Detection of Tor traffic in Web access logs using ML

Author: Eric Van de Lande 300068600

Supervised By: Professor Miguel Garzon

Executive Summary

This report outlines the development and implementation of a machine learning-based tool designed to identify Tor traffic within web access logs. The project's motivation stems from the growing need for privacy and security measures to differentiate between normal and potentially malicious use of the Tor network.

Introduction

In the rapidly evolving digital landscape, the increasing volume and complexity of internet traffic present significant challenges for network security. As the internet becomes an indispensable communication network, the need for robust security measures that can handle large data streams and adapt to changing patterns in real time has never been more critical. Simultaneously, the battle to protect user privacy and anonymity continues, with technologies such as The Onion Router (Tor) network at the forefront of this effort. While Tor offers invaluable services for privacy-conscious users, it also serves as a sanctuary for malicious activities, often hindering efforts to maintain online security and order. Recognizing the need for privacy and the equally important demand for security, this project aims to create a safer environment for all internet users alike.

The motivation for this project stems from the limitations of existing traffic classification methods—many of which suffer from low accuracy, high false positive rates, and inadequate efficiency. Building upon previous research, this project leverages the advancements in machine learning to develop a sophisticated model capable of accurately identifying Tor traffic. By utilizing a Correlation-based Feature Selection (CFS) and a Random Forest classifier, the project aims to offer a significant improvement over traditional techniques, both in terms of accuracy and computational performance.

To tackle these challenges, the following methodology was implemented, consisting of data collection, preprocessing, feature generation, model training, and testing. The data was curated from the ISCX-Tor-NonTor-2017 Dataset, provided by the Canadian Institute for Cybersecurity, which presented a balanced mix of Tor and non-Tor PCAP files. A Python script was created to extract and compute an array of packet features, followed by another script to perform feature selection which is an important process to isolate the most indicative features of Tor traffic. I then reveal the technical intricacies of the feature selection process, which hinges on the calculated merit of feature subsets. This involved the use of priority queues and a backtracking mechanism to refine the feature set, ensuring the model was trained on data that fully covered the complex patterns of Tor traffic.

Central to the project's success was the construction of a machine learning pipeline, culminating in a Random Forest model that showed an impressive accuracy rate. This model was then encapsulated within a user-friendly web application, providing a simple interface for users to upload PCAP files and receive analysis on potential Tor traffic within.

This comprehensive report outlines the entire development process, from the conception of the idea to the realization of a fully functional machine learning model.

Related Work

Prior research in the field has focused on various detection methods, ranging from deep packet inspection to traffic analysis. These techniques often faced limitations in accuracy and computational efficiency. This project builds upon these efforts, leveraging a Correlation-based Feature Selection (CFS) and a Random Forest classifier to enhance detection performance.

Tor Detection using a Machine Learning Approach Using Correlation based Feature Selection with Best First and Random Forest (1)

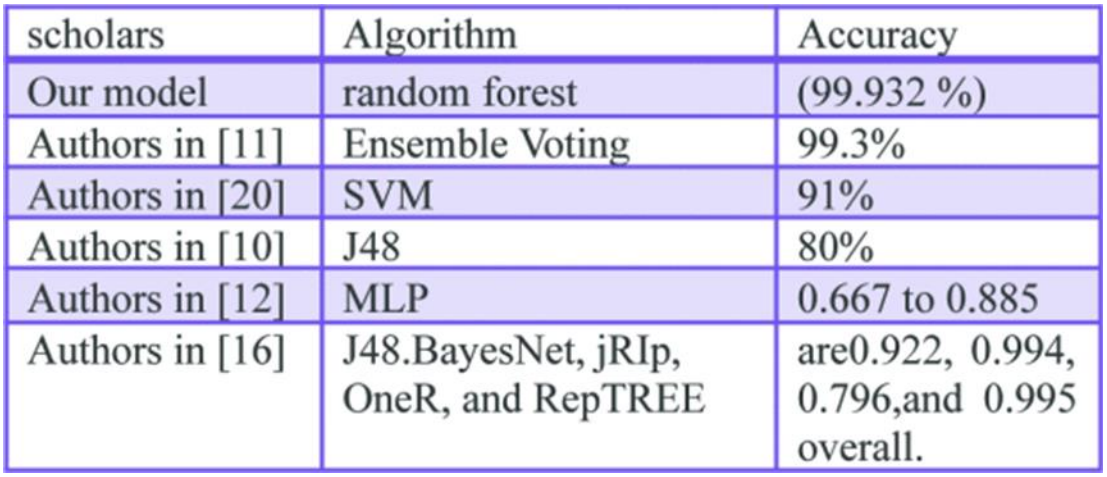
Amongst the noteworthy contributions to this field is the research conducted by Malak Hamad Al-Mashagbeh and Dr. Mohammad Ababneh from Princess Sumaya University for Technology. Their work, "Tor Detection using a Machine Learning Approach Using Correlation-based Feature Selection with Best First and Random Forest," outlines a combination of sophisticated machine learning techniques and comprehensive feature selection processes in enhancing network security protocols.

Their study built upon the foundation of machine learning (ML), delving into its subfields such as supervised and semi-supervised learning, to develop a system capable of identifying Tor traffic with remarkable precision. By harnessing the power of Correlation-based Feature Selection (CFS) and the Best First search strategy, they selected indicative features of Tor traffic. They also selected the Random Forest algorithm, renowned for its collection of decision trees, to classify data with a high degree of accuracy.

The research utilized the robust Weka environment, which provides an assortment of tools for data analysis and algorithmic implementation, to analyze the datasets and save them in the ARFF (Attribute-Relation File Format). Their methodology comprised a careful examination of the available data in Weka, followed by a feature selection and evaluation process that filtered out uninformative features, ultimately leading to a more streamlined and effective classification system.

Their findings showed the significance of time-based characteristics in traffic flows between Tor clients and entry nodes, offering a perspective that showcased the feasibility of detecting Tor traffic with fine-tuned algorithms. The results indicated an impressive accuracy rate of 99.932% in the best-performing scenarios, with low relative absolute errors and strong Kappa statistics, which are indicative of the reliability of the classifications.

In comparison to other scholars' work, the research demonstrated that their machine learning model, particularly when leveraging the Random Forest algorithm, performed with superior accuracy, outpacing a variety of other approaches including Ensemble Voting, Support Vector Machines (SVM), and various forms of neural networks.



scholars	Algorithm	Accuracy
Our model	random forest	(99.932 %)
Authors in [11]	Ensemble Voting	99.3%
Authors in [20]	SVM	91%
Authors in [10]	J48	80%
Authors in [12]	MLP	0.667 to 0.885
Authors in [16]	J48.BayesNet, jRIp, OneR, and RepTREE	are0.922, 0.994, 0.796,and 0.995 overall.

The study concludes with a reflection on the obtained results, noting the impact of flow timeout on the effectiveness of the classification solution and expressing a commitment to further experimentation. The intention is to explore a broader range of algorithms and tools to enhance the detection system and provide an even more robust assessment of Tor traffic flows.

Machine Learning Approach for Detection of nonTor Traffic. (2)

In addition to the prior literature reviewed, the work of Hodo et al. from the University of Strathclyde and the University of Abertay Dundee provides an important perspective on machine learning applications for the detection of Tor traffic. Their study is important in understanding the complexity of intrusion detection systems (IDS) in the context of the Tor network and beyond.

The research by Hodo et al. signifies the ongoing challenges in creating efficient IDS capable of managing the vast amount of data and evolving traffic patterns in real-time situations. They delve into the issues presented by the Tor network: while it provides anonymity and security for users, it also obscures potentially harmful activities. The complexity of Tor's encrypted tunnels makes traffic classification a sophisticated task for modern IDS.

The core contribution of Hodo et al.'s work is the proposal of a hybrid classifier combining an Artificial Neural Network (ANN) with a Correlation Feature Selection (CFS) algorithm. This innovative approach aimed to improve classification performance and efficiently reduce the dimensionality of data for intrusion detection tasks. Their results showed the hybrid classifier, ANN-CFS, outperformed traditional classifiers like Support Vector Machine (SVM) and Naïve Bayes (NB) when applied to the UNB-CIC Tor Network Traffic dataset.

The study emphasizes the significance of feature selection in enhancing machine learning models' efficiency. By selecting the most relevant features from the dataset, the hybrid model achieved high accuracy with a reduced number of features, consequently lowering computational costs and training times. This aspect of their work underscores the importance of careful feature engineering in the development of IDS that can handle encrypted Tor traffic without an excessive computational burden.

Implementation

Summary

The ISCX-Tor-NonTor-2017 Dataset provided by the Canadian Institute for Cybersecurity was utilized. It offers a balanced collection of Tor and non-Tor PCAP files, facilitating a robust training environment for the ML model. The data was cleansed and transformed to ensure compatibility with the ML algorithms. Non-numeric values were converted to floats, and any instances of non-finite values were removed to maintain dataset integrity.

A custom script was developed to extract features from packet headers within PCAP files, producing a structured dataset in CSV format. Features such as packet sizes, timing, and protocol types were included to capture the essence of Tor traffic behavior. A Random Forest classifier was trained using a subset of features identified by the CFS algorithm. Data scaling was applied, and the dataset was split into training and testing sets to evaluate the model's performance.

Feature Engineering and Selection

A critical stage in the preparation of our dataset for machine learning was feature engineering and selection, implemented in a custom Python script. This script processed the concatenated CSV files, each representing a segment of network traffic, into a singular Data Frame with multiple features. There are 26 features:

Feature	Feature Info
Avg_syn_flag	The average of packets with syn flag active in a window of packets.
Avg_urg_flag	The average of packets with urg flag active in a window of packets.
Avg_fin_flag	The average of packets with fin flag active in a window of packets.
Avg_ack_flag	The average of packets with ack flag active in a window of packets.
Avg_psh_flag	The average of packets with psh flag active in a window of packets.
Avg_rst_flag	The average of packets with rst flag active in a window of packets.
Avg_DNS_pkt	The average of DNS packets in a window of packets.
Avg_TCP_pkt	The average of TCP packets in a window of packets.
Avg_UDP_pkt	The average of UDP packets in a window of packets.

Avg_ICMP_pkt	The average of ICMP packets in a window of packets.
Duration_window_flow	The time from the first packet to last packet in a window of packets.
Avg_delta_time	The average of delta times in a window of packets. Delta time is the time from a packet to the next packet.
Min_delta_time	The minimum delta time in a window of packets.
Max_delta_time	The maximum delta time in a window of packets.
StDev_delta_time	The Standard Deviation of delta time in a window of packets.
Avg_pkts_length	The average of packet lengths in a window of packets.
Min_pkts_length	The minimum packet lengths in a window of packets.
Max_pkts_length	The maximum packet lengths in a window of packets.
StDev_pkts_length	The standard deviation of packet lengths in a window of packets.
Avg_small_payload_pkt	The average of packet with a small payload. A payload is considered small if his size is lower than 32 Byte.
Avg_payload	The average of payload sizes in a window of packets.
Min_payload	The minimum payload size in a window of packets.
Max_payload	The maximum payload size in a window of packets.
StDev_payload	The standard deviation of payload sizes in a window of packets.
Avg_DNS_over_TCP	The average of ration DNS/TCP in a window of packets.
Label: 0|1	Respectively if pcap is nonTor or Tor

Data Preprocessing & Data Cleaning

The preprocessing steps included standardizing feature data types, rounding off to three decimal places for precision, and converting labels to integer types. The script also identified and removed features with a standard deviation of zero, indicating no variation in the data, and thus not useful for the model. The cleaning procedures involved replacing infinite values and handling missing data (NaNs), ensuring all features were non-negative, and converting the data to float for consistency.

Feature Selection Algorithm

A significant component was calculating the point-biserial correlation coefficient for feature-class pairs. The script considered the feature-class correlation and feature-feature correlation, calculated using the Pearson correlation coefficient. The correlation analysis was essential for identifying features most indicative of Tor traffic.

A custom Priority Queue class was introduced to manage the feature subsets based on their calculated merit, with a heuristic approach that accounted for both the individual feature's relevance and the redundancy between features. A feature selection loop was implemented to systematically evaluate and select the best features. At each iteration, the algorithm:

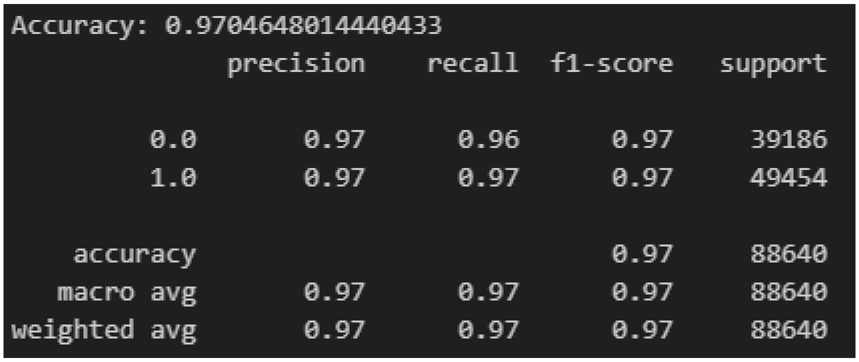
- Identified the feature with the highest correlation to the Tor classification (Label).
- Pushed the feature onto a priority queue based on merit.
- Visited subsets in the queue and assessed them for improved merit.
- Utilized a backtracking mechanism to prevent overfitting and limit computational complexity.

Each iteration refined the feature subset, providing a granular view of which features contributed most significantly to the classification task. The output was a streamlined set of features for effective model training.

This implementation shows the meticulous process of transforming raw network data into a structured, insightful form, preparing it for machine learning analysis. This process not only enhanced the accuracy of the subsequent model training but also improved computational efficiency by eliminating less predictive features.

Results

The application of the feature selection algorithm yielded a feature set optimized for our Random Forest model. This selection process directly contributed to achieving a high model accuracy, evidenced by our test results that demonstrated a consistent 97% accuracy rate. The chosen features captured the essence of Tor traffic patterns, allowing for precise and reliable traffic classification. The high accuracy of the model suggests a promising approach to detecting Tor traffic. However, the project's scope was limited to pre-existing datasets, and future work should involve real-world data for further validation.



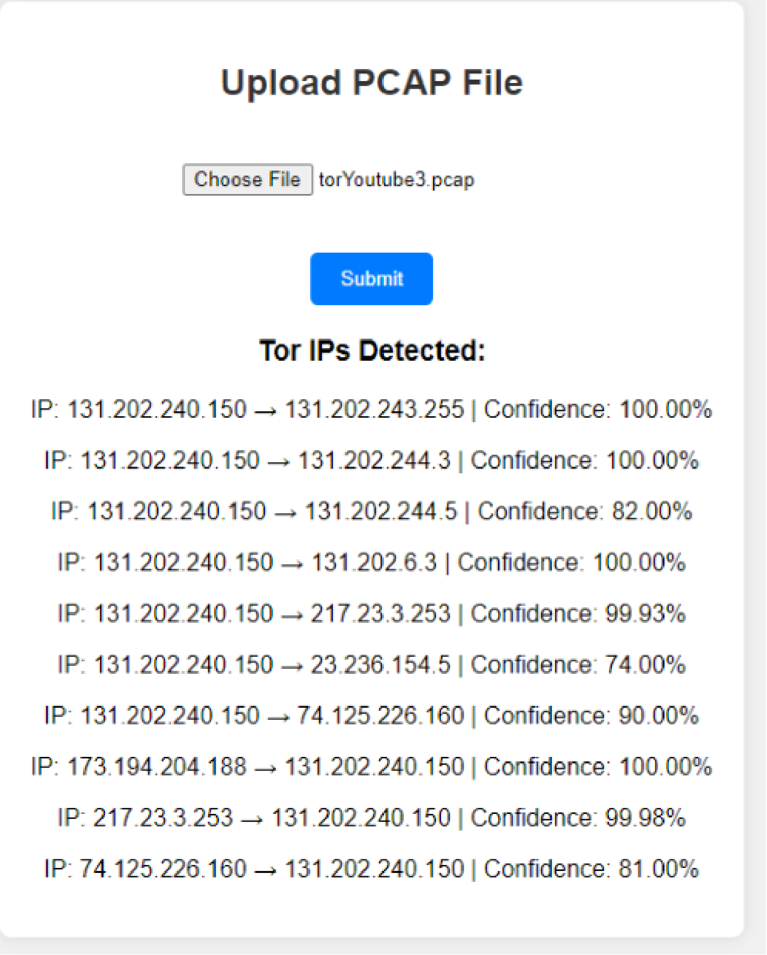
```
Accuracy: 0.9704648014440433
              precision    recall  f1-score   support

         0.0       0.97      0.96      0.97     39186
         1.0       0.97      0.97      0.97     49454

    accuracy                           0.97     88640
   macro avg       0.97      0.97      0.97     88640
weighted avg       0.97      0.97      0.97     88640
```

Web App

The web application demonstrates the model's potential for practical use, providing a user-friendly interface for analyzing PCAP files and identifying Tor traffic with high confidence levels.



Tor IPs Detected:

IP: 131.202.240.150 → 131.202.243.255 | Confidence: 100.00%

IP: 131.202.240.150 → 131.202.244.3 | Confidence: 100.00%

IP: 131.202.240.150 → 131.202.244.5 | Confidence: 82.00%

IP: 131.202.240.150 → 131.202.6.3 | Confidence: 100.00%

IP: 131.202.240.150 → 217.23.3.253 | Confidence: 99.93%

IP: 131.202.240.150 → 23.236.154.5 | Confidence: 74.00%

IP: 131.202.240.150 → 74.125.226.160 | Confidence: 90.00%

IP: 173.194.204.188 → 131.202.240.150 | Confidence: 100.00%

IP: 217.23.3.253 → 131.202.240.150 | Confidence: 99.98%

IP: 74.125.226.160 → 131.202.240.150 | Confidence: 81.00%

Future Work

Moving forward, there are several avenues for further research and development to enhance the capabilities of the traffic classification tool:

Data Collection Expansion:

To test the model more conclusively, a more diverse dataset is needed. This could be achieved by capturing packet data using the Tor browser and Wireshark over an extended period. Alternatively, the tool could be updated to continually capture and analyze packets, providing an active and ongoing detection mechanism.

Algorithmic Exploration:

Although the Random Forest algorithm was effective, exploring additional machine learning algorithms could yield improvements. By testing various algorithms, there is potential to find a model that may perform better under certain conditions or offer improved processing times.

Threshold Optimization:

The web application currently outputs Tor IP predictions without a confidence threshold. Implementing a confidence threshold (e.g., only displaying Tor IPs if confidence is greater than 90%) could reduce false positives and provide users with more reliable results.

Enhanced Packet Window Analysis:

The current model's reliance on packet windows, typically consisting of two IPs communicating, may not always be conclusive. Future iterations could involve analyzing all IPs within a packet window against a database of known Tor relays or utilizing a Tor address checker website. This could significantly improve accuracy but would require a trade-off with increased processing time.

Integration with Tor Relay Databases:

To refine the predictions, integrating real-time checks against databases of known Tor relays would help confirm which IPs are part of the Tor network and if the nodes are currently active. This integration could lead to higher confidence results, albeit with the caveat of potentially longer processing times.

User Feedback Mechanism:

Implementing a feedback mechanism within the web application could allow users to report inaccuracies, thereby creating a dataset of false positives and negatives that could be used to further train and refine the model.

Conclusion

This project represents a significant step forward in identifying Tor traffic for the purposes of privacy protection and security monitoring. The developed tool showcases the effective application of machine learning techniques in network traffic analysis. In conclusion, the research has laid a solid foundation for Tor traffic detection using machine learning. By building on this foundation with the proposed future enhancements, there is a clear path to developing a more robust, accurate, and user-friendly tool for identifying Tor traffic, which has implications for both network security and user privacy.

References:

(1) M. H. Al-Mashagbeh and M. Ababneh, "Tor Detection using a Machine Learning Approach Using Correlation based Feature Selection with Best First and Random Forest," 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 2021, pp. 893-898, doi: 10.1109/ICIT52682.2021.9491772.

(2) Hodo, E. ., X. Bellekens, E. . Iorkyase, A. . Hamilton, C. . Tachtatzis, and R. . Atkinson. “Machine Learning Approach for Detection of NonTor Traffic”. Journal of Cyber Security and Mobility, vol. 6, no. 2, Nov. 2017, pp. 171-94, doi:10.13052/2245-1439.624.

(3) Dataset: http://205.174.165.80/CICDataset/ISCX-Tor-NonTor-2017/Dataset/PCAPs/