Temporal Analysis and Predictions using CVE and CWE codes

KAPSHA SURAJ SNGH

*Dept. of CSE*

*RV College of Engineering*

Bangalore, India email:

[pratikshanng.cs20@rvce.edu.in](mailto:pratikshanng.cs20@rvce.edu.in)

KISHAN KARTHIK S

*Dept. of CSE*

*RV College of Engineering*

Bangalore, India email:

[malavikah.cs20@rvce.edu.in](mailto:malavikah.cs20@rvce.edu.in)

MINAL MOHARIR

*Dept. of CSE*

*RV College of Engineering*

Bangalore, India

email: [minalmoharir@rvce.edu.in](mailto:minalmoharir@rvce.edu.in)

NAMRUTH REDDY

email: [reddynamruth@gmail.com](mailto:sandhya.sampangi@rvce.edu.in)

***Abstract*— Recently, there have been a few situations where security of hundred thousand machine across the world has come to a question. The windows ran in to bluescreen as crowdstrike experience a breach which at that time wasn’t confirmed as a glitch-in-the-system or an attack. Not only proprietry software but also open soruce like unix \ GNU\linux repositories face the issue infact open source is open to more vulnerabilities as its source code is free to edit [ though they maintain protocols to host a modification in public ] and open to read for all. The most popular one and latest one which a CVSS score as high as 10 at once instance was a backdoor attack at libxlzma for the .xz utils which was a dependecy for zip and 7z and other compression tools (more or less) GNU\linux uses. This increase demand and value for a study on the vulnerabilities and its methods of exploitations . This paper give information on the Common Vulnerabilities Exposure and Enumeration based on temporal analysis and predictions made based on CWE codes for the novel attacks which for be prepared for by the commercial world for their privacy and seucrity**

***Keywords— Vulnerabilities, CVE, CVSS, CWE, temporal analysis, data breach, trend prediction.***

1. INTRODUCTION

Temporal analysis and prediction play a crucial role in cybersecurity by examining the timing and patterns of vulnerabilities, such as cataloged in Common Vulnerabilities and Exposures (CVE) and Common Weakness Enumeration (CWE).

CVE codes provide a standardized reference to publicly known security vulnerabilities, while XWE codes classify and descibe software weakness that can lead to vulnerabilities.

By analyzing the temporal aspects of these codes, such as the emergence and frequency of specific vulnerabilities over time, cybersecurity professionals can identify trends and predict potential future threats.

fiThis analysis helps in understanding the lifecycle of vulnerabilities, assessing the effectiveness of security measures, and improving the overall resilience of the system.By leveraging historical data and patterns, temporal analysis enchances proactive defence strategies, ensuring timely responses to emerging security risks

1. Literature Survery

There are a few existing ogranisations that work on collecting the data of CVE entried and few perform data visualisation.

Lili Bo, et.al : The prevalence of multi-core operating systems and the significant improvement in multi-threaded program performance have made concurrency programming mainstream, leading to the emergence of concurrency vulnerabilities. Despite numerous studies addressing software security issues, few researchers have focused on concurrency vulnerabilities. Consequently, a comprehensive analysis of 839 concurrency vulnerabilities from the Common Vulnerabilities and Exposures (CVE) database was conducted, examining trends, classifications, causes, severity, and impact. Key findings include an overall upward trend in disclosures from 1999 to 2021, with race conditions (CWE-362) being the most common type of concurrency vulnerability. The severity of these vulnerabilities is generally medium risk, with CVSS base scores between 4.0 and 6.9, though high-risk instances exist. Concurrency vulnerabilities are almost equally exploitable through local and remote network access, with 53% being local and 45% remote. The majority of these vulnerabilities (571 out of 839) have medium access complexity. Additionally, most concurrency vulnerabilities (785 out of 839) can be exploited without authentication, indicating their ease of exploitation and underscoring the importance of enhanced authentication to mitigate risks. The study highlights that increasing developer awareness and implementing robust security measures are crucial to address the growing number of concurrency vulnerabilities, particularly those leading to denial of service, execution of arbitrary code, and privilege escalation. Historical and ongoing research has primarily focused on detection methods and trends, but this analysis emphasizes the need for improved handling of concurrency vulnerabilities to enhance system security. Recommendations for developers and security administrators include addressing race conditions, enhancing authentication mechanisms, and prioritizing medium-risk and remotely exploitable vulnerabilities to prevent significant security breaches.[1]

Zhen Liu, el. al: The paper discusses a joint neural network model combining an attention mechanism and a bi-LSTM network to analyze and classify relationships in CVE (Common Vulnerabilities and Exposures) dataset sentences. Reinforcement learning enhances the model's ability to recognize high-quality sentences and update its strategy based on rewards. Entities and relationships in CVE descriptions were manually annotated, focusing on the vulnerability attack environment, object, method, and result. The neural network model uses bi-directional LSTM and attention mechanisms, with word and location embeddings to process inputs. Evaluated on 30,000 CVE samples, the model showed superior performance compared to traditional methods, effectively extracting and classifying entities and relationships. Precision, recall, and F1-scores measured performance, with the Union-LSTM model outperforming the original LSTM, especially in noisy data conditions. Reinforcement learning significantly improved handling noise. The extracted entity pairs and predicted relationships were used to construct a Neo4j knowledge graph, enriching network security insights for developers.[2]

MounikaVanamala, et. al: The paper focuses on using the Common Vulnerabilities and Exposures (CVE) repository for cybersecurity purposes, analyzing CVE entries from 2009–2019. It employs Latent Dirichlet Allocation (LDA) Topic Modeling and keyword matching to categorize CVEs according to OWASP Top-10 risks. Analyzing 121,716 unique CVEs, the method facilitates automatic analysis, aiding in creating a vulnerability taxonomy. Previous work involved manual mapping of CVE topics to OWASP risks, setting the stage for the automation introduced here. The LDA technique identifies topics within documents based on word distributions, and preprocessing steps like removing stopwords, lemmatization, and tokenization prepare the data. The methodology includes three phases: data preprocessing, building the LDA model, and rule-based topic classification using OWASP standards. The automated classification method shows high consistency with manual mapping, evidenced by low coefficients of variation (CV) across vulnerability types and time spans, indicating robust automated mapping. Token distribution analysis over five-year periods shows shifts in vulnerability prevalence, such as the increase in tokens for 'A5:2017-Broken Access Control.' The manual mapping provided a baseline, with close alignment to automated results validating the automated technique's reliability. This research significantly impacts cybersecurity practices by automating the analysis and categorization of vulnerabilities, enabling security practitioners to prioritize and address critical risks efficiently. The scalable methodology paves the way for more advanced automated security analysis tools.[3]

Roman Ushakov, et . al: The paper addresses automating the mapping of software product names in system logs to entries in open vulnerability databases, improving the accuracy of identifying known vulnerabilities. It uses the Ratcliff/Obershelp algorithm for comparing strings and detecting similarities, despite minor variations. Implemented in a Python tool, the technique automates mapping to CPE entries, identifying vulnerabilities, and assessing risks based on CVSS scores, achieving 79% accuracy on Windows systems. Compared to previous methods, this approach handles inconsistencies better and reduces the need for manual verification. Future enhancements include analyzing the impact of incorrect mappings on risk scores and integrating machine learning for improved accuracy. [4]

 Stephan Neuhaus, et . al: The study applies topic modeling techniques, specifically Latent Dirichlet Allocation (LDA), to the Common Vulnerabilities and Exposures (CVE) database to uncover prevalent vulnerability types and emerging trends. By analyzing CVE entries from the National Vulnerability Database (NVD) and focusing on descriptions and dates, the study aims to overcome the limitations of fixed classifications like Common Weakness Enumeration (CWE). The methodology involves extracting CVE data, preparing the corpus with stemming and stop word removal, and applying LDA to classify CVE descriptions into 40 topics. Trend analysis reveals changes in topic prevalence over time, highlighting trends such as the decline in PHP-related vulnerabilities and the rise of SQL injection, XSS, and application server issues. The study notes a peak in CVE entries in 2006 and evaluates the effectiveness of LDA through precision and recall metrics compared to CWE classifications. The findings emphasize the flexibility and scalability of LDA-based automated analysis for large volumes of vulnerability data. The study concludes that unsupervised learning techniques like LDA can effectively analyze vulnerability reports and identify emerging security threats, with future work aimed at improving topic modeling techniques and expanding the analysis to include more recent data.[5]

Matthieu Jimenez Snt, el . al: The paper introduces VulData7, a framework and dataset designed to analyze security vulnerabilities in open-source software, focusing on the Linux kernel, Wireshark, OpenSSL, and SystemD. VulData7 automates the extraction and linking of vulnerabilities from the National Vulnerability Database (NVD) to related code instances and patches, providing a comprehensive dataset that includes 2,800 vulnerability instances with detailed information like CVE numbers, descriptions, CWE classifications, CVSS scores, and patches. It currently contains data for 1,600 fixed vulnerabilities. The framework continuously updates and retrieves the latest information from Git repositories and NVD feeds, making it flexible and extendable. Accessible via a Java API and XML exports, VulData7 supports empirical studies, research, and educational purposes by providing rich, real-world vulnerability data. It aims to improve data quality by reducing noise, handling duplicates, and expanding support for other source code management systems. Future enhancements include timeline navigation, better integration with existing tools, and broader support for software projects. Available under the Apache License on GitHub, VulData7 is a valuable resource for developers and researchers focused on secure software development.[6]

Yung-Yu Chang, et . al: One IEEE [7] paper was by a group of university students in Canada who had a very relevant objective [Trend Analysis of CVE for software vulnerable management ] . Though, the data is from 2007-2010 [they claim its from NVD].

They had a conference in 2011 and published in 2012.

We made deliberate efforts to perform a analysis of what could these lack and what could we patched up.

The research on common vulnerabilities and exposures (CVEs) over the period from 2007 to 2010 provides several key insights. It was found that there was a significant 28% decrease in the overall frequency of vulnerabilities, which might indicate improvements in secure coding practices or better vulnerability management processes. However, the decline could also be due to a backlog in reporting, as vulnerabilities discovered in 2010 might have been reported later.

In terms of severity, the study noted a shift from high-severity vulnerabilities to medium- and low-severity ones. Despite this trend, high-severity vulnerabilities still constituted over 45% of the total reported in 2010. This shift suggests a reduction in the proportion of critical vulnerabilities, though they remain a substantial concern.

The exploitability of vulnerabilities showed that over 80% were network-accessible without authentication, highlighting a critical area of concern for network security. These vulnerabilities pose significant risks due to their ease of exploitation.

Regarding CVSS base metrics, there was a 32% decrease in vulnerabilities with low access complexity from 2009 to 2010, potentially reflecting either increased attack complexity or improved defenses. Additionally, vulnerabilities impacting all aspects of confidentiality, integrity, and availability (CIA) increased by about 9%, while those with partial CIA impacts decreased by 12%, indicating a trend toward more comprehensive and critical vulnerabilities.

The study also focused on 15 specific vulnerability types, with Denial of Service (DoS) vulnerabilities showing a notable increase in both frequency and severity, particularly in 2010. This suggests that DoS vulnerabilities are becoming more prevalent and serious.

Finally, the study recommends continued research beyond 2010 to track emerging trends and vulnerabilities. This ongoing analysis will help IT professionals better understand and address evolving security threats.

Ji Sun Park,et . al : Their paper introduces a method for summarizing large-scale CVE (Common Vulnerabilities and Exposures) data using graph-based techniques to help security experts manage and understand complex data more effectively. The CVE data is represented as a graph, with each CVE as a node. Nodes are connected based on the similarity of their features, with edge weights reflecting this similarity.

: The study applies two graph embedding methods: HOPE (High-Order Proximity Preserved Embedding) and Laplacian Eigenmaps. These methods transform the graph into a latent vector space while preserving its structure and relationships. K-Means clustering is then used to group nodes based on these embedded vectors, revealing clusters of related vulnerabilities. The visualizations generated by HOPE displayed more distinct clusters compared to Laplacian Eigenmaps, suggesting that HOPE might better maintain cluster structures.

The summarized graph offers a more compact and interpretable view of CVE data, allowing security experts to quickly identify patterns and relationships. This approach could also be applied to other large-scale graph-based datasets, such as those in bioinformatics, social networks, and defense systems, where understanding complex relationships is crucial.

In conclusion, the study shows that graph summarization and embedding techniques can simplify the analysis of large-scale data, making it more accessible and actionable for experts. [8]

1. Model Inference

This comprehensive report analyzes various machine learning models applied to a dataset of security vulnerabilities to uncover underlying pattern and predict the severity and classification. The models evaluated include Linear Reression, Decision tree Regression, Random Forest, Logistic Regression, Decision Tree Classification, Random Forest Classification, Support Vector Classification (SVC), and K-Nearest neighbors (KNN) existing approaches to develop the framework needs extensive survey.

and Comparative Analysis Using Machine Learning Algorithms [1] studies and contrasts between the different methods of traffic detection and provides an overview of the disadvantages of such traditional methods when it comes to P2P (peer- to-peer) applications which use dynamic port numbers and their ineffectiveness against networks that implement QUIC and TLS 1.3 which have very few unencrypted fields during transmission. Thus, ML based techniques are currently the most preferred method for traffic classification. The work also provides a comparison between different machine learning algorithms in the classification of packets, and finds that C4.5 classifiers obtain the highest accuracy.

Pasyuk A., Semenov E., Tyuhtyaev D in 2019 classified network traffic based on feature selection. Feature Selection in the Classification of Network Traffic Flows [2] talks about the need to classify flows and a short description on the various proposed traffic classification methods. It speaks of the concept of a "flow" and lists the 37 key features out of the 246 statistical features to be used for traffic classification. The paper also highlights the steps involved to clean the dataset captured from applications such as Wireshark, tcpdump packet analysers and other open sources. It has described the four types of sequential selection methods and shown the calculations to select a specific number of features (here 10) out of the 37 for traffic classification using KNN, Random Forest and Gradient boosting ML models. The paper concludes with sequential forward selection methods being suitable for training network flow classifiers based on the analysis obtained from line plots and bar graphs.

1. Tong, H. A. Tran, S. Souihi and A. Mellouk ,in A novel technique for traffic classification based on the convolutional neural networks [3], in 2018 proposed a novel QUIC classifier by integrating the feature extraction unit and the convolutional neural network (CNN) unit. The paper discussed some of the challenges that the current classifiers face that the QUIC-based traffic poses due to its reduced visibility. Hence, the paper proposes a more novel flow static-based method based on CNN that is used to detect different QUIC based services by incorporating flow and packet-based features to improve the performance. The proposed method was successfully able to detect some kinds of QUIC-based services such as Google Hangouts, YouTube, File transfer, etc.

Noora Al Khater; Richard E Overill in 2015 have discussed challenges and techniques for Network Traffic Classification. Classifying network traffic links network traffic with a generated application, is a vital first step for network analysis. It is the core element of network intrusion detection systems(IDS) especially for security purposes such as filtering traffic and identifying and detecting malicious activity. The application of Machine Learning (ML) algorithms in several classification techniques [4],

utilising the statistical properties of the network traffic flow is described. The paper begins with a detailed note on the various proposed methods to classify the network traffic - port-based, payload-based, statistical and behavioural classification. It then discusses the challenges in classification of the traffic using ML models - such as due to higher traffic flows, more complicated ML models leading to high computational costs, trend in the encryption of data packets, etc. It highlights the previous work done in traffic classification using various ML models and algorithms like Expectation Maximisation and concludes that the use of Supervised models to train and of the Unsupervised models to detect new applications in the traffic is one of the best propositions. The paper then highlights the steps involved in traffic classification with sub-flows for faster recognition and timely detection.

In 2020 Shahbaz Rezaei, Xin Liu have proposed Multitask Learning for Network Traffic Classification [5] is proposed To mitigate the need for a large amount of labelled training samples. This paper involves the creation of a multi-task model where bandwidth requirement and duration of a flow are predicted along with the traffic class. It also illustrates that with a large amount of easily obtainable data samples for bandwidth and duration prediction tasks, and only a few data samples for the traffic classification task, one can achieve high accuracy. The input to the model contains the columns time-series, header, payload, and statistical features. The model is trained using CNN. The results show that the accuracy of the traffic class prediction, even with limited labelled samples, is considerably higher with the multi-task learning approach than the transfer learning and single-task learning.

In 2021, Yu Guoy, Gang Xiongy, Zhen Liy, Junzheng Shiy, Mingxin Cuiy, Gaopeng Gouy proposes an end-to-end framework for classifying network traffic employing Generative Adversarial Network (GAN) architecture. GANs are generally used to achieve high overall accuracy without compromising the class balance. The paper [6] proposes a novel GAN architecture that incorporates a classifier which makes the dataset generation process more stable and effective. It provides a comparison some of the oversampling methods like ADASYN, ROS, SMOTE etc.

Maohua Guo, Jinlong Fei, Yitong Meng in 2021 proposed a deep learning model for website fingerprinting in their paper titled ‘Deep Nearest Neighbour Website Fingerprinting Attack Technology’[7]. The paper compares the deep learning model with the older triple fingerprinting model. It also highlights the importance of local fingerprint features along with other general features of website traffic. The deep fingerprinting model uses local fingerprinting features extracted using CNN or convolutional neural networks. It then feeds these features to a K-nearest neighbours or KNN model. The paper also claims that the model has better

performance and is almost immune to concept drift problems.

In 2021, Jean-Pierre Smith, Prateek Mittal, and Adrian Perrig in the Proceedings on Privacy Enhancing Technologies have discussed website fingerprinting. QUIC protocol is harder to fingerprint than TCP because QUIC uses encryption. Website Fingerprinting in the Age of QUIC

[8] investigates the differences in the importance of features from a packet for the two transport layer protocols. Website fingerprinting refers to an observer trying to identify a visited website over an encrypted traffic based on the physical (side-channel) information derived from features like packet sizes and timestamps. The paper gives the idea of splitting a dataset of traffic packets from various websites into monitored and unmonitored web-pages. The authors have adapted the k-fingerprinting (k-FP), Deep fingerprinting (DF), p-FP and Var-CNN classifiers for the same. The paper finds that both QUIC and TCP traces can be fingerprinted easily. However, a model for the classification of a mixer of packets of QUIC and TCP must be developed.

[Vera Rimmer](https://arxiv.org/search/cs?searchtype=author&query=Rimmer%2C%2BV), [Davy Preuveneers](https://arxiv.org/search/cs?searchtype=author&query=Preuveneers%2C%2BD), [Marc Juarez](https://arxiv.org/search/cs?searchtype=author&query=Juarez%2C%2BM), [Tom Van](https://arxiv.org/search/cs?searchtype=author&query=Van%2BGoethem%2C%2BT) [Goethem](https://arxiv.org/search/cs?searchtype=author&query=Van%2BGoethem%2C%2BT), Wouter Joosen in 2017 proposed a novel method based on deep learning. The proposed classifier combines feature extraction with the training process, thus allowing it to classify the packets based on the way it is initially portrayed. Among the existing Deep Neural Networks, three different types of neural networks have been evaluated- feedforward, convolutional and recurrent i.e, Stacked Denoising Autoencoder (SDAE) which is a deep feedforward neural, Convolutional Neural Network (CNN) and Long-Short Term Memory network (LSTM) [9]. They performed semi automatic hyperparameter tuning to evaluate each hyperparameter’s impact. They also discussed a concept called concept drift which involves a decrease in the accuracy of the classifier due to change in the parameters the classifier is using to predict the outcome. For a closed world with 100 websites, the achieved success rate is over 96%, and for our largest closed world with 900 classes, it is over 94%. The most effective deep learning model outperformed the most advanced attack by 2% accuracy in our open world study.

Yong-Jun Wei, Su-Juan Qin in 2016 implemented a web fingerprinting attack based on SSH anonymous communication using random decision forest classifier [10]. They paid more attention to the outgoing traffic sent from server to the client. The model is stable even when the training set is updated frequently.

Various reputed national level institutions and government labs like CAIR, DRDO, IISC, CDAC etc are working rigorously on network traffic analysis. Most of the work published at national level is about basic understanding of QUIC protocol and its performance analysis with existing protocols. There are few papers at international level which

have done characterisation of encrypted traffic to extract domain name and other features. With the baseline of above surveyed we proposed an automated framework to identify QUIC traffic and fingerprinting of the website the user has visited. Further work is extended to simulate drift in data by changing input pattern and dataset characteristics. It automates the identification of variation/drift in Data and analyses the degradation in the performance of the model. The automated framework can be implemented by CNN/GAN based deep learning models to efficiently handle Drift in Data.

III. CONTRIBUTION OF THE PAPER

In light of the rapid adoption of QUIC, this paper aims to:

* 1. To understand the concept of web fingerprinting.
  2. To analyse effectiveness of fingerprinting methods on QUIC packets captured from different sources.

1. METHODOLOGY

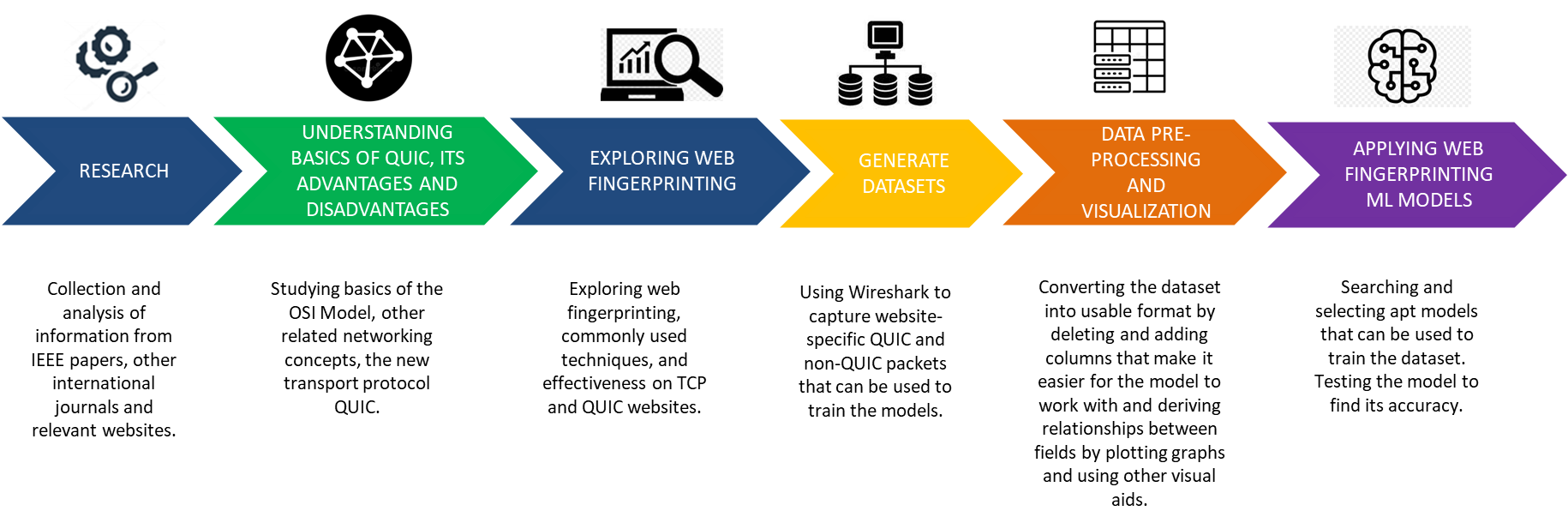


Fig 4.1: Methodology followed

The methodology of the research is depicted in Fig 4.1 and has been explained in this Section in detail.

* 1. DATA GENERATION AND REPOSITORY CREATION FOR DATA SETS
     1. *Data generation for various combinations and capture as pcap*

Publicly available HTTPs data can be gathered by crawling top-accessed HTTPs websites on browsers like Google Chrome, Mozilla Firefox etc., which gives the raw packet capture (pcap) files. The information from the captured file can be extracted using a variety of methods so as to obtain packets of a specific protocol. The most popular options for directly extracting PCAP format files are Wireshark, TCP extract, TCP dump, Pick-Packet, and Network-miner.

The test bed used for the pcap capture includes various OS and web browsers to diversify the data captured as follows: OS: Microsoft Windows 10, Linux

Web browsers: Google Chrome, Mozilla Firefox.

In this paper, Wireshark has been used to capture packets from 50 different websites that use QUIC using a Perl script. In the script, code was written to open the 50 websites one by one and capture packets for a fixed period of 30 seconds.

* + 1. *Repository creation for data set*

The packet data can be exported as a CSV file for the dataset and stored on a remote cloud repository for further use and enhancements. The proposed setup is as shown in

figure 1.1. A python script was written to convert the pcap files to csv and extract the required fields.

Figure 1.1 A scheme for datasets creation

* 1. SET UP OF TRAFFIC PROCESSING TOOL FOR PROCESSING PCAPS AND EXTRACTING FEATURES OR PAYLOAD PORTION

The proposed solution needs to handle diversified traffic hence a python script has been used in conjunction with a primary application which has the packet capture framework initialised already. The script converts the pcap files into csv files by extracting the required features from the captured packet data. Columns are labelled and the csv file is saved as ‘[1-50].csv’ where 1-50 represents the corresponding website number of the site in the list used to refer to it. It will help to extract specific features which can be used for further characterisation. The statistical feature generation and pre-processing is implemented in this step.

* 1. CHARACTERIZATION OF ENCRYPTED QUIC TRAFFIC
     1. *Set up of Machine learning/Deep learning framework* The generated data in step 3 can be analysed using the ML/DL framework. As per the literature, methods used can be p-FP, k-FP, CUMUL, Random Forest Classification, Wang-KNN or Var-CNN.

The network traffic classification structure model includes step by step process as shown in Figure 1.2. The network traffic capture is explained in point 2 (above) – Data generation and Repository creation for dataset.

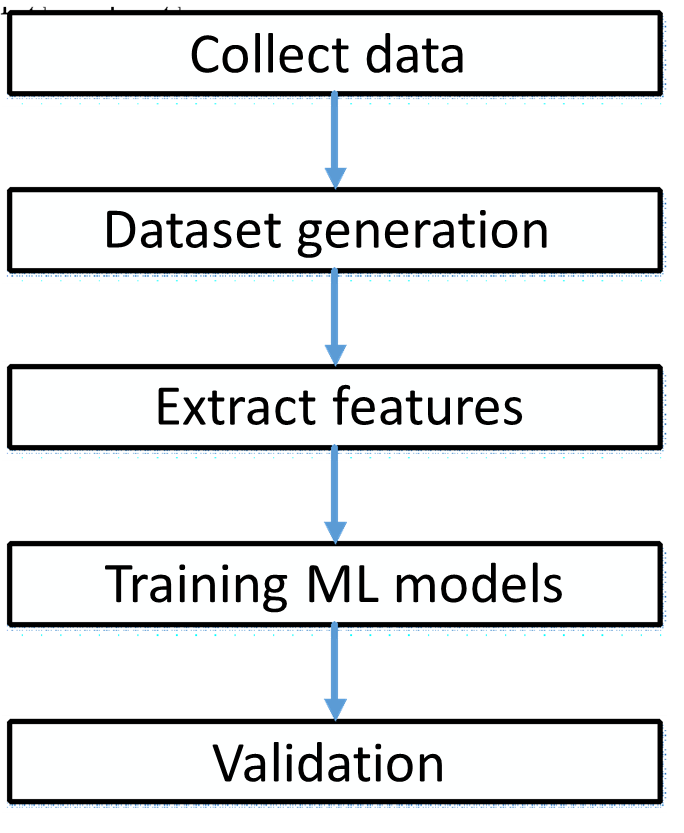


Figure 1.2: Network traffic classification model

* + 1. *Features Extraction Selection focuses on the features that are extracted from the captured data*

The typical features employed in traffic classification literature for modelling traffic can be categorised into the following groups:

* + - 1. Flow Statistics: Measures of central tendencies of features such as TCP flag counts. Flow durations, number of packets, number of bytes, packet lengths, etc.
      2. Raw Bytes: The number of bytes in the header and the payload aggregated.
      3. Time-series: This refers to the data of a flow that can be recorded over consistent intervals of time. For example, the size of packets in a flow is a valid time-series feature.

The final features extracted and chosen after preprocessing of data are elaborated in the Table below:

* + 1. *Training machine learning model on extracted features*

1. Import the required libraries and modules - Pandas,

Numpy, Keras and Sklean

1. Read the csv files containing the captured packets’ data into a dataframe.
2. Append the website number to the corresponding csv files - Each csv file represents a website.
3. Drop the features that are insignificant to train the model- such as the ethernet source and destination addresses, the IP source and destination addresses, IP checksum, TCP or UDP source and destination addresses - and shuffle it. The features have been scaled to increase the accuracy.
4. The results column i.e., name of the website (Y) is to be separated from the rest of the features in the dataframe (X).
5. Using the train\_test\_split function in the Sklearn library, the test and the train data are obtained in an 8:2 ratio. In this stage, data sets are sampled, data are first labelled to classify unknown network applications. The standard ML tools/libraries can be implemented in Python: Sci-kit learn

library, some methods from NumPy, and DataFrame from Pandas have been used.

* + 1. *Implementation of Machine Learning Algorithms*

This is the implementation step which includes applying machine learning algorithms or classifiers on the instances such as applying supervised, unsupervised and semi supervised learning algorithms. In this paper, the implementation has been done on Python using Google Colab. A brief description of the models implemented have been given below:

p-FP: p-FP is a deep learning model consisting of an input layer. One or more convolutional layers are followed by one or more fully connected layers to create a CNN. Convolution, pooling, and classification are the three series of operations that the forward propagation conducts. The convolution operation extracts features from the input layer using filters which can be slided across the width and height of the input layer to form a 2D feature map. These feature maps are used to learn about the pattern. Next, pooling is done to reduce the size of feature mas generated so that the computation required can be reduced. The features learned by the convolutional and pooling layer are fed into the model for classification. This model gave an accuracy of around 70%.

Var-CNN: Var-CNN is a deep learning based website fingerprinting attack. It uses ResNets and convolutional neural networks as its base model. Additionally, it has insights specific to network classification. It uses a combination of manually and automatically extracted features. Timing and direction related features are found to be beneficial to the model. This model was found to have a 1% higher true positive rate as compared to prior models used for the same.

Wang-KNN: Wang-KNN is a supervised learning model that uses distance metrics (Euclidean or Manhattan) to predict the website to which a packet belongs to. While training the model, the dataset is split into test and train sets and distances between the testing and all the training points is determined. The distances are then arranged in the ascending order and the top k rows are selected to determine its class. The class (or website) of the packet is obtained by majority voting or with the nearest neighbour of the testing point.

LSTM: It is a deep learning technique mainly consisting of a series of cells whose output depends on three parameters - the cell state, the hidden state and the current input to the LSTM cell. It is a feed forward network that involves the retention of selective data from the previous state to filter the information required to be fed to the next state. LSTM makes small modifications on the data by simple positional additions and multiplications. In this way, the model forgets and remembers data selectively, which makes it more efficient than Recurrent Neural Networks (RNNs).

CUMUL: CUMUL is a website fingerprinting attack based on Support Vector Machine (SVM). The kernel is the set of mathematical functions used to convert the linear data into high dimension spaces. For CUMUL, the kernel used is Radial Basis Function or RBF. It derives the cumulative length and uses this feature to train the model. It requires a large amount of computation time and resources to complete training. Hence, we only passed a small subset of the dataset to this model.

Random Forest Classifier: Random Forest is an ensemble method involving a number of decision trees. It can also evaluate the importance of the features. Even if a significant amount of data is missing, random forest was able to approximate the missing data and maintain accuracy.

k-FP: The k-fingerprinting classifier integrates two ML models - Random Forest and k-nearest neighbours. The Random Forest technique helps in determining the contribution of various features for website fingerprinting. This is useful to understand which features of the network traffic leak the most information about which website is being accessed by the user. Some of the most important features of the network traffic that helps in website fingerprinting are volume information and timing information. The k-nearest neighbours algorithm classifies the website. If the attacker’s model was trained for the website, the sample website is classified into one of the ‘monitored’ websites. Else, in case of any ambiguity, it is classified as an ‘unmonitored’ website.

1. RESULTS AND DISCUSSION

After applying the ML algorithm, validation of the result has been done using the validation or testing portion of the dataset. The table below gives the accuracy of the different models tested in the paper.

Table 2: Accuracy for models used in the training

|  |  |  |
| --- | --- | --- |
| Number of Packets | Model | Accuracy (in %) |
| 583281 | Var-CNN | 45 |
| 583281 | p-FP | 70 |
| 583281 | Wang-KNN | 83.71 |
| 583281 | LSTM | 19.30 |
| 150000 | CUMUL | 11.65 |
| 583281 | Random Forest | 96.89 |

As noticed, Random forest classification gives the best accuracy whereas k-FP gives the least. This is due to the fact that only a small fraction of the original dataset is used to train k-FP so sufficient data is not obtained. The figure below also gives a graphical representation of the different models and their accuracies.

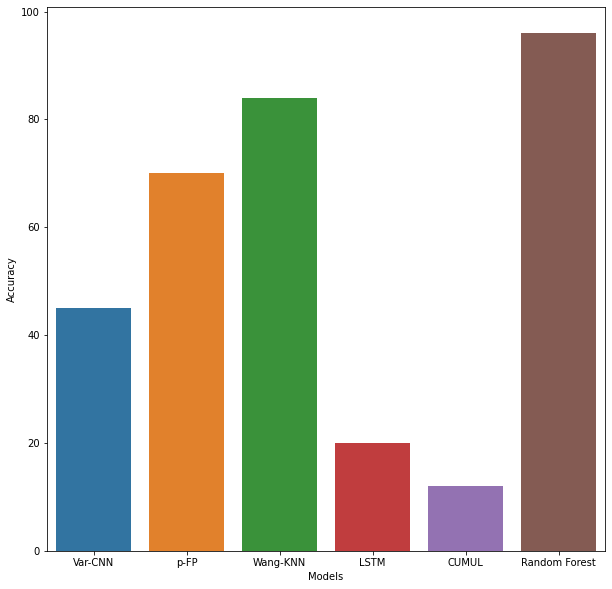


Fig 2: Graphical representation of results

1. CONCLUSION AND FUTURE SCOPE

A website fingerprinting attack uses patterns of data flows, such as packet size and direction, to determine the content of encrypted and anonymous connections. This poses a huge threat to the user’s privacy. Although VPNs try to reduce this type of attack, they are still vulnerable to it. In TCP networks, such attacks have high efficiency and much research has been done on both, fingerprinting as well as defence methods. However, when it comes to QUIC traffic, Website Fingerprinting is still a relatively unexplored threat. Therefore, this paper presents a thorough examination into the efficiency of commonly used fingerprinting techniques that have high efficiency on TCP traffic, on sites that use QUIC.

The dataset used contains both TCP and QUIC packets. The results of the study demonstrate that although these methods are very effective on TCP traffic alone, they give lower accuracy on TCP plus QUIC traffic. As a result, it can be inferred that using the same methods of fingerprinting is not sufficient. Attackers would have to develop new methods or train the current models extensively with QUIC data and combination data with QUIC and TCP packets to attain the same accuracy.

In the future, training with a larger amount of data and including the direction of each trace in the dataset can give a higher accuracy. Due to computational limitations, the k-FP and CUMUL models were run with a subset of the used dataset. Increase in the size of the training dataset would guarantee a better accuracy. Since QUIC is governing most of the internet traffic, attackers will quickly put more resources into improving fingerprinting methods. To resist this, research is needed to develop defence methods specific to QUIC traffic.

1. REFERENCES

[1] Lili Bo, Xing Meng, Xiaobing Sun, Jingli Xia, and Xiaoxue Wu “A Comprehensive Analysis of NVD Concurrency Vulnerabilities" 2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security (QRS)

[2] Zhen Liu, XiaoQiang Di, Wei Song, WeiWu Ren” Vulnerability knowledge extraction method based on a joint extraction model” 2021 Ninth International Conference on Advanced Cloud and Big Data (CBD)

[3]MounikaVanamala, Xiaohong Yuan, Kaushik Roy ”Topic Modeling and Classification of Common

Vulnerabilities and Exposures Database

”2020 International Conference on Artificial Intelligence, Big Data, Computing, and Data Communication Systems (icABCD)

[4]Roman Ushakov, Elena Doynikova, Evgenia Novikova, Igor Kotenko“CPE and CVE-based Techniques for Software Security Risk Assessment”The 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications 22–25 September 2021, Cracow, Poland 978-1-6654-2605-3/21/$31.00 ©2021 IEEE

[5] Stephan Neuhaus, Thomas Zimmermann“Security Trend Analysis with CVE Topic Models” 2010 IEEE 21st International Symposium on Software Reliability Engineering

[6] Matthieu Jimenez Snt, Yves Le Traon Snt, Mike Papadakis Snt, “Enabling the Continuous Analysis of Security Vulnerabilities with VulData7” 2018 IEEE 18th International Working Conference on Source Code Analysis and Manipulation

[7]Yung-Yu Chang, Pavol Zavarsky, Ron Ruhl, and Dale Lindskog “Trend Analysis of the CVE for Software Vulnerability Management” 2011 IEEE International Conference on Privacy, Security, Risk, and Trust, and IEEE International Conference on Social Computing

[8] Ji Sun Park, Mingu Kang, Sungryoul Lee, Dong-Kyu Chae“Graph Summarization for Human-Understandable Visualization Towards CVE Data Analysis”2022 IEEE International Conference on Big Data and Smart Computing (BigComp)