Université de Namur
Faculté des Sciences
Département d'informatique

Report : Flagging suspicious hosts from DNS traces

Luyckx Marco 496283
Bouhnine Ayoub 500048

Professors: ROCHET Florentin

Course: Data analysis for cybersecurity ICYBM201

Delivered in November 2023

Academic year 2023-2024

Contents

1 Introduction

This project aims to develop a classifier that is capable of distinguishing between three different kinds of traffic : human, bot and a combination of human and bot. Based on labeled data sets, we aim to train a model to classify each host as either human or bot. This report will outline the methodologies employed, the challenges faced and the results obtained. Additionally, a critical discussion of the result will be given regarding the common pitfalls for learning-based IDS.

The project is supervised machine learning since we have labels for the training datasets !

1.1 How to run the project ?

TODO at the end improve this section

See the main README.md for more information.

2 Explain how we approach the problem

TODO : faire une introduction pour cette section

- we were given 2 trainings datasets : webclients.txt + bots.txt

- evaluation datasets + botlists

So the first thing we had to do is to identify the format of the data that we were working with to be able to identify the relevant features, etc

2.1 Format of DNS queries

We started by examining the structure of a typical DNS query from the tcpdumps that were given. Consider the following line : `13:22:44.546969 IP unamur021.55771 > one.one.one.one.domain 18712+ A? kumparan.com.  (30)`

Breaking this down, we find :

1. **13:22:44.546969** : timestamp at which the DNS query was logged. It indicates the exact date the event occurred.
2. **IP** : Specifies that this log entry is concerning an IP packet.
3. **unamur021.55771** : This is the source of the DNS request. `unamur021` is the hostname of the device making the request and `55771` is the source port number from which the request is coming. Attention à savoir pour ne pas filter sur n'importe quoi : Port numbers are often chosen at random for outgoing requests, so it's not necessarily a fixed port for DNS queries from this device.
4. **¿** : Indicates the direction of the traffic.
5. **one.one.one.one.domain** : This is the destination of the DNS request.

6. **18712+** : This is the query ID for this DNS request. The '+' indicates that this is a recursive query.
7. **A?** : This indicates the type of DNS record being requested.
8. **kumparan.com.** : This is the domain name for which the A record is being requested.
9. **(30)** : This is the length of the DNS request packet in bytes.

When it comes to the response of a DNS query, the format remains mostly identical. However, there are some key differences :

- **X/Y/Z** : This is a breakdown of the answer sections in the DNS response:
  1. **X** : indicates there are X answers.
  2. **Y** : indicates there are Y authoritative nameservers.
  3. **Z** : indicates there are Z additional records.
- **A 104.18.130.231, A 104.18.129.231** : These are the answers to the A record query for the earlier example query `kumparan.com` and it means that the domain has at least two IPv4 addresses associated with it : `104.18.130.231` and `104.18.129.231`.

Understanding the format of these queries is essential for subsequent feature selection. We must identify features that are relevant and impactful in distinguishing between human and bot traffic.

2.2 Our assumptions

To effectively address the problem, we need to acknowledge certain assumptions (some of these will be explain in details later during the pitfalls) :

- Some responses may appear without a preceding request, which we consider as unusual behavior.
- Sampling bias (P1) : we trust the teachers with his datasets, that he gave us good training datasets that are representative from the real traffic.
- Label inaccuracy (P2) : we trust that the teachers correctly labelled the datasets.
- Our analysis is confined to DNS captures and thus, the approach is tailored exclusively for DNS data.
- ? trouver autre chose ici
- ? trouver autre chose ici

2.3 Features selection

The development of an effective IDS based on ML requires careful consideration of the features that will be used for classification. Let's discuss the rationale behind our feature selection :

- **Timestamps** : / : We decided that this feature would not be relevant in our analysis because ....
- **Protocol** : / : When looking at the tcpdumps that are provided, you can see that the protocol is always 'IP' which does not bring any value to the analysis. Thus, we decided to not take it.
- **Hostname + source port number** : / : Does not matter because ...
- **Direction of traffic** : / : This feature does not matter because ...
- **Destination domain** : TODO
- **Query ID** : / : This feature does not matter since the query IDs are fully random which is not useful.
- **Type of DNS record** : OUI
- **Domain name requested** : TODO
- **Length of DNS request** : OUI
- **Length of DNS response** : OUI
- **DNS responses** : OUI
- **counts** : OUI

Hence, we are left with X remaining relevant features and our next step involves exhaustively exploring every possible combination to identify the optimal one with the relevant machine learning algorithm (see next section faire lien hyperref).

faire graphiques pour les autres configs c-a-d des changements et alternances de features qu'on a selectionné pour voir les changements sur l'accuracy du modele + voir avec appropriate baseline

expliquer les choix de features et comment elles influencent l'accuracy mais pas que !

POSSIBLE DDoS : we get a response where we did not ask a question for a NXDOMAIN 14:55:35.387285 IP one.one.one.one.domain ¿ unamur138.47506: 10732 NXDomain 0/1/0 (120) pas de requête

2.4 Diagrams

"to be effective, the IDS must have high accuracy but to be secure, IDS must have high BDR"

- ROC curve =¿ relire base-rate fallacy p 47-48 absisse : False alarm rate ordonnée : Detection rate

absisse : false positive (en %) ordonnée : true positive (en %)

- Recall - Bayesian detection rate (BDR) =¿ relire base-rate fallacy p 44-46 (question qui me vient en lisant : comment on peut, avec les données du projet, calculer : 1. True positive

rate donc le detection rate (TP) 2. False positive rate donc le false alarm rate (FP) 3. False negative rate 4. True negative rate

quand explique a ayoub, checkez fluo p 45) absisse : False alarm rate ordonnée : Bayesian detection rate

- Accuracy - Precision - comparaison entre les différents algos - comparaisons entre les différentes features - precision-recall (recall veut dire true positive rate) absisse : recall (en %) ordonnée : accuracy/precision (en %)

- F1-score =¿ aucune idée de ce que c'est mais ayoub a dit que c'est important

Certainly! The classification report you've provided gives metrics that evaluate the performance of a binary classification model on distinguishing between "human" and "bot". Let's break down each part of the report:

1. **Precision**: This metric answers the question, "Of all the instances the model labeled as a particular class, how many were actually that class?" It's calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- For "human": Of all instances predicted as "human", only 49- For "bot": Of all instances predicted as "bot", 57

2. **Recall**: This metric answers the question, "Of all the actual instances of a particular class, how many did the model correctly identify?" It's calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- For "human": The model correctly identified only 2- For "bot": The model correctly identified 99

3. **F1-Score**: This metric provides a balance between Precision and Recall. It's particularly useful when the class distribution is uneven. The F1-score is the harmonic mean of Precision and Recall:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- For "human": An F1-score of 0.03 indicates very poor performance for the "human" class. This is likely due to the extremely low recall. - For "bot": An F1-score of 0.72 is much better, driven by the high recall.

4. **Support**: This shows the number of actual instances for each class in the dataset that was used to compute these metrics. - 83,314 actual instances of "human" - 108,870 actual instances of "bot"

5. **Accuracy**: This gives the overall proportion of predictions that were correct, across both classes. In this case, 57

6. **Macro avg**: This averages the unweighted mean per label. - Precision, Recall, and F1-score are each averaged, giving equal weight to both classes. - This can be particularly informative if there's a class imbalance, as it treats both classes equally.

7. **Weighted avg**: This averages the support-weighted mean per label. - Metrics are calculated by considering the number of true instances for each label (support). This gives a weighted average which is more representative when there's a class imbalance.

From the report, it's clear that the model does a good job at identifying bots (recall of 99

3 Background knowledge

Prior to the project, we had a whole lesson concerning IDS and about 2 critical aspects of IDS :

- **Effectiveness** : This dimension measures the accuracy with which an IDS **correctly** identifies intrusions within network traffic.
- **Security** : We discussed around the notion that high accuracy alone does not guarantee security. In particular, we emphasized the significance of **bayesian detection rates** and underscored the counter-intuitive nature of **the base-rate fallacy**. This fallacy suggests that even with a high overall accuracy, a system can still yield a considerable number of false positive alarms. Such a scenario is undesirable, and we must be mindful of it.

Summing up the course insights, we arrived at a fundamental understanding that, to be effective, an IDS must achieve a high level of accuracy, but to be truly secure, it must also attain a high bayesian detection rate.

In addition to the course materials, we had to read further research by examining 2 articles closely linked to the project :

1. **The base-rate fallacy and the difficulty of intrusion detection**[2] : TODO demander à gépéto de faire un résumé de précis de l'article + dire que le base-rate fallacy se focus sur la partie effectiveness des 6 problèmes qu'un IDS peut avoir qui sont :
   - Effectiveness
   - Efficiency
   - Ease of use
   - Security
   - Inter-operability
   - Transparency
2. **Dos and Don'ts of machine learning in computer security**[1] : TODO demander à gépéto de faire un résumé de précis de l'article (pas oublier de mettre l'accent à fond la caisse sur les pitfalls vu que c'est ce qui nous intéresse le plus)

3.1 Pitfalls

As stated earlier, from the second article[1], we learnt a lot about pitfalls in machine learning. Understanding these pitfalls is really important for our project since we need to take them into account during our analysis and it will serve as a shield against making erroneous assumptions.

In the forthcoming sections, we will dive deeper into the specifics of these pitfalls, how they apply to our project and what we did to identify them.

OSEF

for US, from the article, the most common is : P1 : 90% =¿ we got it P3 : 73% =¿ we got it P10: more than 50% P9 : more than 50% P7 : more than 50%

donc c'est surement les plus probable qu'on ait dans notre projet à nous aussi, donc faut faire gaffe

/ OSEF

3.2 Data collection and labelling

The first step for the design and development of a machine learning IDS is to obtain a representative dataset. From this stage, we need to consider the two following pitfalls : **Sampling bias** and **Label inaccuracy**.

3.2.1 Sampling bias

In our case, we used datasets provided by the professor, namely 'bots_tcpdump.txt' and 'webclients_tcpdump.txt'. The pivotal question we must ask ourselves is, "Do these datasets genuinely mirror real-world practices (which in our case are the dataset evaluations) ?" If there are not representative of the real-world distribution of human and bot DNS traffic, then our model might not perform well in practice.

Since the datasets where given by our teachers, we've assumed the datasets are representative without extensive verification and that the evaluation datasets (we suppose that the professor has other datasets than from the ones that he gave us) that are going to be tried on our model are similar.

Remarks :

- The limitation of the datasets provided lies in the fact that it has a limited number of DNS traces which do not really represent the true data distribution. Indeed, these traces are restricted to a specific range of hours (within a single day) and as a result, they do not adequately represent the whole network traffic. While efforts can be made to mitigate this bias, it is practically impossible to entirely represent the whole network behavior across all possible scenarios. Since this was given by the teacher, we cannot reduce this issue.

- PAS SUR QUE LE MOT ADEQUAT : Additionally, the bots contained in the DNS trace are performing DNS requests from a particular botnet. Indeed, if there is another kind of botnet that reaches the network, the IDS could fail in identifying the bots since it will perform different requests. Therefore, the data will be different (It is only applicable for the case where we chose the domain as a feature !!) Therefore, they might not represent the broad spectrum of bot behaviors.

3.2.2 Label inaccuracy

Label inaccuracy refers to the incorrectness of the labels assigned to data points in a dataset. Accurate labels are really important for classification machine learning algorithms since they rely on these labels to learn the relationship between the input data and the desired output. If it fails to do so, the overall performance of the model wil be affected.

Since the professor is the one that gave us the training datasets, it is unlikely that the ground-truth of the datasets is wrong, because it would imply that we would have to relabeled all the datasets which would be very very hard. So, the assumption that we are making is that the datasets have been correctly labeled by the professor.

Given that the training datasets were provided by our professor, it is reasonable to assume that the ground-truth labels are accurate because if they are not, it would necessitate to relabel all the datasets which would be very challenging. Consequently, we operate under the assumption that the professor has correctly labeled the datasets.

Remarks :

- If the labeled datasets were classified based on certain heuristics or assumptions, there might be mislabeling. For example, a host with unusual web requests that could be performed by a human might be labeled as a bot.
- Additionally, if the bots change their behaviour after deploying the IDS. This could introduce a bias known as label shift (see this article[1]). The IDS will probably not adapt to these changes which will experience performance decay in real scenarios.

3.3 System design and learning

Now that we have the data, a learning-based IDS can be designed and trained using these data. The goal is to process and use the meaningful features that are interesting for our project.

3.3.1 Data snooping

Data snooping is a phenomenon that occurs when a model is trained with information that is not representative of real-world scenarios. This typically happens when the test set is employed for experimental purposes before the final evaluation. Data snooping can result in overly optimistic results, where the model appears to perform exceptionally well on its training data but fails to generalize effectively to new, unseen data.

In our opinion, we do not fall in this pitfall because our test data is exclusively reserved for the final evaluation and is not used during the training phase.

For the sake of completeness, in the article titled "Dos and Don'ts of Machine Learning" [1], 3 types of data snooping are distinguished : **test snooping**, **temporal snooping** and **selective snooping**. The overview of these snooping types can be found in Table 8 of the article :

- **test snooping** :
  - Preparatory work : This aspect is not applicable to our project. We have ensured that the test set is used **ONLY** for the final model evaluation. No additional knowledge is gained from it during the learning process.
  - K-fold cross-validation : Given our dedicated evaluation datasets, we do not employ K-fold cross-validation in any of our algorithms. (TODO : VERIFY THIS INFORMATION)
  - Normalization : We never use any normalization function in the whole project.
  - Embeddings : TODO : we did not code the neural networks part of the project so the answer could change, but for the moment, the answer is 'no'.
- **temporal snooping** :
  - Time dependency : Since we don't use the timestamps in our project, this feature will not be considered when training the model. If bot behavior is specifically time-dependent (for example, an attack pattern that manifests at particular times), our model will likely not be able to detect it since it doesn't take into account for time.
  - Aging datasets : This does not apply in our case since we use an already provided dataset which is not publicly provided.
- **selective snooping** :
  - Cherry-picking : We do not eliminate any data from our training datasets.
  - Survivorship bias : TODO je sais pas

In summary, we believe that the only type of snooping that could potentially pose an issue for our project is "time dependency" since we do not consider timestamps in our analysis.

3.3.2 Spurious correlations

Spurious correlations involve relying on apparent data relationships that occur by chance. In the course of our project, we needed to select specific features that we believed were important for distinguishing between bots and human behavior (refer to the relevant section for details *[insert hyperlink]*). However, if our training data exhibits spurious correlations between these selected features and the labels, it can lead to model overfitting.

Given that many machine learning models operate as black boxes, it becomes needed to ensure that these selected features are well used within the model. This can be checked by

reducing "unnecessary" data from the original dataset. A common challenge in learning-based IDS is the potential for the model to inadvertently focus on identifying specific hosts within the network, rather than concentrating on recognizing patterns (in our case, to differentiate a bot from an human). This might cause problems and too much focus on how individual hosts behave, making the model not work as well as it should.

3.3.3 Biased parameters selection

This pitfall involves selecting the right hyperparameters during training based on the testing dataset to achieve high accuracy. However, it's important to recognize that the IDS may behave differently in real-world situations.

In the project, we did a pre-selection of the most relevant features (based on assumptions and our background knowledge) and then, we tried every combinaitions of the paramters and checked who had the best accuracy (FAUT FAIRE GAFFE PRC HIGH ACCURACY VEUT PAS DIRE BON IDS). The best-performing model is picked and its performance on the **test set is presented** -¿ this could potentially suffer from a biased parameter selection. For example, over-optimistic results can be easily produced by tuning hyperparameters or calibrating thresholds on the test data instead of the training data.

In our project, we initially did a pre-selection of the most relevant features, guided by our assumptions and background knowledge. Then, we explored all combinations of features to identify the model with the best performance. The model's performance is then assessed using the **test set**. This approach potentially introduces bias in parameter selection. For instance, it can generate overly optimistic results by fine-tuning hyperparameters or adjusting thresholds based on the test data rather than the training data.

Hyperparameters fine-tuned during training may perform differently in real-world scenarios (which in our case, correspond to the different evaluation datasets).

3.4 Performance evaluation

The way that we evaluate our model can greatly influence the outcome and lead to misleading and biased results.

TODO: Comparer avec ROC curve, F1-score, Precision, and Recall (maybe dans result après ??? ou on chevauche pitfall et solution qu'on a faite ???) =¿ excellente question, faut qu'on discute de ca

3.4.1 Inappropriate baseline

To show to what extent a novel method improves the state of the art, it is vital to compare it with previously proposed methods.

In our project, we decided to create 3 different models and see how they perform by comparing them side by side. Indeed, if we directly chose a complexe learning method it will increase the chance of overfitting and will raise also other problems (security, performance etc...)

(Baseline pour nous c logistic regression ? algo simple qui peut servir de baseline ??) Maybe on essaie une non-learning approach pour avoir une bonne baseline ??

3.4.2 Inappropriate performance measures

Using the wrong metric for evaluation. -¿ We've used accuracy, which might not be the best measure if our classes are imbalanced (very likely since we have 5474 lines for bots and 343835 lines for webclients) since true-positive and false-positive predictions will not be observable. Also, ROC curve may not suitable for this case. Therefore, other metrics like precision, recall, or F1-score are more appropriate for our project.

3.4.3 Base-rate fallacy

VERY IMPORTANT FREROT

This concept highlights the tendency to overestimate performance when interpreting metrics for datasets with a significant class imbalance.

If the negative class is predominent (in our case legitimate host), even if we have a low false-positive rate, we could have in practice a high number of false positive.

the base-rate fallacy is about the misleading interpretation of results.

Cette section devra etre TRÈS grosse car on devra reprendre des idées du base-rate fallacy article

base-rate fallacy affects the required performance of the IDS with regard to false alarm rejection

In the training dataset, there are **343835** entries for `webclients_tcpdump.txt` whereas for for `bots_tcpdump.txt` there are only **5474**. This clearly shows that a high accuracy might be misleading.... à developper

il faut parler du false alarm rate

parler des humans aussi qui si trop de choses sont labelissés error, au fil du temps, on perd la trust du system : as the article said, "Trust once betrayed is hard to recover"

on pourrait maybe faire un Venn dessin comme pour la médical p55 de base-rate fallacy

is about the misleading interpretation of results

This pitfall has a big influence on the effectiveness parameter.

3.5 Deployment and operation

3.5.1 Lab-only evaluation

Only testing in controlled environments. -¿ Our evaluation is on the provided datasets. Real-world performance might differ.

timestamps feature is during one hour only (maybe if the bots actively perform more request during a certain period of time - outside the time that we have in our dataset - the IDS could not be able to classify it well)

If we take the hostname feature in account during the training and that during the evaluation phase, we encounter a new host that was not previously in the dataset, the model would not know how to behave.

3.5.2 Inappropriate threat model

We need to make assumptions on how we differentiate a human and a bot what are the potential data poisoning on the datasets that where given by the professor ?

pas grand chose d'autres ici je crois

4 Algorithms and techniques used

In selecting the algorithms for this project, our initial step involved determining the most suitable intrusion detection strategies to implement. The seminal work on the base-rate fallacy by Axelsson [2] highlighted three major types of intrusion detection : **anomaly**, **signature** and **classical**.

4.1 Our choice

We opted to focus on **anomaly detection** as our primary approach. Anomaly detection is a technique that identifies deviations from expected behavior within a system or dataset. In the context of our project, it is particularly valuable because it allows us to differentiate between human-generated and bot-generated traffic without relying on predefined "signature" files. This is especially pertinent as we are not provided with such signature files in our dataset.

Anomaly detection, as a machine learning approach, is well-suited to handling cases where the behavior of bots may vary and evolve over time, making it challenging to create precise signatures for them. Instead, anomaly detection algorithms learn the baseline behavior of the network and can identify deviations from this norm. These deviations may indicate the presence of bots, which often exhibit patterns that deviate significantly from human traffic.

A TESTER, UTILISER 'SMOTE' =¿ intéressant pour augmenter le nombre de samples pour maybe éviter le base-rate fallacy

Différences entre traffic humains et bot

- humain : bombarde pdt quelques minutes, sur les +- memes sites

- bot : en continue mais sporadique (par exemple, toutes les 10 secondes), sur full de site différents

https://www.turing.com/kb/zone-intrusion-detection-with-opencv

https://www.geeksforgeeks.org/intrusion-detection-system-using-machine-learning-algorithms/

4.2 Algorithm 1 : Decision tree

Give explanations why we choose the decision tree has a first algo

4.3 Algorithm 2 : Logistic regression

4.4 Algorithm 3 : Random forest

4.5 Algorithm 4 : Neural networks

5 Comparison of the algorithms

6 Discussion and critics about our methodology

6.1 Issues faced during the project

+ parler de ce qui pourrait etre improve

7 Conclusion

Faire un IDS basé sur du ML est super compliqué, pleins de paramètres à prendre en compte pour etre sur que le IDS est accurate et secure

VOIR PITFALLS ici - données de bases : - type d'algo

parler de bayesian etc aussi ici

8 Bibliography

https://www.bibme.org/bibtex

References

[1] Daniel Arp et al. "Dos and Don'ts of Machine Learning in Computer Security". In: *CoRR* abs/2010.09470 (2020). arXiv: 2010.09470. URL: https://arxiv.org/abs/2010.09470.

[2] Stefan Axelsson. "The base-rate fallacy and the difficulty of intrusion detection". In: *ACM Transactions on Information and System Security* 3.3 (2000), pp. 186–205. DOI: 10.1145/357830.357849.

9 For us only

we are expected to provide visuals for the projects deadline : 8th of November. MAKE A requirements.txt at the end

on a fait des constantes pour se faciliter la tache mais le prof veut que les paths soient pris en argument CHANGER CA DANS LES FICHIERS