National College of Ireland

A lightweight framework for detecting malicious traffic evasion via DNS over HTTPS

MSc Research Project
MSc in Cybersecurity

Surya Kant Karanwal
Student ID: 23385405

School of Computing
National College of Ireland

Supervisor: Khadija Hafeez

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Surya Kant Karanwal
Student ID:	23385405
Programme:	MSc in Cybersecurity **Year:** 2025-2026
Module:	Practicum Part 2
Supervisor:	Prof. Khadija Hafeez
Submission Due Date:	28th January 2026
Project Title:	A lightweight framework for detecting malicious traffic evasion via DNS over HTTPS
Word Count:	8967 **Page Count:** 20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Surya Kant Karanwal

Date: 26th January 2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A lightweight framework for detecting malicious traffic evasion via DNS over HTTPS

Surya Kant Karanwal
23385405

Abstract

Encrypted traffic of DNS over HTTPS (DoH) uses the same protocol that is used by normal web traffic, which makes it easy for an attacker to use this channel for data exfiltration or as a command and control channel (C2) without getting detected by traditional security devices. As almost all existing studies heavily rely on expensive and high computational solutions and leave a gap for a cost effective and easy to use solution. This study aims to solve this issue by building a lightweight rule-based solution for the detection of malicious traffic hiding in the network stream of DoH traffic. Additionally, this research also addresses the issue of detecting short lived traffic. The final artefact proposed in this study detects harmful DoH tunnels by analysing the patterns and behaviour of the flow using two types of rules, one of which is built using heuristics, such as flow time, packet count, TLS handshake behaviour, byte shares, etc, and the other is built by combining two detection layers, which are heuristics and JA3 signature/fingerprint to tackle both short lived and long lived traffic. As a result, the model was able to give a detection rate of 95% while tested with an individual type of rule, which was increased to 99% when both types of rules were combined. However, the model might need some tuning or adjustment for the detection of unseen malicious behaviour of the flow and TLS handshake.

1 Introduction

Over the past 2 decades, the world has experienced a significant digital transformation, which has resulted in making technology an integral part of everyone's day-to-day life. The internet acts as the backbone of this entire tech ecosystem, which further makes it the core centre of research and development. This continuous research tries to make the internet more secure, reliable and efficient. One of the major enhancements was made in 2018 with the introduction of DNS over HTTPS (DOH). This new design helped DNS traffic to leverage HTTPS for encryption of its packets in the network flow.

Traditionally, DNS uses TCP/UDP port 53, under which all the traffic is sent and received in plain text. This makes it very easy for any threat actor to monitor, capture or alter the DNS information. Due to this loophole, in past years, numerous attackers used DNS to successfully eavesdrop on the network and inject malicious code by altering the request. The vulnerable nature of DNS led to the development of DOH-over-HTTPS, where instead of using TCP/UDP port 53, DNS request uses TCP port 443, which is secure and already in use at a massive scale. This encrypts and hides the DNS packets in the stream of regular and standard web traffic. Now, even though this whole new structure is improving user privacy and mitigating

eavesdropping or manipulation of the data, it brings other security concerns. Threat actors are now utilizing this secure DOH channel to perform sophisticated cyberattacks, such as malware invasion, data exfiltration and malicious command and control channels. Since the DOH packet stream is encrypted, lacks visibility and looks like normal HTTPS traffic, many existing and traditional security solution struggles to identify the malicious DOH traffic and even the normal DOH traffic. This allows attackers to send malware to infect the targeted machine or create a persistent and stable command and control channel, using which multiple other attacks could be performed. (Garmiza, 2022)

Detecting DOH traffic accurately is crucial for network transparency and threat visibility in the enterprise or any other environment. As of now, there are multiple studies active in this field, but most of them heavily rely on machine learning and artificial intelligence. These models require high computational power, a large dataset, continuous retraining, and they also have various limitations, such as a black box nature, lack of model training on real data, lack of integration capabilities with the network devices, etc. These limitations make them less suitable for real world environments, especially for small enterprises and organizations.

This study proposes a lightweight, rule-based framework for the detection of malicious traffic hiding under the blanket of legit DOH flow. For the designing and development of the underlying structure of the model's core engine, TLS level fingerprinting (JA3/JA3S) and flow based heuristics play a crucial role. Furthermore, this rule based model addresses many angles which are important for real world implementation, such as cost, computational power, interpretability, reproducibility and ability to implement in all types of network environments. Additionally, the complete development of this model is done using open source tools, which will make it suitable for startups, small scale organization or even for research purposes.

Research Question
"How can a lightweight rule-based approach developed to detect malicious DNS-over-HTTPS traffic effectively without relying on expensive and high computation methods like machine learning, and how effective will that solution be?"

Limitation:
This research focuses on the development based on HTTP/2-based DOH traffic and does not cover HTTP/3 (QUIC) based DOH traffic due to the limited time and malicious traffic data constraints. HTTP/2 also holds the highest share of 49.6% of the overall internet requests, whereas HTTP/3 only covers about 20.5% (Belson, 2024). Additionally, due to time constraints, advanced features will not be included, such as zero day attack detection

Report Structure
This report will start with a critical analysis of similar existing studies, which will be followed by a methodology section, under which all the necessary steps taken to achieve the end goal will be explained. Afterwards, all the findings during research and analysis of the results will be discussed under the evaluation section. Lastly, the report will end with a conclusion and future work.

2 Related Work

Over the last decades, there has been a massive growth in the study of detecting malicious encrypted traffic traversing the internet, which has made DNS over HTTPS (DoH) a hot topic of discussion among researchers. It has become a challenging task for industry experts to make a unified secure solution to detect malicious DOH traffic without affecting users' security and privacy. There are numerous studies conducted using various methods and technologies, such as machine learning, statistical analysis, heuristics/patterns, and rule based approaches. This literature review will try to analyse all the approaches to understand their strengths and limitations, which will further help in identifying the gaps of the current research and build upon them.

2.1 Machine learning based approaches

(Monroy, Gupta and Wahlstedt, 2023) used an artificial neural network model called an autoencoder. It uses unsupervised learning in the backend, which helped the researchers in detecting malicious DoH traffic without labelling the data. Their model learned the normal traffic behaviour as its baseline, which then served as a threshold for normal traffic and anything that looks different or unusual is considered harmful. For the complete experiment, they used CIRA-CIC_DoHBrw-2020 as the primary dataset, and as a result, they were able to achieve a high recall with F1 score of 99%. Overall, the model performed well in the testing environment, and the detection of unseen abnormal traffic became one of the key highlights of the overall experiment. However, all the testing conditions and the normal baseline were configured according to the selected dataset, which makes this model unfit for the heterogeneous real world traffic. This study displays the strengths of the unsupervised learning approach, but also highlights the limitations, such as less adaptability and interpretability.

Later, (Huang, 2024) tried to provide an interpretable deep learning model, where he focused on delivering good results and also the explanation behind each output by integrating explainability with the help of SHAP methods. This proposed model used packet level flow features as input and was able to produce visual outputs, which help in understanding the detection process. This study was also conducted using the CIRA-CIC_DoHBrw-2020 as the primary dataset, and in the results, they were able to achieve commendable results with an accuracy score of 97% and a recall of 98.1%. This research was able to deliver a non black box behaviour, but it still has many limitations, such as relying on a fixed set of features, requires high computational power and having a limited scope of scalability. All the drawbacks make it less suitable for a real world scenarios.

(Wu, Wang and Ding, 2024) build their own machine learning based system called DoH-TriCGAN for the detection. Additionally, they also used a deep learning model known as a conditional generative adversarial network (GAN) for solving the issue of dataset availability. They also used CIRA-CIC_DoHBrw-2020 for the testing, but they have also produced the fake but realistic DoH traffic for better training and outcomes. As a result, the system was able to achieve more than 99% accuracy when tested on the selected dataset. They tried to solve the issue of limited data, but GAN models are very complex and require good knowledge of machine learning. Moreover, it also requires a lot of computational power and highly skilled engineers for proper operations, which makes it difficult to implement this model in the real infrastructure and especially for small organizations.

(Wang, Li and Wang, 2025) also develop a new system by combining CNN, BiLSTM, and hierarchical attention for detection. This three layer model helped them in understanding and extracting the structure and duration of packets. They also used a method called reptile meta learning, which makes the model more adaptable and effective across different network types. Their model performs well in testing and was able to capture complex patterns, however, the proposed model has a large parameter space and requires extensive training. Additionally, it is computationally intensive and takes longer to produce outputs, it was kind of a theoretical model rather than a practical solution. Later, (Feng et al., 2025) have also developed a new model called DualConBound, an advanced contrastive learning framework. This model captures two separate views of network traffic, which helped in adjusting the decision boundary between benign and malicious traffic. Overall, this unique approach performs very well, with 91% accuracy, and shows promising results with unseen data at 71% accuracy. Making it a strong option for detecting zero day attacks, it also shares the limitations of resource heavy machine learning models, which again makes it less suitable for the real world implementation.

(Fatema Bannat Wala, Campbell and Kiran, 2023) took a different direction and carried out their study by integrating statistical approach and machine learning techniques. They focused on simple features of network traffic and used machine learning models such as decision tree and random forest. They were able to achieve more than 98% accuracy, but they only done this experiment to detect not DoH traffic and not malicious DoH traffic. Later, (Sepideh Niktabe, Arash Habibi Lashkari and Sharma, 2023) also went for a simpler approach and instead of relying on heavy machine learning techniques, they used simple machine learning models such as linear regression and logistic regression. Furthermore, for the training of models, they used attributes like packet size and flow duration. They were also able to produce good results with the CIRA-CIC_DoHBrw-2020 dataset with 95.35% of accuracy. Additionally, it was comparatively less complex and required little less computational power, but they only tested it in a controlled environment with a fixed threshold.

(Zhang et al., 2025) constructed a voting based system called EDDM-SCM using machine learning models, which include random forest, XGboost and KNN. They just tried to distinguish the DoH traffic from the overall network flow with the help of port information, HTTP header and TLS certificate. As a result, they were able to achieve a recall of 94% and reduce the discovery time by 70%. However, they had issues like high false positives and all the results were gained in the testing and controlled environment. Later, (Talabani, Abdul and Mohammed Saleh, 2025) made an interesting and smart improvement by using a method called Ant Colony Optimization (ACO), it helped them with the selection of best features for the model supervised training and testing. Since they are selecting the most relevant and important features, their model was able to produce better results even with different datasets. With this approach, they were able to train their model fast and reduce redundancy. Even after having many solid strengths, it has some limitations, such as dependency on the labelled data, lack of available data and computational power. (Jerabek, Hynek and Rysavy, 2024) performed a comparative analysis of different approaches using various datasets. In their study, they covered machine learning, deep learning and feature based approaches, as a result, they found deep learning models were the best performers in the test environment but there was a drop in accuracy when those models were tested with the unfamiliar data. They used multiple datasets, which later revealed the problem of overfitting in most of the DoH classifiers.

Overall, there was seen a continuous development with the various machine learning areas, from a simple anomaly detection to a more advanced meta learning and deep learning models.

All the proposed models were able to give the high accuracy results, but they have numerous limitations, such as a complex nature, resource heavy, dataset specific, lack training dataset and issue of overfitting. These restriction make these models incompetent for the current real world implementations and especially for small organizations or startups.

2.2 Metadata driven and fingerprint approaches to detect network traffic

(Heino, Hakkala and Virtanen, 2023) conducted their study to take a deeper look at pre hashed values of TLS handshake and understand the differences between various TLS traffic. They went with two distinct approaches, one where they considered each TLS handshake attribute like cipher suite or extension, as a single entity and the other, where they addressed all the values as one document. Additionally, they used K-Means algorithm with Levenshtein distance for topic modelling and to make a cluster of similar types of TLS traffic. They collected all the data from Forcepoint firewall logs, which were generated using Mozilla and Chromium-based browsers. As a result, traffic generated from Chromium-based browsers looked very similar and hard to distinguish, but Mozilla traffic had clear and unique pattern for distinct types of requests. Researchers were able to display a detailed look at TLS protocol and the differences between requests just on the basis of TLS data.

Later in 2024, (Siwakoti and Rawat, 2024) integrated machine learning models and TLS fingerprint with some other TLS information like session duration, number of cipher suites and direction of the traffic to detect malicious traffic. Their model was trained on both benign and malicious traffic TLS data, and as a result, among all the models, the ensemble (ENS) was able to detect command and control malicious channels with 97% accuracy. The researchers were able to demonstrate that the inclusion of JA3 fingerprints with other traffic details can be a real game changer approach. Additionally, already discussed papers like (Zhang et al., 2025) and (Huang, 2024) also leveraged TLS handshake information in their studies for better outputs.

Overall, all these prior studies prove that even if the model does not decrypt the traffic, it is able to identify the DoH traffic on the basis of metadata and TLS handshake information. Moreover, these methods are easy to understand and can be implemented in real world models without any complex issues. However, alone TLS handshakes are prone to false positives which may affect the trustworthiness of the model.

2.3 Detection strategies using Statistical Analysis

(Moure-Garrido, Campo and Garcia-Rubio, 2022) introduces a core idea of developing a non complex and less resource dependent architecture to detect the malicious use of DoH tunnelling. They used only simple statistical information, such as packet size, flow direction, and delay between the packets or requests. During the testing, they configured fixed thresholds in their model for the detection of normal and harmful traffic. As a result, they were able to get an accuracy of 95% without even decrypting the traffic. The model's simplicity, transparency and reproducibility made it feasible for the real world implementation, but the use of only a statistical layer makes this model weak and less effective for the live environment, and an attacker might be able to bypass it with little changes to the packets. Building on the previous study (Moure-Garrido, Campo and Garcia-Rubio, 2023) created a real time detection model for the detection of malicious DoH connections, completely based on a single layer of statistical analysis. Instead of just dividing a bunch of packets into malicious and normal in an offline setup, a live flow was simulated using a PCAP files of well know dataset such as CIRA-CIC_DoHBrw-2020 and DoH-TunnelTraffic-HKD.

The test was initiated with the cleaning and restructuring of the data by grouping packets into TCP connections based on source/destination IP and source/destination port. Furthermore, they computed 37 features in total per connection, which include packet size, time intervals and connection durations. Furthermore, they conducted the experiment in 2 different steps, one where they used machine learning classifiers such as random forest and decision tree to validate the 37 features and select the most useful features for the real implementation, and the other, where they configured the selected features and their thresholds in a simulation test environment. As a result, they achieved 99% accuracy in the machine learning testing, and after further analysis, selected packet delay and packet size as their primary threshold for the network simulation. During the network testing, the packet length was configured with a threshold of 175 bytes and a packet delay of 0.05s, where they observed, the model was able to detect the malicious packet after about 80 packets or 26 seconds of the TCP connection.

This real time model showed promising results with 95% detection rate, but the model was not able to detect the harmful connection with shorter duration, due to which it did not perform well with malicious packets related to dnstt and dns2tcp. Additionally, their proposed model only works on a single statistical layer, which makes it a comparatively weak option for the current internet.

2.4 Research gap and future direction

In this entire literature review, many major progresses and promising results were displayed by various researchers with the help of distinct technologies. However, even after showcasing significant progress there are still multiple limitations attached to it. All the machine learning based models were able to give exceptional results but also have many downsides such as the need of high computational power, black box nature, expertise in machine learning, lack of data availability and high cost, making them unsuitable for real infrastructure. Other methods which were based on TLS metadata or statistical analysis able to give a lightweight and non complex solution, but on the other hand, they are easy to break in.

All these constraints bring problems of interpretability, scalability, adaptability and security. This leaves a gap for a more secure, flexible, lightweight and easy to understand solution, and additionally, one that easily integrates with the real world infrastructure with minimum disruption. This study proposes a lightweight hybrid solution built by combining a rule based approach and the TLS handshake metadata, which will create a more secure, hardware friendly, adaptable and easy to implement model.

2.5 Summary Table

This summary table gives a quick recap of the literature review by highlighting methodology, results and limitations of some of the papers.

Name	Methodology	Results	Limitations
Comparative analysis of DNS over HTTPS detectors	Instead of developing something new, they compared 7 ML models	Showcased the comparative analysis of the accuracy and effect over time	Only a comparative analysis

Detecting malicious DoH traffic: Leveraging small sample analysis and adversarial networks for detection	Distinguish between normal and malicious DoH traffic with the help of a generative adversarial network (GAN), a type of multi classification	Performed really well with Random Forest and XGBoost	Requires labelled data for further learning and is very computationally intensive
An Adaptive DoH Encrypted Tunnel Detection Method Based on Contrastive Learning	For the detection, they used techniques like fusion learning and contrastive sparse autoencoder	Delivered even more than 99% accuracy for some datasets	Heavy rely on machine learning, no real environment use cases were demonstrated
Malicious DNS Tunnel Tool Recognition Using Persistent DoH Traffic Analysis.	Used a technique called hierarchical machine learning and used some tools to generate malicious traffic	The model was able to deliver an accuracy of 98%	Need continuous feature updates for the model and it also heavily relies on machine learning.
Real time detection of malicious DoH traffic using statistical analysis	Used machine learning to identify the best feature, then used those features a detect traffic using a statistical analysis approach	With the ability to give good results with long lived traffic, but had difficulties in identifying short lived traffic.	Used only 2 features in the final model, this may generate more false positives. Additionally used machine learning in the initial pipeline of the feature section.

3 Research Methodology

3.1 Research Approach

This study was conducted using the blueprint of a well-known problem solving approach known as Design Science Research Methodology (DSRM) (Peffers et al., 2007). This framework is mostly adopted in fields like Information Systems, Engineering and Computer Science, and with intensive research, it also focuses on the development of artefacts to solve real world issues, which makes it a perfect choice for this project, as this study also proposed a real world lightweight artefact for the detection of malicious traffic and unauthorized command and control communication channels.

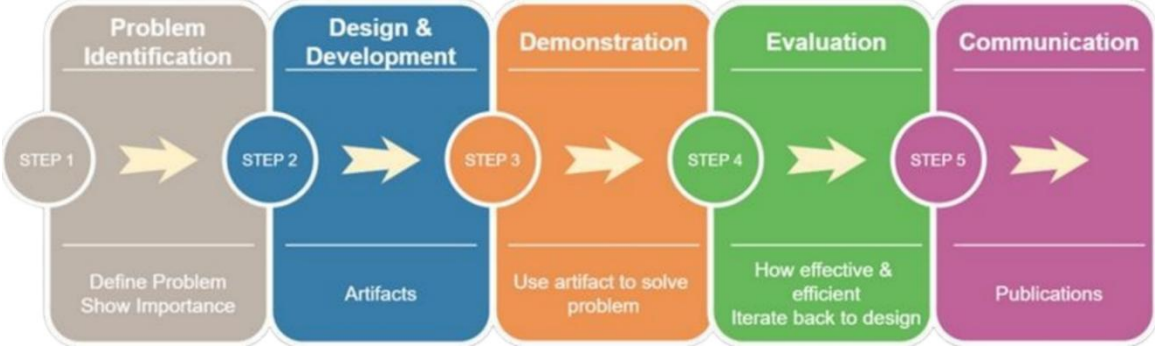

Fig 1. Flow of DSRM methodology (Arsalan et al., 2023)

To begin with, a broad and in-depth literature review was completed to gain an understanding of the DNS over HTTPS underlying architecture, its working and the motivation behind its introduction. Afterwards, the flow of research shifted its focus to the current limitations and challenges of DoH for proper problem identification. During the critical analysis, it was observed that all the existing solutions were dominated by machine learning approaches, and these approaches mostly required high computational power, employees skilled in AI/ML, a large dataset for training and time for decision making. Additionally, all these models possess a black box nature and are also prone to false positives, which makes them infeasible for real world implementation. Moreover, computational overhead and time consuming solutions are not an ideal choice for production infrastructure, where the system should be able to make a decision in seconds, and available hardware will mostly be prioritized for the main products of the companies. Therefore, the rule based solution proposed in this study was developed without the inclusion of machine learning and purely based on industry leading open source tools, which made it easy to implement in the existing architecture without or least service disruption.

Furthermore, all datasets and features were selected based on academic papers to maintain the quality and credibility of the solution. After trial and error with various angles, some of the main features were chosen for the actual model implementation, which include JA3 TLS fingerprint, flow duration, packets sent/received, bytes sent/received and some more flow based which helped in differentiating the pattern of malicious and normal DoH traffic. Later, the successfully implemented detection model was tested using 3 different test cases, to check its accuracy, adaptability and false positives. All the testing was done on the real network traces to observe the actual behaviour of the malicious channels, and the artefact was evaluated on the flow and even on the packet level. Additionally, this model also tried to solve the problem of short lived traffic by developing dedicated rules with heuristics and signatures. Everything was performed in an iterative cycle of design, testing and improving, which made the final solution more robust, reliable and effective.

3.2 Data Collection

In this study, three renowned and publicly available DoH traffic datasets, CIRA-CIC-DoHBrw-2020, DoH-Tunnel-Traffic-HKD and DoH-DGA-Malware-Traffic-HKD are used. Both datasets have live malicious and normal DoH traffic in the form of packet capture (PCAP) files. Additionally, they also cover the DoH traffic generated with help of daily use applications such as Firefox and Google Chrome.

In 2020, the CIRA-CIC-DoHBrw-2020 dataset was developed by the combined efforts of researchers from the Canadian Institute of Cybersecurity (CIC) and the Canadian Internet Registration Authority (CIRA). All the normal traffic was captured from famous DoH enabled browsers connecting with DNS servers of Google and Cloudflare. On the other hand, all the malicious traffic was produced and captured with the help of tunnelling tools. The other two datasets were developed by Mitsuhashi et al. (2022) at Hokkaido University, where the DoH-Tunnel-Traffic-HKD mostly try to cover covert channels, data exfiltration logs, and long lived traffic, along with short lived. Whereas the DoH-DGA-Malware-Traffic-HKD also covers short lived and long lived traffic, along with malware communications generated with the help of domain generation algorithms, where logs show the pattern of malware contacting the attacker server.

Since all three datasets were available in PCAP files, a deep level analysis and investigation on all layers was possible. Additionally, for this study, all the long lived and short lived traffic were deeply studied from the gathered data.

3.3 Data Preparation

For the proper and effective data pre-processing, all the PCAP files underwent through cleaning and organization process, where big files were carefully segmented into small chucks to fast track the analysis with minimum load on the testing environment. Additionally, with the help of the Wireshark tool, a transparent view of the packet was achieved and with the help of in build filters, all the unnecessary and unrelated protocols were removed to minimise the background noise.

Furthermore, with the help of CLI based tool TShark, all the PCAP files were processed for flow level metadata. It helped in extracting the different timestamps, flow durations, packet count, bytes transferred, etc, and later, all the JA3 TLS fingerprints were also extracted with the help of TShark and Python scripts. Additionally, after the extraction, all the data was saved into a CSV files and converted into a consistent format, for futher better analysis and comparison.

3.4 Evaluation Methodology

The testing was done on multiple malicious and benign DoH traffic and replayed multiple time by feeding to the same set of rules, which helped in cross checking the stability and fairness of the deployed rules. Since the proposed model does not take the help of any machine learning algorithms, each test result was checked manually for the correctness, with the help of already existing labels in the datasets.

After each test, three things were noted, which include the number of real harmful flows identified as malicious traffic (true positive), the number of missed malicious flows (false negative) and the flows which are non malicious but the model detected them as malicious (false positives).

The detection rate was calculated with the help of a standard formula of recall.

Detection Rate = TP/TP+FN, where TP is True Positive and FN is False Negative. In other words, the amount of malicious traffic detected divided by the total amount of malicious traffic injected.

4 Design Specification

As discussed, this artefact uses heuristics and TLS fingerprints as the main detection logic, where it also aims to solve the detection issue of identifying encrypted and malicious DoH traffic without requiring the decryption of the packet payload. The proposed detection model has various components in its underlying architecture and each has its own separate role and importance. Now, before selecting each element, a list of key requirements was considered while constructing the system design.

4.1 Design Requirement

These were some of the main requirements which drove all the further choices and development.

- **Encrypted Traffic Analysis –** A system capable of detecting network packet activities without decrypting them, which will ensure the privacy of the user and make the overall process fast.

- **Lightweight Performance –** A system should be efficient enough to work on the minimum computational power, which will make it cost effective.

- **Isolation and Safety –** An isolated system for the development and all testing activities, since all the datasets were live packet captures and may include malicious code inside the packet, which could result in infecting the host machine or even the entire network.

- **False Positive Reduction –** The brain and the logic of the detecting system should be smart enough to differentiate between the normal and malicious traffic and keep the false positives as minimal as possible.

4.2 System Architecture

The model architecture was built using a modular approach, where each component is independent and interchangeable without impacting the other parts of the system. Additionally, the system is not bound to any proprietary technology and is flexible for various platforms and standards. Due to its elastic nature, people could make adjustments and interchange elements they are familiar with.

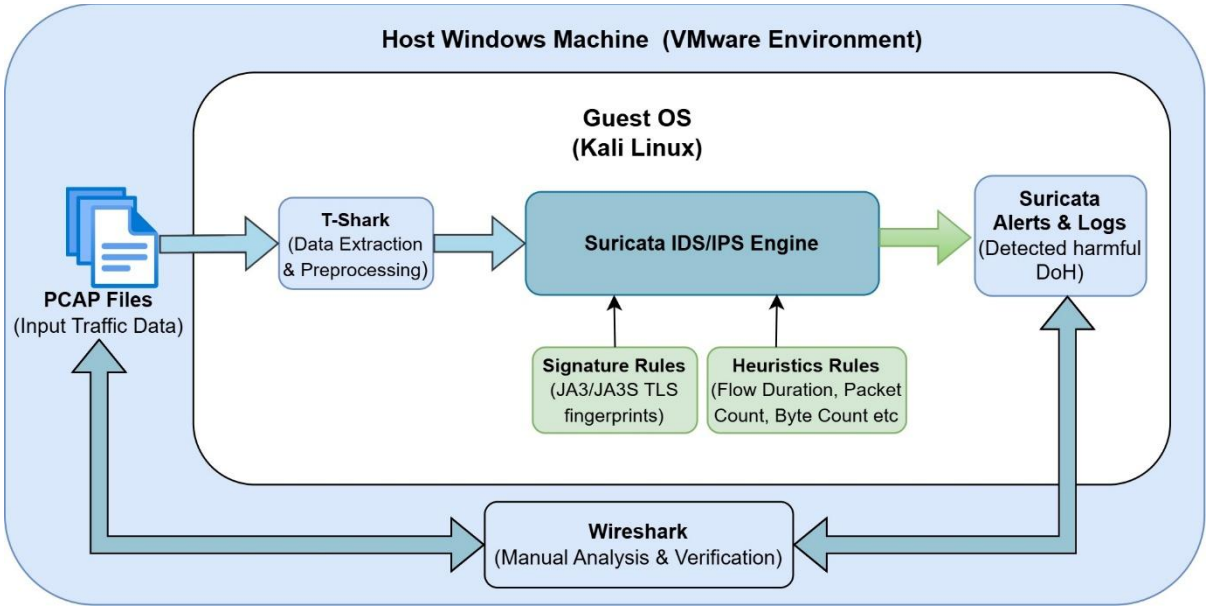

Fig 2. System Architecture

Isolated Virtualized Environment – For the isolation system setup, a VMware workstation, which is a type 2 hypervisor, was installed on a Windows 11 machine and on top of the hypervisor, Kali Linux was installed as a Guest OS.

Detection Engine – In this step, Suricata was used as a primary IPS/IDS engine, which is open source and lightweight.

Analysis – Before feeding the PCAP files to Suricata, all the analysis was performed with the help of Wireshark and TShark.

Verification – After the successful processing of the PCAP file, results are verified with the help of earlier analysis and labels on the dataset.

4.3 Features Selection

All the features were selected after an in-depth analysis of malicious and normal traffic in parallel, where they showed different characteristics and points which could be used to differentiate between the two traffic flows. Additionally, all the features implemented in the final solution can be extracted from encrypted traffic without decrypting the payload of the packet during processing.

Flow Duration – This attribute tells about the amount of time for which a single connection or flow remains active. It was noticed that a malicious connection tends to have a longer time than a normal connection. However, in a few cases, the presence of short lived traffic (less than 1 second) along with long lived traffic was also found in a single connection.

Packet Counts and Directional Patterns – This tells about the number of packets shared between client and server and it also keeps track of their directional distribution. Normal DoH traffic sends less and irregular packets, but malicious traffic has frequent and regular intervals.

Bytes Transferred – This feature tells about the amount of data transferred between a client and server. Due to long flows and more packet counts, the number of bytes/bits shared during the malicious connection is also high.

TLS Handshake Metadata – Even for encrypted traffic, the TLS handshake data is sent in plain text, which could uncover some useful information about the client session, such as TLS version, details of ciphersuits, ALPN identifiers, SNI and extension fields. These attributes could help in distinguishing a non standard DoH implementation.

JA3 TLS Fingerprint – On the basis of all the attributes shared from the client and server side during TLS handshake, a JA3 (client) and JA3S (Server) hash value is generated using the MD5 algorithm. Since some malware families or malicious services share the same JA3 fingerprint, it adds another layer along with heuristics in the deployed system.

4.4 Rule Designing

To analyse traffic from various angles and to produce high confidence outputs, rules were designed in three forms, which include standalone, helper and meta. This approach gives a structured layout to the core logic implementation and also makes the detection more reliable, interpretable, and easy to maintain or modify.

Standard/Traditional Rules – These rules were created using the classic Suricata rule configuration method, where these rules only keep track of a single threshold or conditions, which should be a strong differentiator between connections by its own. These could also be used to generate an early suspicious alert, which can be reviewed by the security professional.

Helper Rules – The model contains multiple helper rules and each helper rule keeps track of a single threshold or condition. These helper rules do not generate any alert, but if any helper rule is matched, its counter (flowbits) gets activated. These flowbits then work as an input for the big meta rules.

Meta Rules – Traffic flow is not directly matched with the meta rule, instead each meta rule takes the input from multiple helper rules. When all the conditions or thresholds mentioned in the metarule are matched, it triggers an alert. Due to this design, an alert is generated when multiple conditions are found and it makes the overall system more accurate and also reduces the false positives.

All these rules give a solid foundation to the deployed solution and this layout also makes it very flexible for any future changes without making any disruption in a live environment.

4.5 System Architecture Flow

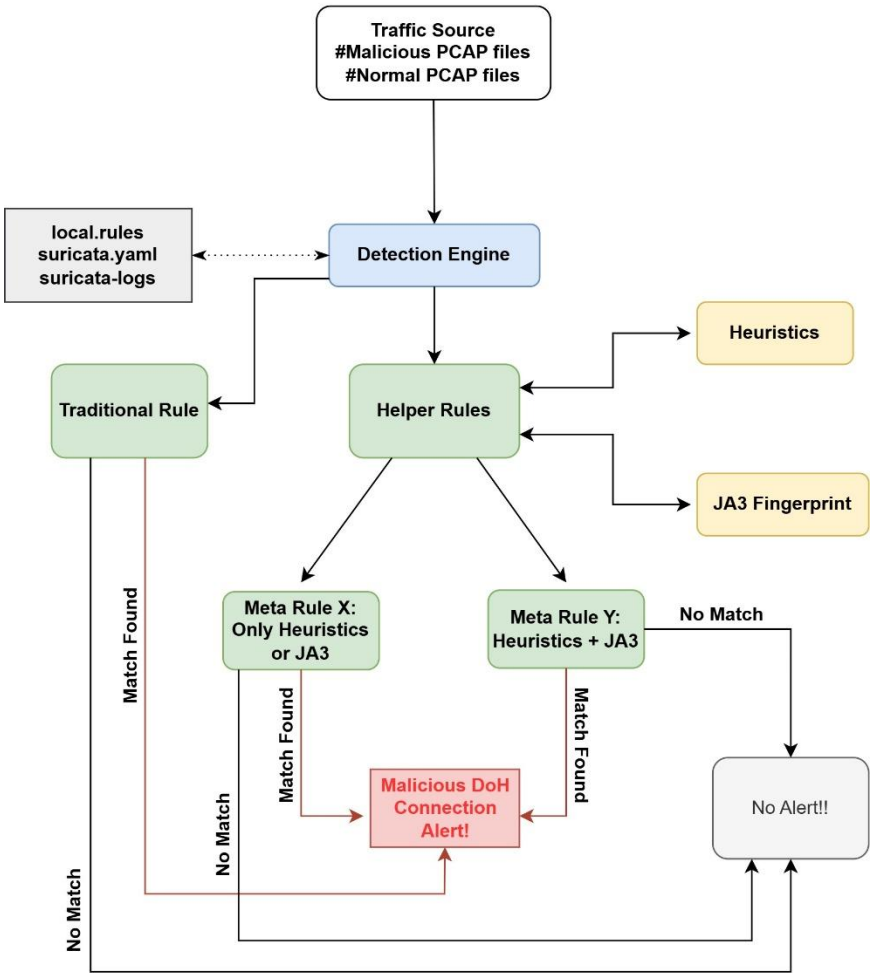

Fig 3. Flow Diagram

The process is initiated by running a PCAP file as live traffic through Suricata detection engine, the entire flow goes by packet by packet, and once the first packet hits the engine, the main work begins, where packets which are from the same network flow are grouped together based on 5 tuples, which include Source/Destination IP, Source/Destination Port and Protocol. While grouping the packets, all the metadata is also extracted for each flow, such as time, size, frequency of packets, signatures etc. In parallel, all the extracted metadata is continuously matched with the configured customs rules saved under local.rules files.

Now, in the case of only heuristics or JA3 Fingerprint + Heuristics rules, just hitting or crossing a single threshold or rule will not trigger any alert, since the model is designed to work on a

cumulative point system with the help of helper rules and meta rules, which help in maintaining high accuracy. Each threshold has its own track and once a configured set of thresholds are crossed, an alert is generated for a malicious traffic presence. Additionally, as a secondary system, there exist some standalone traditional rules which directly get triggered once the condition of that single rule is matched. All the alerts are saved into a log file (fast.logs), which can be easily fetched with the help of some simple commands.

5 Implementation

Under implementation, all the conceptual discussions and studies were converted into a fully working product. This section will showcase all the necessary steps that were taken from setting up the base environment to making adjustments to the core configuration files and designing the rules. Additionally, the justification behind each rule design and the selected threshold will also be discussed.

5.1 Sandbox Setup

Once Kali Linux was up and running in the virtualized environment created using VMware Workstation, all the required tools and datasets were downloaded with the help of the advanced package tool (apt) and a browser, which included Wireshark, Tshark, Suricata, Python with required packages, all 3 datasets and all the other dependencies. After successfully acquiring all the essential things and before interacting with any malicious files, the virtualized system was completely cut off from the internet and the connection between the host machine and guest machine was also terminated for the safe execution of all further activities.

This isolated or sandbox environment was achieved by multiple steps, which include selecting a host only under the network adapter, disabling all the network adapters inside the Guest OS (Kali Linux), disabling shared folder, disabling drag & drop, etc.

5.2 Custom Changes of Core Detection Engine

After the successful base setup, few changes were made to the code of the core configuration file of Suricata. These changes helped Suricata in recognizing the TLS traffic and custom made rules. Some of the main changes include enabling TLS traffic detection and logging, Enabling Support for JA3 & JA3S detection, integrating a custom rule file inside the code and PCAP processing mode was also activated.

5.3 Rule Implementation

The final rule implementation consists of three kinds of arrangements, where each rule is able to target different behaviours of malicious traffic. These rules include only heuristics, only JA3 fingerprint, and heuristics with JA3 fingerprint.

Heuristics Only – These rules look out for strong indicators which are similar to the malicious DoH flow and flag the traffic. In the implementation, these rules use variables like flow.age, flow.bytes, flow.pkt and protocol field for tracking. Since there is a very noticeable gap between the characteristics of normal and malicious traffic, these rules perform really well.

JA3 Fingerprint Only – These rules look out for the known DoH malicious TLS fingerprint. Additionally, they could also be configured in another way, where they will only pass the traffic if there exists a known fingerprint for it in the database and all the other unknown fingerprints found will be flagged for review.

Heuristics + JA3 Fingerprint – These rules will look at both angles, which also makes it a high confidence rule. These rules will only trigger when a specific condition will be matched along with known malicious TLS fingerprints. These rules were implemented to detect the short lived traffic cases because a heuristic alone was prone to more false positives in this kind of case. By combining both the layers, a significant drop was seen in the false positives and an increase in the detection ratio.

6 Evaluation

The model's brain logic was purely built using all the information gained during the in depth research and manual analysis. Each test was performed repeatedly with certain expectations to check the reliability of the deployed rules. Additionally, in most cases, testing was done using the same set of rules to check the adaptability with different network flows and network environments. Finally, after all the analysis and testing, the findings and results were evaluated for the actual ground truth.

6.1 Research & Analysis Findings

Before the development of any logic for the detection system, a vast literature review and a deep packet inspection were conducted. During this period, various characteristics of the normal DoH network flow and the malicious command and control channel over DoH were studied, which further helped in differentiating between both types of connections. Additionally, within some malicious DoH traffic, two types of flow coinciding within the same network stream were also observed. Some of the observations are as follows:

Long Lived Flows – Flow time for malicious DoH traffic ranged between 5 and 35 minutes, which is considered too long, whereas flow time for normal traffic was mostly between 0.1 and 45 seconds, which was similar to usual traffic.

Consistent Pattern – Unlike normal traffic, malicious traffic showed repetitive and consistent flow patterns throughout the attack, for example, if one flow in the traffic was of around 2260 seconds and exchanging 582 KB of data, then all the other flows were also following the same pattern, unlike normal traffic (as shown in Figure 4).

TCP Conversations
Filter:<No Filter>

		←		→		Total		Relative	Duration
		Frames	Bytes	Frames	Bytes	Frames	Bytes	Start	
192.168.11.12:35324	↔ 192.168.11.16:443	2018	322 kB	2017	260 kB	4035	582 kB	18127.515718000	2255.0077
192.168.11.12:42056	↔ 192.168.11.16:443	2019	322 kB	2016	260 kB	4035	582 kB	305889.536233000	2274.4321
192.168.11.12:35642	↔ 192.168.11.16:443	2018	322 kB	2016	260 kB	4034	582 kB	49845.392530000	2271.5825
192.168.11.12:35832	↔ 192.168.11.16:443	2018	322 kB	2016	260 kB	4034	582 kB	70219.716914000	2269.0304
192.168.11.12:36034	↔ 192.168.11.16:443	2018	322 kB	2016	260 kB	4034	582 kB	92867.731577000	2267.8880
192.168.11.12:42060	↔ 192.168.11.16:443	2018	322 kB	2016	260 kB	4034	582 kB	310435.500021000	2274.4662

Fig 4. Consistent Pattern in the malicious file

Similar Inbound & Outbound data – It was also observed that in some cases, the amount of data being sent and received was in the same ratio. Whereas in normal web behaviour, inbound and outbound data in not similar.

Short Lived Brust – In some malicious network traffic, the presence of short lived traffic was also found. The average single short flow lived for around 0.05-0.07 seconds. These streams showcased the presence of beacon traffic, which might be used for keeping the tunnel alive.

Behaviour of Benign Traffic – Benign traffic was not consistent and had a variety in packet count, size and duration, which showcased a normal end user behaviour. In malicious traffic, the amount of upload data was large, but in normal traffic, the amount of download data was larger, ranging from 90KB to 2MB, which again shows usual web behaviour.

Abrupt Termination of Session – In a normal TLS session, a flow is usually closed with a proper handshake and FIN flag, but malicious flows were terminated abruptly and showed RST flag instead of FIN.

Suspicious SNI – Along with the normal server name indicator (SNI), such as api.snapcraft.io and cd.fwupd.org, suspicious SNI were also found, like dns.host-rcks.net. These suspicious SNIs were repeated in high numbers in the multiple malicious network streams and none of the normal traffic had those suspicious SNIs.

Malicious JA3 Fingerprint – The JA3 fingerprint found in the malicious flows did not match any of the normal flows in the testing dataset. Additionally, these fingerprints were also seen in a consistent and repeated pattern, which shows similar TLS handshake characteristics across all sessions. This also indicated a malicious behaviour of the flow.

Now, based on the findings, the rules were developed into the detection system, which was further tested to find out the model's capabilities of detecting malicious long lived and short lived traffic. Additionally, these rules are also tested on normal traffic to see the number of false positives generated.

6.2 Case Study 1 (Data tested against rules made for long lived traffic)

Long lived DoH traffic in the network stream shows the presence of malicious tunnel hiding in the DoH flow, these tunnels were used for command and control (C2) channels and data exfiltration. To detect long lived traffic, rules were built using the heuristics and JA3 fingerprint. For the thresholds, time was configured as 300 sec, packet and the number of bytes shared was set as 800 and 160000 respectively, so if all the conditions were true, then only this rule would trigger. Additionally, the JA3 fingerprint used was found during the packet analysis.

Malware File Name	Total Number of Malicious Sessions	Total Number of Malicious Sessions Detected	Detection Rate	False Negative
tuns	80	76	95%	5%
tcp over dns	139	136	97.84%	2.16%
padcrypt	218	207	94.95%	5.05%
sisron	27	26	96.30%	3.70%
dnstt	1152	574	49.82%	50.17%

Table 1: Showcase the result of long lived rules testing

Takeaway

In almost all the testing data, the detection rate was near or more than 95 %, which shows the correctness and reliability of the deployed rules. Now, since this testing was done purely against long lived rules, there is a drop in the detection rate of dnstt because this dataset contains around 50% of its flow or sessions as short lived. Other than that, these rules performed well on all the long lived flows and the model was able to deliver good outcomes.

6.3 Case Study 2 (Testing only for short lived malicious data)

Each short lived traffic flow, usually lasting less than 1 second, shows the presence of repetitive and frequent small bursts that are mainly used for data exfiltration without being noticed or as beacon traffic to keep the established malicious tunnel alive. For the detection of this flow, the packet count was limited to 20 or fewer, the range for server to client data transfer was set to 4000 to 8000 bytes, and 500 to 3000 bytes for reverse traffic. The following results only show the number of flows detected out of the total number of short lived flows present in the data.

Malware File Name	Total Number of Short Malicious Sessions	Total Number of Short Malicious Sessions Detected	Detection Rate	False Negative
dnstt	578	576	99.65%	0.35%

Table 2: Showcase the result of short lived rules testing

Takeaway

The rules created for short lived traffic also showcased amazing results and the combination of short lived rules and long lived rules gave around 99% detection rate. In the case of dnstt, when it was tested against the combination of both types of rules, the model was able to detect 1150 malicious flows out of 1152 and delivered a detection rate of 99.83%.

6.4 Case Study 3 (Normal DoH Traffic)

Lastly, all the rules were tested against normal traffic to check the number of false positives generated. Since the model mainly contains two types of rules, only heuristics and heuristics with JA3 fingerprint, the testing was also done separately on both, which further helped in comparing both types of rules.

Type of rules	False Positives
Only Heuristics	2 to 20%
Heuristics with JA3 fingerprint	Less than 1 %

Table 3: Showcase the results for false positives

Takeaway

Almost all the false positives are encountered while testing with short lived rules without any JA3 signature, because a few backend activities of normal browsing can mimic the characteristics of a malicious short lived burst flow. Afterwards, when those same testing was performed by adding the JA3 fingerprint with heuristics, the false positive rate dropped drastically and reached near zero.

6.5 Example of a model in action

In this example, a dnstt malicious file was passed through Suricata. Since it contains short lived and long lived traffic, multiple types of alerts as shown in Figure 5). Here, 3 rules are triggered, rule 300001 is detecting long-lived flows, rule 300002 is detecting short lived flows, and rule 400001 is also detecting short lived flows but with heuristics and JA3 fingerprint.

Fig 5. Generated logs by the detection model

6.6 Discussion

The first layer of the model, which is the heuristics or behaviour based tracking, helps the model to evaluate each packet with its complete flow rather than individually, which gives a more holistic view of the traffic and helps in making better decisions. Additionally, due to this approach, the model does not generate millions of alerts and only shows the necessary output, which makes everything cleaner and easier for a better understanding for an engineer. This layer itself performed really well in the testing and was able to detect malicious flow even when different PCAP files are tested against a single rule, making it flexible and adaptive. This layer also delivered respected results when tested with short lived traffic, but the amount of false positives was around 20%, which was resolved with the addition of JA3 fingerprint. Additionally, during short lived traffic detection, the heuristics layer only looks for sessions which are abruptly terminated, which again boosts the overall performance and reduces the false positives.

The added layer of TLS or JA3 fingerprint/signature in the model helped increase the detection rate and drastically reduce false positives. It turned out to be a strong discriminatory feature because if the attack follows the same pattern during the TLS handshake, there are very high chances that the JA3 hash generated will also be the same (edgewatch, 2023), whereas the probability of the same fingerprint appearing in the normal traffic is very less. Moreover, unlike normal traffic, where almost every TLS session has a different JA3 fingerprint, in a single attack, traffic uses the same TLS stack or attributes in every TLS session, which results in the repeated use of the same JA3 fingerprint throughout the attack.

The traffic analysis also showcased the same behaviour and pattern of both short lived and long lived malicious DoH traffic as mentioned in the early studies like (Moure-Garrido, Campo and Garcia-Rubio, 2023). Additionally, in (Moure-Garrido, Campo and Garcia-Rubio, 2023), the researchers had difficulties in detecting the short lived traffic with their approach, whereas the model proposed in this thesis showed promising results with a high detection rate.

Now, even though the model seems good in many aspects and can be used in a live environment as it is, it still has some limitations that a user must be aware of before installing it into their infrastructure. Since the main engine of this model was developed with the information of known attacks and known TLS/JA3 signatures, it might not perform well in the situation of unknown or zero-day attacks. Additionally, due to safety reasons, all the testing is done using the packet capture (PCAP) files, even though these files simulate like real live traffic, the model might need a few tunings in the threshold while using or testing in a real world environment. Moreover, all the thresholds are configured according to DNS over HTTPS (DoH) traffic, so it might not perform as well with HTTP/3/QUIC traffic or Web 3.0. Even after these limitations, the solutions give the perfect baseline, which can be easily modified or adjusted with some simple tweaks to make it suitable for any environment. Lastly, all the things were done using simple rules based methods, open source tools and without any use of machine learning and still able to give a good output.

7 Conclusion and Future Work

This study was initiated with the goal of solving the issue of detecting malicious activities hiding in DoH traffic, using a simple rule based method instead of relying on high computational technology like machine learning. To achieve this aim, multiple datasets containing PCAP files of normal and harmful traffic were analysed to understand the differences and behaviour/pattern of both types of traffic. An additional deep packet inspection was conducted for malicious PCAP files to study the structure of individual packets and the flow. During this analysis, multiple characteristics of the malicious flow were observed, such as repetitive patterns, long durations, high volume packets shared, large bytes sent, presence of suspicious SNI and consistent JA3 fingerprint throughout the traffic. These findings further helped in establishing the thresholds and other unique attributes for the detection system. For the testing of the developed logics, a sandbox was created using virtualization and Suricata was used as the main detection system. Later, all the rules were configured according to the developed logic and tested in three phases in a safe and isolated environment.

The results from all three test cases demonstrated the model's high efficiency and accuracy in detecting malicious traffic or tunnelling patterns. It was also observed that even a single rule for long lived flows was able to detect malicious flows from different datasets with a detection rate of around 95%, which showcases the flexibility and adaptability of the model. Moreover, unlike existing research, the model also performed well with the short lived traffic, where the identification of the RST flag instead of the FIN flag in the session end helped in making a more tuned rule and a realistic rule. When the testing was done with the combination of all types of rules, the model was able to detect almost all the malicious flows with a 95-99%

detection rate and near zero false positives. Moreover, with simplicity of understanding, deployment and use, it also brings some limitations with it, which include that some environments might require rules tuning or rule addition, might not work well on unseen and zero day attacks. These limitations can be easily adjusted quickly with some rule addition without affecting anything. In future, this model can be improved by adding automatic learning behaviour according to the environment, which could also help in detecting zero day threat. Since this model is purely command line interface (CLI) based, integration with the SIEM tools or network analysis tools can be tested for better visualization of the working.

In conclusion, the artefact and all the research were able to demonstrate that a simple, flexible and lightweight model can perform well in the detection of malicious DoH traffic and deliver good outcomes while maintaining the interpretability and ease of use.

References

Monroy, S.S., Gupta, A.K. and Wahlstedt, G. (2023). Detection of Malicious DNS-over-HTTPS Traffic: An Anomaly Detection Approach using Autoencoders. *arXiv (Cornell University)*. doi: https://doi.org/10.48550/arxiv.2310.11325 .

Huang, R. (2024). A DoH Traffic Detection System Based on Interpretable Deep Learning Neural Networks. *2024 IEEE 7th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pp.802–807. doi: https://doi.org/10.1109/iciscae62304.2024.10761846 .

Wu, S., Wang, W. and Ding, Z. (2024). Detecting malicious DoH traffic: Leveraging small sample analysis and adversarial networks for detection. *Journal of Information Security and Applications*, 84, pp.103827–103827. doi: https://doi.org/10.1016/j.jisa.2024.103827 .

Wang, Y., Li, Y. and Wang, H. (2025). DOH Detection Model Based on CNN-BILSTM-Hierarchical Attention and Reptile Meta-Learning. pp.198–203. doi: https://doi.org/10.1109/mlise66443.2025.11100155 .

Sepideh Niktabe, Arash Habibi Lashkari and Sharma, D.P. (2023). Detection, characterization, and profiling DoH Malicious traffic using statistical pattern recognition. *International Journal of Information Security*, 23(2), pp.1293–1316. doi: https://doi.org/10.1007/s10207-023-00790-z .

Jerabek, K., Hynek, K. and Rysavy, O. (2024). Comparative analysis of DNS over HTTPS detectors. *Computer Networks*, pp.110452–110452. Doi: https://doi.org/10.1016/j.comnet.2024.110452 .

Talabani, H.S., Abdul, Z.K. and Mohammed Saleh, H.M. (2025). DNS over HTTPS Tunneling Detection System Based on Selected Features via Ant Colony Optimization. *Future Internet*, 17(5), p.211. doi: https://doi.org/10.3390/fi17050211 .

Fatema Bannat Wala, Campbell, S. and Kiran, M. (2023). Insights into DoH: Traffic Classification for DNS over HTTPS in an Encrypted Network. doi: https://doi.org/10.1145/3589012.3594895 .

Feng, B., Wang, Q., Nan, Z., Xie, J., Zang, T., Xu, X. and Ma, J. (2025). Towards Open-World DoH Tunnel Detection: A Dual-View Contrastive Learning Framework with Adaptive Feature Boundaries. *2025 28th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp.2617–2622. doi: https://doi.org/10.1109/cscwd64889.2025.11033452

Heino, J., Hakkala, A. and Virtanen, S. (2023). Categorizing TLS traffic based on JA3 pre-hash values. *Procedia Computer Science*, 220, pp.94–101. doi: https://doi.org/10.1016/j.procs.2023.03.015 .

Siwakoti, Y.R. and Rawat, D.B. (2024). Detecting Malicious Traffic using JA3 Fingerprints Attributed ML Approach. *2024 IEEE 44th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, [online] pp.128–133. doi: https://doi.org/10.1109/icdcsw63686.2024.00024 .

Zhang, Z., Cheng, Y., Xu, H., Zhang, Z. and Li, C. (2025). Efficient DNS over HTTPS servers discovery method: A voting-based stacked ensemble model with secure connection metadata. *Computer Networks*, [online] 259, p.111073. doi: https://doi.org/10.1016/j.comnet.2025.111073 .

Moure-Garrido, M., Campo, C. and Garcia-Rubio, C. (2022). Detecting Malicious Use of DoH Tunnels Using Statistical Traffic Analysis. doi: https://doi.org/10.1145/3551663.3558605 .

Moure-Garrido, M., Campo, C. and García-Rubio, C. (2023). Real time detection of malicious DoH traffic using statistical analysis. *Computer Networks*, 234, pp.109910–109910. doi: https://doi.org/10.1016/j.comnet.2023.109910 .

Garmiza, M. (2022). *DNS over HTTPS as a covert Command and Control channel*. [online] www.varonis.com. Available at: https://www.varonis.com/blog/dns-over-https-as-a-covert-command-and-control-channel.

Belson, D. (2024). *Cloudflare 2024 Year in Review*. [online] The Cloudflare Blog. Available at: https://blog.cloudflare.com/radar-2024-year-in-review/.

edgewatch (2023). *JA3 Fingerprinting in Cybersecurity - Edgewatch*. [online] Edgewatch. Available at: https://edgewatch.com/blog/ja3-fingerprinting/ .

Arsalan, A., Burhan, M., Rehman, R.A., Umer, T. and Kim, B.-S. (2023). E-DRAFT: An Efficient Data Retrieval and Forwarding Technique for Named Data Network Based Wireless Multimedia Sensor Networks. *IEEE Access*, 11, pp.15315–15328. doi: https://doi.org/10.1109/access.2023.3244247 .

Peffers, K., Tuunanen, T., Rothenberger, M.A. and Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), pp.45–77. doi: https://doi.org/10.2753/mis0742-1222240302 .