1.1.4
Client 3 -> 10.1.1.5
Client 4 -> 10.1.1.6
Client 5 -> 10.1.1.7
Client 6 -> 10.1.1.8
Client 7 -> 10.1.1.9
Client 8 -> 10.1.1.10
Client 9 -> 10.1.1.11

```

La suite également attribuer à chaque nœud (Serveurs et Equipements)

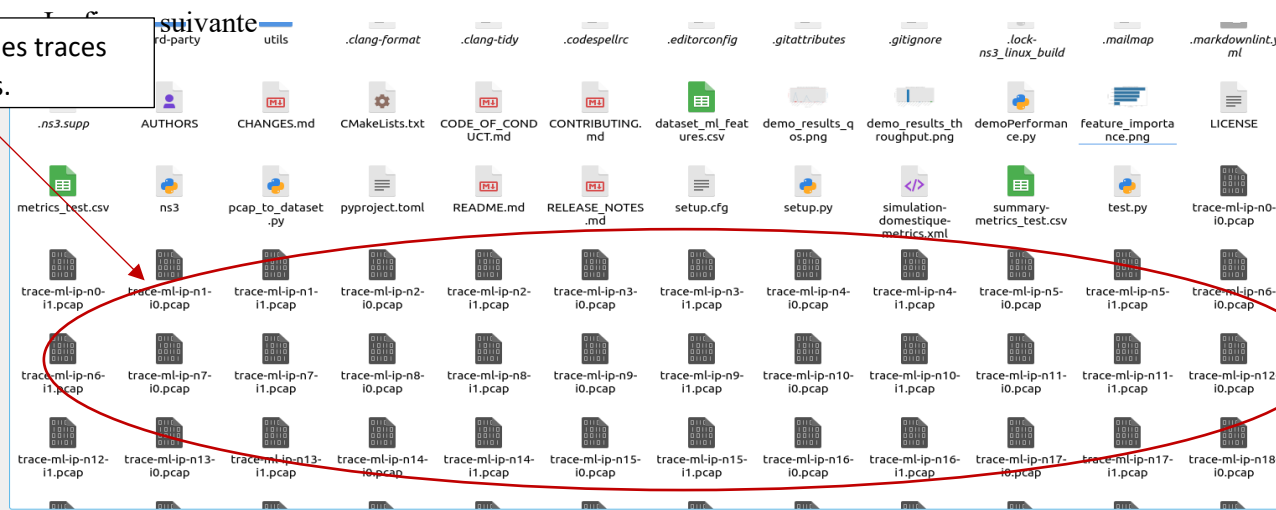
```

Lubuntu@Lubuntu22-virtualbox:~/Desktop/ns-3-dev$ ./ns3 run scratch/simulation-domestique -- --duration=300 --enableFlowMonitor=true --enablePcap=true --enableCsv=true --csvOutput=metrics_test.csv --flowOutput=traces de simulation
s de simulation test fm.xml
Récepteur déjà installé sur le nœud 34 port 9003; réutilisation du récepteur existant.
Installation d'un récepteur sur le nœud 10 -> 10.1.1.20:9004
Installation d'un récepteur sur le nœud 19 -> 10.1.1.21:9004
Installation d'un récepteur sur le nœud 36 -> 10.2.5.2:9005
Installation d'un récepteur sur le nœud 20 -> 10.1.1.22:9006
Récepteur déjà installé sur le nœud 36 port 9005; réutilisation du récepteur existant.
Installation d'un récepteur sur le nœud 21 -> 10.1.1.23:9006
Récepteur déjà installé sur le nœud 36 port 9005; réutilisation du récepteur existant.
Installation d'un récepteur sur le nœud 23 -> 10.1.1.25:9006
Installation d'un récepteur sur le nœud 37 -> 10.2.6.2:9007
Récepteur déjà installé sur le nœud 37 port 9007; réutilisation du récepteur existant.
Récepteur déjà installé sur le nœud 37 port 9007; réutilisation du récepteur existant.
Récepteur déjà installé sur le nœud 37 port 9007; réutilisation du récepteur existant.
Installation d'un récepteur sur le nœud 28 -> 10.1.1.30:9008
Installation d'un récepteur sur le nœud 39 -> 10.2.8.2:9009
Installation d'un récepteur sur le nœud 30 -> 10.1.1.32:9010
Installation d'un récepteur sur le nœud 41 -> 10.2.10.2:9011
Installation de la capture PCAP : traces-simulation-domestique*
Activation de la capture PCAP sur l'interface AP (traces-simulation-domestique-ap)
FlowMonitor enregistré dans traces de simulation test fm.xml
Téléchargement (Nœud 18, Port 9004) | Reçu: 0 Octets | Débit: 0.000 Mbps | Perte: 100.000% | Délai moyen: 0.000 ms | Jitter moyen: 0.000 ms
Téléchargement (Nœud 19, Port 9004) | Reçu: 10000000 Octets | Débit: 2.667 Mbps | Perte: 0.132% | Délai moyen: 5.145 ms | Jitter moyen: 0.067 ms
VoIP LiaisonDescendante (Nœud 20, Port 9006) | Reçu: 1761120 Octets | Débit: 0.047 Mbps | Perte: 34.529% | Délai moyen: 1.107 ms | Jitter moyen: 0.116 ms
VoIP LiaisonDescendante (Nœud 21, Port 9006) | Reçu: 2661660 Octets | Débit: 0.071 Mbps | Perte: 0.852% | Délai moyen: 1.731 ms | Jitter moyen: 0.333 ms
VoIP LiaisonDescendante (Nœud 22, Port 9006) | Reçu: 2661660 Octets | Débit: 0.071 Mbps | Perte: 0.792% | Délai moyen: 1.641 ms | Jitter moyen: 0.229 ms
VoIP LiaisonDescendante (Nœud 23, Port 9006) | Reçu: 2670480 Octets | Débit: 0.071 Mbps | Perte: 0.483% | Délai moyen: 1.617 ms | Jitter moyen: 0.311 ms
Diffusion (Nœud 28, Port 9008) | Reçu: 67089480 Octets | Débit: 1.789 Mbps | Perte: 0.000% | Délai moyen: 1.253 ms | Jitter moyen: 0.156 ms
MiseAJourFirmware (Nœud 30, Port 9010) | Reçu: 0 Octets | Débit: 0.000 Mbps | Perte: 0.000% | Délai moyen: 0.000 ms | Jitter moyen: 0.000 ms
Caméra (Nœud 32, Port 9001) | Reçu: 35617200 Octets | Débit: 0.950 Mbps | Perte: 0.185% | Délai moyen: 1.161 ms | Jitter moyen: 0.101 ms
Capteur (Nœud 33, Port 9002) | Reçu: 0 Octets | Débit: 0.000 Mbps | Perte: 11.765% | Délai moyen: 20.508 ms | Jitter moyen: 6.076 ms
AssistantVocal (Nœud 34, Port 9003) | Reçu: 440500 Octets | Débit: 0.012 Mbps | Perte: 0.000% | Délai moyen: 1.202 ms | Jitter moyen: 0.159 ms
VoIP LiaisonMontante (Nœud 36, Port 9005) | Reçu: 10682100 Octets | Débit: 0.285 Mbps | Perte: 0.546% | Délai moyen: 1.126 ms | Jitter moyen: 0.106 ms
Domotique (Nœud 37, Port 9007) | Reçu: 4224 Octets | Débit: 0.000 Mbps | Perte: 1.493% | Délai moyen: 2.230 ms | Jitter moyen: 1.788 ms
Sonnette (Nœud 39, Port 9009) | Reçu: 0 Octets | Débit: 0.000 Mbps | Perte: 0.000% | Délai moyen: 5.349 ms | Jitter moyen: 1.754 ms
Supervision (Nœud 41, Port 9011) | Reçu: 44400 Octets | Débit: 0.001 Mbps | Perte: 1.333% | Délai moyen: 1.290 ms | Jitter moyen: 0.261 ms
CSV des métriques FlowMonitor sauvegardées dans metrics_test.csv
Résumé CSV par application sauvegardé dans summary-metrics_test.csv
Métriques sauvegardées dans simulation-domestique-metrics.xml

```

Ensuite voici les traces générées ainsi que les métriques en fichiers .xml et csv, NB : J'ai dû optimiser mon code .cc plusieurs fois pour avoir les .pcap plus raisonnables pour la gestion des ressources. Je suis passé du décodage radio (qui était encapsuler) aux traces IP.

Aperçu des traces générées.

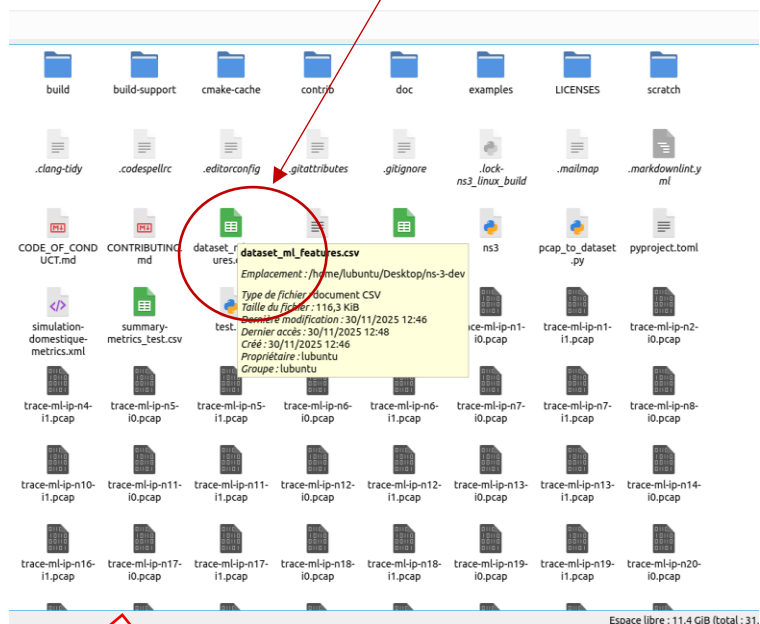


II. Transformation (Du Pcap au dataset)

Méthodologie d'Extraction : J'ai utilisé la librairie `scapy` en Python pour lire la trace PCAP(des fichiers générés ci-dessous) . J'ai itéré sur les paquets, calculé l'IAT à la volée en utilisant l'horodatage des paquets, et construit une trame de données Pandas (DataFrame) contenant les caractéristiques extraites et le label d'application. Le code dans `pcap_to_dataset.py` et le dataset `dataset_ml_features.csv`

Sortie après exécution du code

```
PROBLEMES  SORTIE  CONSOLE DE DEBOGAGE  TERMINAL  PORTS
lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$ python3 pcap_to_dataset.py
-> Lecture : trace-ml-ip-n22-10.pcap
-> Lecture : trace-ml-ip-n25-11.pcap
-> Lecture : trace-ml-ip-n25-10.pcap
-> Lecture : trace-ml-ip-n6-10.pcap
-> Lecture : trace-ml-ip-n37-10.pcap
-> Lecture : trace-ml-ip-n42-17.pcap
-> Lecture : trace-ml-ip-n26-10.pcap
-> Lecture : trace-ml-ip-n24-11.pcap
-> Lecture : trace-ml-ip-n33-10.pcap
-> Lecture : trace-ml-ip-n36-11.pcap
-> Lecture : trace-ml-ip-n11-10.pcap
-> Lecture : trace-ml-ip-n30-11.pcap
-> Lecture : trace-ml-ip-n4-11.pcap
-> Lecture : trace-ml-ip-n8-10.pcap
-> Lecture : trace-ml-ip-n0-11.pcap
-> Lecture : trace-ml-ip-n16-10.pcap
-> Lecture : trace-ml-ip-n37-11.pcap
-> Lecture : trace-ml-ip-n42-19.pcap
-> Lecture : trace-ml-ip-n27-10.pcap
-> Lecture : trace-ml-ip-n12-10.pcap
-> Lecture : trace-ml-ip-n19-10.pcap
-> Lecture : trace-ml-ip-n23-11.pcap
-> Lecture : trace-ml-ip-n12-11.pcap
-> Lecture : trace-ml-ip-n20-10.pcap
-> Lecture : trace-ml-ip-n10-10.pcap
-> Lecture : trace-ml-ip-n40-10.pcap
✓ Succès ! 1820 chunks générés dans 'dataset_ml_features.csv'
lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$
```

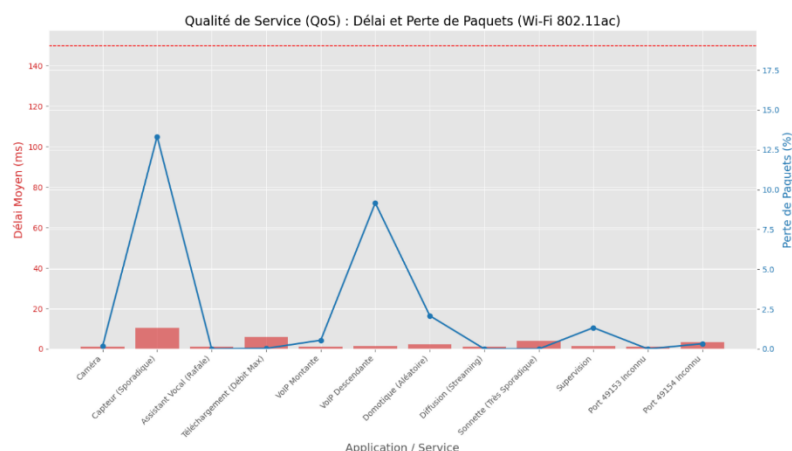
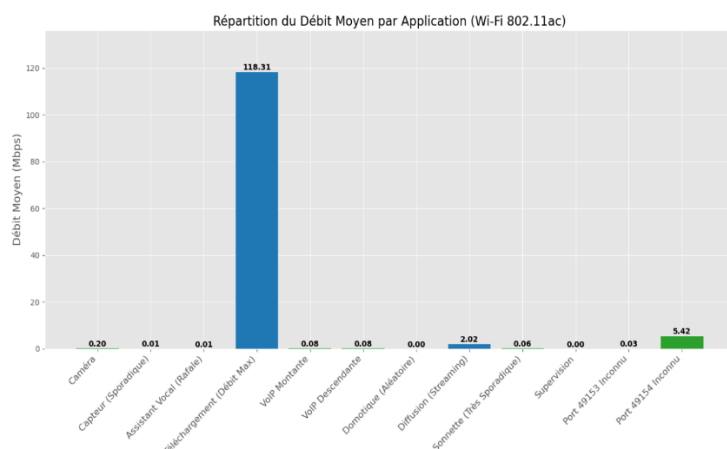


AINSI, LE TRAVAIL DE NS3 EST TERMINER ! NOUS AVONS MAINTENANT LE DATASET

Passons maintenant à quelques analyses de performance

III. Analyse de performances Réseaux(Resultats Qos)

J'ai créé un script python `demoPerformance.py`, j'ai codé l'analyse de **Débit Effectif** (Throughput) mesuré par le FlowMonitor de NS-3 et les métriques de **latence** les plus sensibles à la congestion et aux mécanismes de partage d'accès.



Voir ces résultats (figures) : les remarquent et interprétations sont faits dans la démo

IV. Mise en Œuvre et Résultats de la Classification ML

Choix de l'algorithme : Forêts Aléatoires (**Random Forest**) pour la classification multiclasse.

J'ai codé un script Python `train_classifieur.py` : L'objectif de cette phase est de valider l'hypothèse centrale du Sujet : les caractéristiques de bas niveau (Taille, IAT, Protocole) suffisent à identifier les K=10 types d'applications Wi-Fi avec une haute précision.

```

Lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$ pip install scikit-learn
Requirement already satisfied: scikit-learn in /usr/lib/python3/dist-packages (1.8.0)
Installing collected packages: threadpoolctl, joblib, scikit-learn
Successfully installed scikit-learn-1.7.2 joblib-1.5.2 threadpoolctl-3.6.0

Lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$ python3 train_classifieur.py
--- Chargement du Dataset : dataset_ml_features.csv ---
Nombre d'échantillons (chunks) : 1820
Aperçu des données :
  LABEL  CHUNK_ID  NB_PAQUETS  VOL_BYTES  PROTO_TCP_RATIO  IAT_MEAN  IAT_STD
0      3    chunk_0         3        164             1.0  0.017754  0.017754
1      3    chunk_12        14       5228             1.0  0.016636  0.016385
2      3    chunk_13        98      37596             1.0  0.019679  0.021577
3      3    chunk_25       113     43376             1.0  0.019424  0.020519
4      3    chunk_37        70     27140             1.0  0.017778  0.011059

--- Entraînement du modèle (Random Forest) ---
Précision Globale (Accuracy) : 99.73%

--- Rapport de Classification ---
  precision  recall  f1-score  support
Camera (UDP)      1.00    1.00    1.00         96
Capteur (TCP)     1.00    1.00    1.00          4
Assistant (TCP)   0.89    1.00    0.94          8
Download (TCP)    1.00    1.00    1.00          4
VoIP (UDP)        1.00    1.00    1.00         84
Domotique (UDP)   1.00    1.00    1.00         80
Streaming (TCP)   1.00    1.00    1.00         39

```

Le modèle a correctement classifié l'ensemble paquets

Précise l'équilibre entre précision et rappel

```

Lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$ python3 train_classifieur.py
Sonnette (TCP)      1.00    0.98    0.99    48
Firmware (TCP)      1.00    1.00    1.00     1

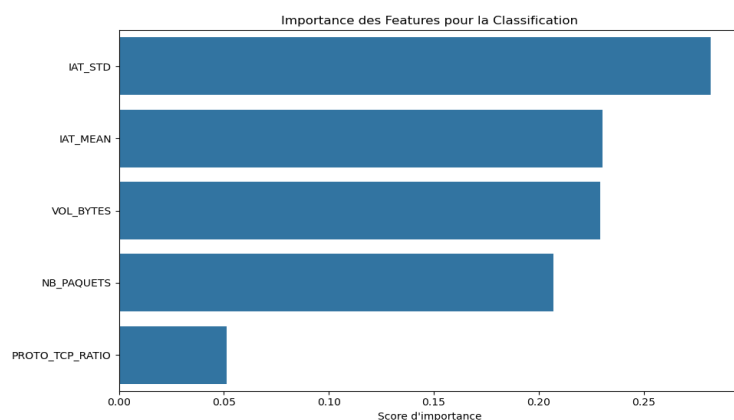
accuracy      0.99    1.00    0.99    364
macro avg     0.99    1.00    0.99    364
weighted avg  1.00    1.00    1.00    364

--- Matrice de Confusion ---
[[96  0  0  0  0  0  0  0]
 [ 0  4  0  0  0  0  0  0]
 [ 0  0  8  0  0  0  0  0]
 [ 0  0  0  4  0  0  0  0]
 [ 0  0  0  0  84  0  0  0]
 [ 0  0  0  0  0  80  0  0]
 [ 0  0  0  0  0  0  39  0]
 [ 0  0  1  0  0  0  0  47]]

--- Importance des Caractéristiques ---
IAT_STD      0.282021
IAT_MEAN     0.230326
VOL_BYTES    0.229333
NB_PAQUETS   0.206818
PROTO_TCP_RATIO 0.051502
dtype: float64

Graphique 'feature_importance.png' sauvegardé.
Lubuntu@lubuntu22-virtualbox:~/Desktop/ns-3-dev$

```



Interprétation : L'IAT_STD (Écart-type du Délai Inter-Arrivée) et l'IAT_MEAN (Moyenne du Délai Inter-Arrivée) sont les deux caractéristiques les plus importantes ensuite les autres : régularité (Caméra) et de la variabilité (Téléchargement, Capteur).

Conclusion Finale : Le succès de la classification (**99.73% d'Accuracy**) est le résultat direct de la capacité des caractéristiques IAT et Taille à capturer les signatures uniques de chaque application, même dans un environnement Wi-Fi congestionné.