IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 April 2023, accepted 3 May 2023, date of publication 12 May 2023, date of current version 26 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3275744

APPLIED RESEARCH

DNS Over HTTPS Detection Using Standard Flow Telemetry

KAMIL JERABEK[1], KAREL HYNEK[2,3], ONDREJ RYSAVY[1], AND IVANA BURGETOVA[1]
[1]Faculty of Information Technology, Brno University of Technology, 601 90 Brno, Czech Republic
[2]CESNET z.s.p.o, 160 00 Prague, Czech Republic
[3]Faculty of Information Technology, Czech Technical University in Prague, 166 36 Prague, Czech Republic

Corresponding author: Kamil Jerabek (ijerabek@fit.vutbr.cz)

This work was supported in part by the Ministry of Interior of the Czech Republic (Flow-Based Encrypted Traffic Analysis) under Grant VJ02010024; in part by the Grant Agency of the Czech Technical University in Prague under Grant SGS23/207/OHK3/3T/18; and in part by the Technology Agency of the Czech Republic (Context-Based Encrypted Traffic Analysis Using Flow Data) under Grant FW03010099.

ABSTRACT The aim of DNS over HTTPS (DoH) is to enhance users' privacy by encrypting DNS. However, it also enables adversaries to bypass security mechanisms that rely on inspecting unencrypted DNS. Therefore in some networks, it is crucial to detect and block DoH to maintain security. Unfortunately, DoH is particularly challenging to detect, because it is designed to blend into regular HTTPS traffic. So far, there have been numerous proposals for DoH detection; however, they rely on specialized flow monitoring software that can export complex features that cannot be often computed on the running sequence or suffer from low accuracy. These properties significantly limit their mass deployment into real-world environments. Therefore this study proposes a novel DoH detector that uses IP-based, machine learning, and active probing techniques to detect DoH effectively with standard flow monitoring software. The use of classical flow features also enables its deployment in any network infrastructure with flow-monitoring appliances such as intelligent switches, firewalls, or routers. The proposed approach was tested using lab-created and real-world ISP-based network data and achieved a high classification accuracy of 0.999 and an F1 score of 0.998 with no false positives.

INDEX TERMS DNS over HTTPS, DoH, machine learning, detection, classification, network monitoring, network flows.

I. INTRODUCTION

Domain Name System (DNS) is one of the core elements of the modern internet that allows translation between human-friendly domain names and IP addresses. Since the DNS is transferred unencrypted, anyone on the path can intercept, analyze and misuse the domain names for user surveillance. Government-operated mass DNS surveillance programs such as QUANTUMDNS or MORECOWBELL proved to be very effective [1] and thus triggered concern about users' privacy. Therefore, DNS over HTTPS [2] (DoH) and other encrypted DNS approaches have been proposed as a natural response to protect users' privacy.

The associate editor coordinating the review of this manuscript and approving it for publication was Mahdi Zareei.

On the other hand, mass DNS encryption also brings additional challenges. Visibility into DNS is crucial for many security systems. DNS traffic analysis can reveal many security threats, such as the presence of malware, policy violations, or data exfiltration using DNS tunneling [3]. There are many computer networks, such as corporate networks or restricted institutional networks, where DNS inspection is justifiable and necessary as an additional level of security protection. Detection of encrypted DNS in the network traffic has become essential since it might indicate malicious behavior [4]. Hynek et al. [3] described multiple types of malware that currently leverage DoH to gain its command and control infrastructure stealthfully. Moreover, DoH is also weaponized in various attack vectors by Red Teams.

DoH is widely misused due to its stealthiness in the network. Compared to other encrypted DNS approaches, DoH is designed to blend into regular Hypertext Transfer Protocol Secure (HTTPS) traffic, making its reliable detection challenging. According to Garcia et al. [5], DoH cannot be reliably detected by blocklists of Internet Protocol (IP) addresses and domain names due to multiple small and private DoH resolvers on the internet. Therefore, researchers already use the statistical properties of connections in combination with Machine Learning (ML) to recognize DoH and achieve an accuracy of more than 99%. The high accuracy of these detectors comes with the requirement of additional information about the connections, such as individual packet lengths [6] or median and mode of packet sizes [7], [8]. Extraction of these traffic features is usually not supported by standard network monitoring tools that can be deployed on high-speed networks, which significantly limits the possibility of their deployment.

Additionally, survey [3] also identified the difficulty of detecting single query DoH, which needs to be addressed. Since DoH creates a DNS Application Programming Interface (API) over HTTPS, short connections have very similar characteristics to other HTTPS API requests. Moreover, other no-payload parameters, such as HTTP headers, or the size of the HTTP/2 preface, influence the connection shape much more significantly than the DoH payload itself, making the classifier fit specific server settings [9]. Nevertheless, previous studies did not focus on short DoH connection [3] problems, even though they significantly influence the accuracy of DoH detection in the real world.

Given that none of the existing proposals provide a satisfactory solution for reliable DoH detection in the real environment. Particularly, they use hard-to-compute features and ML-based classifiers with low but still unacceptable false positive rates—with 1000 flows per second, even 99% false positive rate produce in multiple misclassifications per second and possibly overwhelming the security personnel with a high amount of false alarms [10]. Therefore, this study proposes a novel DoH detector that can work with the standard flow data source, making it deployable to almost any flow monitoring infrastructure (from local area networks to high-speed backbone lines) while still maintaining high accuracy over 99.9%. The standard flow statistics can be obtained from all NetFlow capable appliances, such as switches[1], firewalls[2], or specialized flow monitoring probes such as Flowmon[3]. To overcome the information-limited standard flow telemetry data source, a combination of heterogeneous classifiers is used—IP-based, machine learning, and active probing detectors. The industry-proven IP filtering is based on the dynamic list of identified DoH servers, automatically adjusted when the new DoH server is identified in the monitored communication. The ML-based DoH classifier is used to identify candidate DoH servers by performing an online analysis of flow data, which are then confirmed by the active probing detector resulting in almost zero false positives. Moreover, the proposed multi-classifier approach can detect even short DoH connections, which are neglected by existing works.

The contributions of this paper can be summarized as follows:

1) This paper explores the properties of short DoH connections which pose the main limitation of state-of-the-art DoH detectors. The findings were then used for false positives reduction, which is essential for production-grade threat detectors.
2) This paper proposed a novel DoH detector that can correctly operate with standard flow telemetry, making it deployable to almost any monitoring infrastructure ranging from small local area networks to large Internet Service Provider (ISP) based infrastructures connecting millions of devices.
3) This paper proposed a feedback detection system that effectively mitigates the disadvantages of using limited network telemetry data.
4) Compared to previous studies, this paper evaluates the novel detector on the additional independent dataset containing traffic from more than 11 different resolvers, simulating the behavior of the same network making the evaluation more realistic due to significant inconsistencies in DoH resolvers' behavior [9]. Moreover, the evaluation is also made on real-world data captured from a completely different network to analyze the robustness of the approach.

This paper is organized as follows: Section II summarizes related works. Section III provides background knowledge about DoH, flow monitoring, and datasets used for classifier training and overall system evaluation. Section V introduces the concept of the proposed detector. Section VI evaluates the whole detector and presents its performance. Section VII discusses the detection results and its limitations. Finally, Section VIII concludes this article.

II. RELATED WORK

Since its proposal by RFC8484 [2] in 2018, DoH was intensively studied from various perspectives. The survey performed by Hynek et al. [3] summarizes the DoH research and divides it into four main perspectives: 1) DoH performance measurements, 2) Research on DoH adoption, 3) Privacy perspective, and finally, 4) DoH security research part where DoH detection belongs.

The first mention of the necessity of DoH detection was in 2020 by Bumanglag and Kettani [4] in their survey about the impact of mass DoH deployment. In the same year, Bushart and Rossow [21] used a list of DoH resolver IP addresses to recognize DoH in their traffic fingerprinting approach. In addition, Garcia et al. [5] studied the completeness of

[1]https://www.cisco.com/c/dam/en/us/td/docs/security/stealthwatch/netflow/Cisco_NetFlow_Configuration.pdf

[2]https://www.cisco.com/c/en/us/td/docs/security/asa/special/netflow/asa_netflow.html#bidirectionalflows

[3]https://www.flowmon.com/en/products/appliances/probe

TABLE 1. Comparison of related work metrics. The abbrevations stands for: NF - Number of required statistical features; D - Dataset; S - Number of DoH Servers in the dataset; SFE - Short flow elimination before passed to ML model; DoHBrw - *CIRA-CIC-DoHBrw-2020* dataset [11].

Paper	S	NF	Features	Dataset	SFE	Best algorithm	F1	Accuracy
This paper	**12**	**4**	**Standard**	**DoHBrw + Custom [12], [13]**	**< 120 pkts**	**XGBoost**	0.998	99.9%
Vekshin et al. [6]	2	18	Extended	Custom	< 5 payload pkts	Ada-Boosted DT	-	99.6%
MontazeriShatoori et al. [7]	4	28	Extended	DoHBrw	-	Random Forest	0.993	-
Banadaki[a][8]	4	34	Extended	DoHBrw	-	LGBM	-	100%
Jha et al. [14]	2	N/A[c]	Extended	Custom	-	DeepFM	0.995	-
Casanova et al. [15]	4	28	Extended	DoHBrw	-	BiLSTM	0.987[b]	99.0%
Behnke et al. [16]	4	26	Extended	DoHBrw	-	Random Forest	0.998	-
Mitsuhashi et al. [17]	4	28	Extended	DoHBrw	-	XGBoost	0.998	99.8%
Nguyen et al. [18]	4	29	Extended	Custom	-	Transformer NN	0.99	-
Konopa et al. [19]	N/A[c]	3	Standard	Custom	-	Feed Forward NN	-	94.4%
Zebin et al. [20]	4	29	Extended	DoHBrw	-	Balanced Stacked RF	0.999	99.98%

[a] There is concern about general applicability of this detector, since the classifier is fitted on IP addresses from the used dataset.
[b] Score is computed from provided confusion matrix.
[c] Authors do not provide this information.

publicly available DoH blocklists. According to their conclusions, IP-based DoH blocklisting is not currently efficient due to the incompleteness of publicly available blocklists.

One of the first DoH detection approaches that used teletraffic engineering to recognize DoH was proposed by Vekshin et al. [6]. They studied the shape of the DoH traffic using flows extended for information about the first 30 individual packets — packet lengths, packet times, Transmission Control Protocol (TCP) flags, and direction. They extracted 18 discriminatory features from those extended flows that proved efficient in DoH recognition, and their detector achieved an accuracy of 99.6%.

DoH detection was also studied by MontazeriShatoori et al. [7]. They use extended flows with 28 traffic features along with machine learning to distinguish DoH from regular HTTPS traffic. They evaluated multiple machine learning approaches, and the best one achieved an F1 score of 0.993. Moreover, they published the *CIRA-CIC-DoHBrw-2020* [11] dataset used during their study, which became the de-facto standard dataset for DoH detection.

Banadaki [8] designs his method using the *CIRA-CIC-DoHBrw-2020* dataset, which achieves an accuracy of 100%. His approach utilizes machine learning algorithms with time-related features. However, he also uses IP addresses and ports in its feature vectors, which is criticized by Behnke et al. [16]. Behnke et al. then improved the Banadaki proposal by further exploring his feature vector. They removed overfitting features (IP addresses and ports), further reduced statistically insignificant features, and finally measured several trained models. Their improved approach achieved a high F1 of 0.998. Following studies that also used *CIRA-CIC-DoHBrw-2020* dataset made by Casanova and Lin [15], Jha et al. [14], Wu et al. [22], Zebin et al. [20] and Mitsuhashi et al. [17] also performed well in DoH detection tasks achieving over 99% accuracy or 0.99 of F1 score.

The majority of DoH detection studies use *CIRA-CIC-DoHBrw-2020* for evaluation; however, there are also works that created their own dataset. A recent approach by Nguyen and Park [18] proposed a Transformer Neural Network to detect DoH. They used custom datasets and CICFlowmeter to extract 29 features and achieved F1 score of 0.99.

Even though all mentioned studies claimed very high accuracy, there is still a significant limitation preventing their mass deployment. Either they are packet-based or require specialized data sources — flows extended for information about individual packets as in the case of Vekshin et al. [6] or non-standard time-related features as created by DoHLyzer tool and provided within *CIRA-CIC-DoHBrw-2020* dataset or by *CICFlowmeter*, which are then used by several researchers [7], [8], [15], [16], [17], [18], [22]. These approaches require features such as median or mode of packet lengths, which cannot be computed on the running sequence. It means that the monitoring apparatus needs to hold the information about all packets in the memory. The computational and memory complexity prevents its deployment on monitoring infrastructures where hardware resources are limited—router or high-speed monitoring probes. The only approach which relies on standard telemetry data was proposed by Konopa et al. [19], who trained a standard multilayer forward neural network to distinguish between DoH, regular HTTPS, and HTTP. Nevertheless, it achieved only 94.4% accuracy—significantly lower compared to other proposals.

Table 1 summarizes previous flow-based detection methods. Compared to the majority of approaches, the proposed method uses standard telemetry data obtained from all flow-monitoring devices while achieving similar or better accuracy. The flow-monitoring approach is standard and widely deployed. For example, the Steinberger et al. [23] survey showed that 70% of network operators have devices supporting flow export. Moreover, the proposed method uses only features that can be computed on running sequences, making it deployable to any flow monitoring infrastructure, even those monitoring high-speed ISP networks. The only approach that uses standard flow telemetry data is from Konopa et al. [19]. However, its classification performance is relatively low, which prevents its deployment in

practice—one misclassification occurs every 20 flows. The proposed method outperforms Konopa et al. [19] by almost five percentage points. Moreover, we are the first flow-based approach that uses more than ten different DoH resolvers, which is crucial for trustworthy evaluation due to the entirely different behavior of some DoH implementations [9].

III. BACKGROUND

This section contains the necessary background information about DoH protocol and flow-based network monitoring.

A. DOH PROTOCOL

DoH [2] uses HTTPS secure communication to transfer DNS messages protecting the domain name resolution against eavesdropping. The DoH is designed and recommended to be used in cooperation with HTTP/2 [24] to maintain its efficiency due to the possibility of sending multiple concurrent requests. The protocol utilizes the features such as creating multiple streams within a single TCP connection and multiplexing using separated frames to transmit headers and data. Nevertheless, despite the RFC recommendations, some DoH resolvers only support HTTP/1 [5].

DNS messages, either queries or responses, are encoded in DNS wire format [25], which is a compact binary representation where the average size of each message payload is about a hundred bytes [26]. Also, DoH uses only GET and POST methods for queries. DNS queries are transmitted to a resolver in the body of HTTP POST methods encoded using DNS wire format or embedded in the query string of the GET method encoded as Base64. To identify DNS data in the HTTP stream, string *application/dns-message* should be used for *Content-Type* header in all DoH messages and in *Accepts* header field for queries. Except for new headers, possible cache-control options, and the body data format, RFC 8484 does not define any additional specialties that should be included in the application messages. However, the amount of headers is not limited, and DoH clients and servers are free to include more headers in DoH messages as necessary.

The considerable leeway in DoH specification leaves space for many server behavior discrepancies. According to previous research [9], DoH traffic shape significantly depends on the client and server implementation, HTTP headers, and used protocol. Currently, DoH is used by default in all major browsers, such as Chromium[4] based browsers or Mozilla Firefox, for users in the US[5]. Additionally, DoH is already supported on the OS level in all major operating systems[6].

[4]https://blog.chromium.org/2020/05/a-safer-and-more-private-browsing-DoH.html

[5]https://www.zdnet.com/article/mozilla-enables-doh-by-default-for-all-firefox-users-in-the-us/

[6]https://www.inspire2rise.com/how-to-use-dns-over-https-on-windows-android-mac-ios.html

TABLE 2. Flow record features available across multiple Flow Export protocols.

Flow Record
Source IP address
Destination IP address
Source Port
Destination Port
Transport Layer protocol
Number of packets
Number of bytes
Time start
Time end

B. NETWORK MONITORING

The flow-based network monitoring approach is one of the prevalent approaches [27], mainly due to its flexibility and broad support by networking hardware [23]. The diagram of typical flow monitoring infrastructure is shown in the left part of Figure 1 and consists of network probes or network devices capable of flow export distributed throughout the monitored network. The probes observe the ongoing traffic and create statistical flow records, which are then sent to a single collector device, where the analysis and post-processing are performed.

Flows record can be defined as aggregated information about packets transferred through the network that shares some common properties [27]. The common properties are called flow-key and usually consist of IP addresses, ports, and transport protocol. Besides the flow key, there are no standardized features that each flow record needs to carry. Flow traditionally contains the number of transferred bytes and packets [28]. These standard flow features can be extended for information from an application layer (such as HTTP headers, DNS payload, or Transport Layer Security fingerprints) or statistics of the aggregated communication. However, these extended flow fields are not standard and vary widely based on monitoring infrastructure [27].

The flows are transmitted between probes and collectors using flow export protocols. Multiple standard flow export protocols are currently in use [27]—NetFlowV5 [29], NetFlowV9 [30], and IPFIX [31]. While NetFlowV9 and IPFIX support templates, and thus it is possible to customize the flow features highly, NetFlowV5 has fixed records and does not allow the addition of flow features [29]. NetFlowV5 thus highly limits the feature availability; however, according to Hofstede et al. [27], it is still the most used flow export protocol. Table 2 shows features that are considered standard flow features [27] and are supported by all currently used flow export protocols.

According to Hofstede et al. [27], flow records usually describe only unidirectional communication. Thus the bidirectional network communication is split into two records, aggregating packets in a single direction. When necessary for additional analysis, the bidirectional flows can be created from two unidirectional by applying flow stitching on the collector.

TABLE 3. Overall information about used datasets containing the number of used DoH resolvers, number of DoH flows, and the number of Non-DoH flows.

Dataset	Sources	Servers	DoH	Non-DoH
Design dataset	[11], [12]	12	191 340	1 318 676
Evaluation dataset	[13]	16	253 788	1 142 329
Real World dataset	[13]	137	1 591 590	123 050

IV. DATASETS

High-quality datasets are crucial for designing a reliable and accurate network classifier. The de-facto standard dataset created for DoH detection is called *CIRA-CIC-DoHBrw-2020* [11], and it is used by the majority of related works (see Section II). The *CIRA-CIC-DoHBrw-2020* contains DoH and regular HTTPS traffic in the form of extended flow records and pcap files. The traffic, either DoH or regular HTTPS, was generated by Firefox and Chrome browsers querying only four DoH resolvers at varying network speeds.

Nevertheless, the behavior of DoH resolvers and browsers significantly differ as described in previous research [9] (see Section III-A); thus, the four resolvers in *CIRA-CIC-DoHBrw-2020* cover just a fraction of DoH traffic shapes that can be observed on the internet.

To overcome the limited number of DoH resolvers in the *CIRA-CIC-DoHBrw-2020* dataset, the *DoH Network Traffic* dataset [12] is also used. This dataset contains both DoH and web session HTTPS traffic resembling the characteristics of a real-world environment. The HTTPS traffic was generated by accessing web pages taken from the Majestic Million list[7]. The DoH queries are generated against 11 different DoH servers making the dataset more comprehensive.

Packet captures (pcap files) from both *CIRA-CIC-DoHBrw-2020* and *DoH Network Traffic* datasets were merged into one *Design dataset*, which was used for detector design. The packet captures were then processed by NetExp tool[8], aggregating the packets into standard bidirectional flow records.

The whole proposal was evaluated using completely different datasets to make a trustworthy and independent evaluation. For this purpose, the generated and real-world parts from the huge collection of DoH datasets [13] were used. The generated part named Evaluation dataset was created similarly to *CIRA-CIC-DoHBrw-2020* and *DoH Network Traffic*; nevertheless, it was created in a different time and contained traffic towards more DoH servers; thus, its use for evaluation simulates the system's real-world deployment in the same environment. The second part represents capture from a real-world network called Real World dataset which simulates deployment of the method in a different environment and helps evaluate the robustness and sensitivity to data drift. The summary of all used datasets is in Table 3.

[7]https://majestic.com/reports/majestic-million
[8]https://github.com/kjerabek/netexp

Algorithm 1 Flow Records Labelling

Input: *list_of _flows*, *rules*, *classifier*, *length*
Output: setting of *flow.doh*

```
for all flow in list_of _flows do
  if flow.dst_ip is in rules.doh then
    flow.doh ← true
  else if flow.dst_ip is in rules.non_doh then
    flow.doh ← false
  else if flow.length ≤ length then
    flow.doh ← false
  else if not classifier(scaling(features(flow))) then
    flow.doh ← false
  else
    correct_result ← doh_query(flow.dst_ip)
    if correct_result then
      flow.doh ← true
      rules.doh.add(flow.dst_ip)
    else
      flow.doh ← false
      rules.non_doh.add(flow.dst_ip)
    end if
  end if
end for
```

V. THE PROPOSED DOH DETECTOR

Since the detector is constrained to use traditional flow telemetry with the standard features (see Section III-B), it cannot solely rely on a statistical machine-learning method as that would result in a high number of false positives. Instead, the deployment of a heterogeneous detection approach was chosen. The detector utilizes three different types of detection—IP-based, ML-based, and active probing detection.

The detection pipeline uses a feed-forward loop, where a reliable but resource-consuming verification step creates a blocklist/allowlist, which is then utilized by a fast IP-based detector. To limit the number of active probes, an ML-based classifier was deployed, which selects the DoH-suspicious flows that are worth verifying. Using three different classifiers, the proposed approach overcomes each detector limitation: The obsolescence of IP lists is mitigated by active verification and continuous updates, the ML inaccuracy is mitigated by active verification, and the resources needed by active verification are minimized by IP list and ML prefiltration. The processing pipeline of the proposed detection system is described by Algorithm 1 and further depicted in Figure 1 labeled as DoH Detector and can be divided into six following steps:

Flow Stitching is an optional step needed to be deployed into flow monitoring infrastructure with unidirectional flow records. Since the proposed approach requires bidirectional flow records, which are still not widely adopted by flow monitoring infrastructures [27], their reconstruction from two unidirectional records is needed.

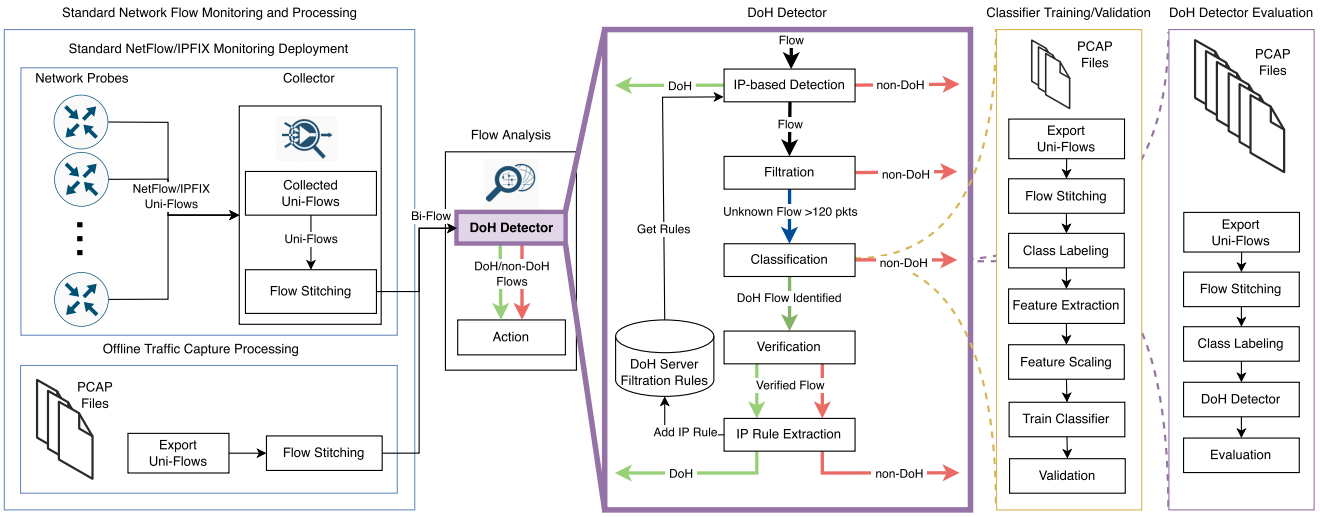

FIGURE 1. Holistic view of the proposed approach with the DoH detector design in context of the monitoring infrastructure.

IP-based Detection uses knowledge acquired from previous IP inspections to recognize DoH resolvers. In the absence of any previous knowledge about the destination IP address, the flow is forwarded to the next step.

Filtration step performs pre-filtration of flows. When the flow is too short for reliable DoH detection, it is directly labeled as non-DoH. Other flows proceed to the following step. Further information about this step is provided in Section V-D.

Classification uses machine learning to detect DoH traffic. Moreover, this step also performs feature extraction. A detailed description of this step is provided in Section V-C.

Verification step uses active DoH queries to the suspected DoH resolver to confirm its DoH resolution capability. When verified, the DoH resolver's IP address is stored in the DoH blocklist, which is then used in the filtration step. This step is further described in Section V-E.

IP Rule Extraction processes the detection results and maintains a blocklist/allow list which is then used by the IP-based detector.

Each component is described in the following sections, their description follows the order as they are depicted in Figure 1 except for the classification and filtration. The necessity of the filtration step arises from the limitations posed by the ML classifier and only standard NetFlow telemetry data. Hence it is described after the classification. Measurement and demonstrations in the following sections are made on the design dataset. The evaluation and real world datasets are held exclusively for the final evaluation of the whole system (see Section VI).

A. FLOW STITCHING

Flow stitching performs the conversion of unidirectional flow records into bidirectional. Flow stitching is a standard process that can reconstruct one bidirectional flow record from two unidirectional records. It is often placed or implemented on the collector. Multiple open-source tools are capable of flow stitching, such as BiFlow Aggregator[9] or Cisco Joy[10]. The features of bidirectional flows created by stitching are shown in Table 4. These flow records are then directly forwarded to the IP-based detection stage.

TABLE 4. Bidirectional flow record after application of flow stitching. The fields denoted in italic are statistical features and thus can be used for detection. Other features are specific to the dataset and should not be used.

Flow Record
Source IP address
Destination IP address
Source Port
Destination Port
Transport Layer protocol
Time start
Time end
Total Number of packets
Total Number of bytes
Number of packets $S \rightarrow C$
Number of bytes $S \rightarrow C$
Number of packets $C \rightarrow S$
Number of bytes $C \rightarrow S$
Duration

Naturally, the flow stitching adds some latency to the detection pipeline depending on the aggregation time window of the stitching tool. The aggregation windows should be set for particular monitoring infrastructure and its overall latency and jitter. Nevertheless, since flows from both directions are going to be exported at a similar time, the flow stitching aggregation time window could be very small in the order of seconds; hence similar latency would be added.

[9]https://github.com/CESNET/Nemea-Modules/tree/master/biflow_aggregator

[10]https://developer.cisco.com/codeexchange/github/repo/cisco/joy

B. IP-BASED DETECTION

The IP-based detection uses previously acquired knowledge about server IP addresses, maintained by IP Rule Extraction and the database, to prevent unnecessary active verification. It directly detects DoH or regular HTTPS by observing the servers' IP address field in the flow. When there is no prior knowledge about the servers' IP address, the flow is forwarded to the next stage.

C. CLASSIFICATION

The classification step uses a machine-learning classifier for an additional selection of DoH-suspicious flows. The description of the classification step is provided before filtration since the ML classifier showed unsatisfactory results (but comparable with previous proposals) when used with short connections and only NetFlow telemetry data. This limitation is then mitigated by the filtration step, which is described afterward. The machine-learning-based classifier relies solely on the statistical properties of the flows to learn the DoH traffic shape. This section describes the process of creating the machine learning model for DoH detection.

1) FEATURE SELECTION

Identifying discriminative features from the incoming flow data is one of the essential tasks during the detector design. The selected features directly influence the detector's performance. To determine features properly, the analysis of the Design dataset was performed. Moreover, the information from previous research [9] has also been used. The following DoH characteristics can be observed that can discriminate DoH from other HTTPS traffic:

Observation 5.1: The DoH, in comparison to non-DoH HTTPS traffic, **transmits fewer data** in requests and responses.

Observation 5.2: The DoH connections last longer, containing many short transactions, creating an **overall higher number of smaller packets** in the flow.

Observation 5.3: DoH in browser **creates multiple streams** over the same HTTP/2 connection and uses multiplexing for faster DNS resolution.

Observation 5.4: The **packet size variance** of DoH **is lower** compared to other HTTPS traffic.

Observation 5.5: The DoH connections are **more symmetrical**, than regular HTTPS in terms of transferred packets and bytes in each direction.

The statistical bidirectional flow feature set is relatively limited, as seen in Table 4. Instead of using features directly, all possible pairs were created by applying division, leaving us with 21 features—ratios of the original features. Consequently, a feature reduction step was performed by calculating the pair-wise Pearson correlation coefficient. The feature reduction removed features that showed near-perfect correlation—their Pearson correlation coefficient was higher than 0.9 [32].

TABLE 5. Selected features together with their Gini importance. The abbreviation stands for: $C \rightarrow S$ - represents Client-to-Server direction, $S \rightarrow C$ - represents Server-to-Client direction, *sc-bytes/sc-packets* represents number of transferred bytes/packets in $S \rightarrow C$ direction, *cs-bytes/cs-packets* represents number of transferred bytes/packets in $C \rightarrow S$ direction.

id	Feature Name	Gini	Formula
1	mean payload size $S \rightarrow C$	0.293	$\frac{sc-bytes}{sc-pkts}$
2	mean time between packets $S \rightarrow C$	0.269	$\frac{sc-pkts}{duration}$
3	mean payload size $C \rightarrow S$	0.247	$\frac{cs-bytes}{cs-packets}$
4	num packets $C \rightarrow S$ to packets ratio	0.191	$\frac{cs-packets}{total-packets}$

The final feature set is listed in Table 5 with the corresponding importance based on the computed Gini index. The final features are also related to traffic observations. The most essential feature #1 and also feature #3 capture the DoH behavior from observations 5.1 and observations 5.2. Feature #4 captures the flow symmetry property from observations 5.5. observations 5.3 is captured by feature #2. Unfortunately, used features cannot describe observations 5.4 since it cannot be computed or approximated from traditional flow data.

2) MACHINE LEARNING CLASSIFIER

Given the features identified in Section V-C1, several classification algorithms were applied to the dataset. Python programming language was employed using Scikit-Learn[11] and XGBoost[12] libraries for learning classification models. Random Forest, K-Nearest Neighbors (KNN—in this case, 4NN was used, which was chosen as the best performing during the hyperparameter tuning phase), Naive Bayes classifiers, and popular boosted algorithms XGBoost and AdaBoosted Decision Trees classifiers were utilized. The detailed description of these ML-based algorithms, which are commonly used in networking tasks [33] is provided by Han et al. [34].

The standard metrics for unbalanced datasets was used to determine the best-performing algorithm for DoH detection. In particular, *precision* (1), *recall* (2), and *F1 measure* (3) were calculated[13]. The primary measure is the *F1 measure* showing the classifier's overall performance. However, the precision metric is also important since practical applications in network monitoring aim to reduce the number of false positives.

$$precision = \frac{TP}{TP + FP} \tag{1}$$

$$recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3}$$

The process of algorithm evaluation followed a standard machine-learning classifier design. First, the Design dataset

[11] https://scikit-learn.org

[12] https://xgboost.ai

[13] Abreviations have following meaning: True Positives (TP), False Positives (FP), False Negatives (FN), True Negatives (TN)

TABLE 6. Performance metrics of classifiers.

Algorithm	F1	Precision	Recall
XGBoost	**0.973**	**0.981**	**0.966**
Ada-boosted DT	0.970	0.978	0.963
KNN	0.963	0.972	0.955
Random Forest	0.955	0.979	0.931
Naive Bayes	0.422	0.304	0.693

TABLE 7. Confusion matrix for XGBoost classifier.

		Predicted	
		DoH	HTTPS
Actual	DoH	46 192	1643
	HTTPS	881	281 609

(see Table 3) was divided into the train (75%) and test (25%) parts using stratified sampling. The train part was used for hyperparameter tuning using cross-validation and training the classifier, and the validation part was reserved for classifier evaluation.

Then the features within each part were standardized using Z-score normalization (also called standardization) since the feature values fall into different ranges. Feature normalization was applied, particularly for features #1, #2, and #3 from Table 5, which contain values ranging from 0 to hundreds.

Each algorithm requires setting proper input parameters, also called hyperparameters. The hyperparameter tuning of all algorithms was done experimentally using a grid-search technique and 5-fold cross-validation on the train part. Models with the best hyperparameters were then trained on the whole train part and evaluated on the test part. Table 6 presents performance results computed using standard metrics numerically. Most algorithms performed well, capable of achieving an F1 score of higher than 0.95, except for the Naive Bayes. Among them, the best performance is achieved by boosting algorithms. The XGBoost algorithm had the best performance and achieved $F1 = 0.973$ with the following hyperparameters: maximum depth of seven and subsample of 0.9.

The detailed performance of the best-performing XGBoost classifier for DoH recognition can be seen in Table 7. Although the F1 of 0.973 can be considered high accuracy, it may still not be accurate enough for deployment on the computer network due to a high number of predictions per second. According to Hofstede et al. [35], the Czech national research and educational network create up to 10 000 flows per second—the same number of predictions per second would be made by the classifier when deployed on this network. Therefore, the classifier would produce around 2.6 false positive[14] detections per second (9360 false positive detections per hour). Generating such a large number of misclassifications would overwhelm the administrators and the system, causing such detection not to be beneficial. Hence,

[14]FP/(FP+FN+TP+TN) = 0.0026

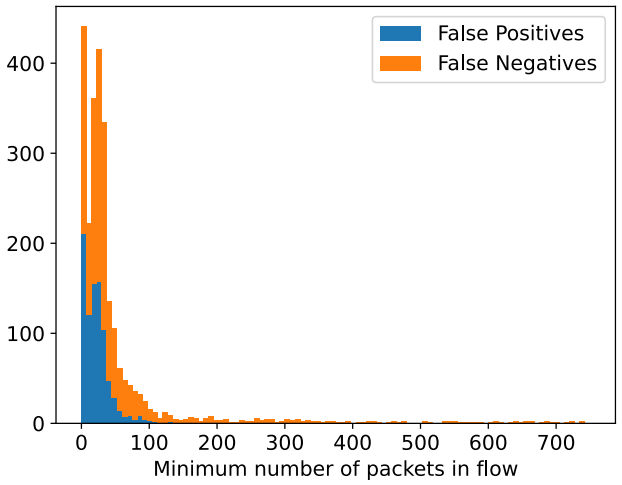

FIGURE 2. Stacked histogram of misclassified flows.

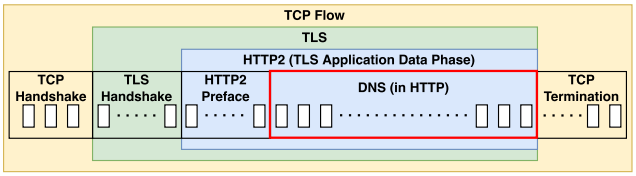

FIGURE 3. Schema of HTTPS (DoH) flow with all connection prefaces highlighted.

there is a need to push the detection system toward more accurate detection.

Further investigation revealed that short flows are the main cause of reduced accuracy. Figure 2 shows the number of false positive detections depending on the number of packets in flows. It can be seen that misclassification happens more often when flows have less than 100 packets. To reduce false positives, the elimination of shorter flows is needed since those cannot be reliably recognized by the machine-learning classifier that uses standard flow information.

D. FILTRATION

The DoH connection (similarly to other HTTPS connections) involves a TCP handshake, Transport Layer Security (TLS) handshake, HTTP/2 preface, application data transfer phase, and TCP termination, as shown in Figure 3. Each phase amounts to several packets exchange. Based on the measurement [9], it was determined that one DNS request and response over HTTP/2 requires (on average) 28 packets (HTTP/1 requires on average only 17 packets[15]), with only two packets carrying the actual DNS payload.

The shorter flows might not contain any DoH queries at all. Browsers usually create multiple connections to the target server to optimize the loading process. The browser contacts

[15]Lower number of packets is caused by missing novel HTTP/2 features such as connection preface, multiplexing and HTTP header and payload transmission separation.

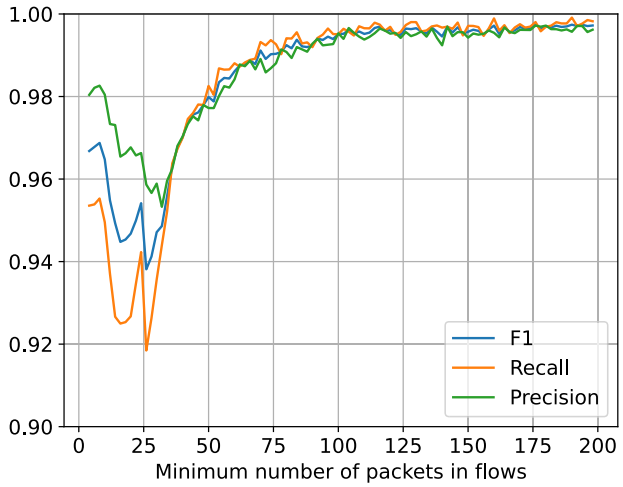

FIGURE 4. Minimum number of packets in flow hyperparameter tuning.

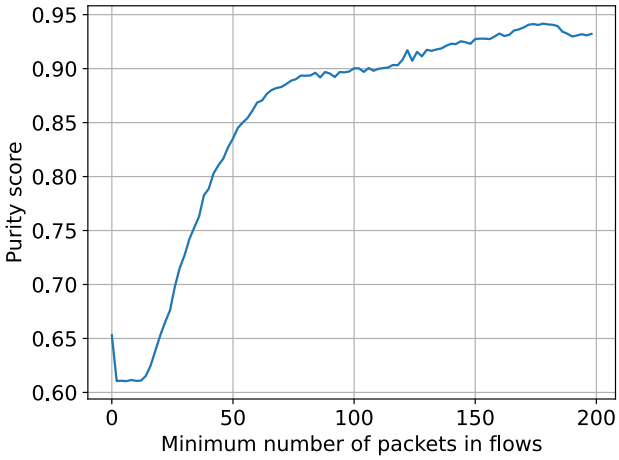

FIGURE 5. Purity score of the converged clusters based on minimum packets in flows.

the server with several connections—at least one connection is then used, and the rest are failed or terminated[16]. Such connections do not transmit any application data.

Hynek et al. [3] mentioned short DoH connections as challenging. Particularly the single-query DoH produces flows with similar statistics as other short API calls. The influence of many HTTP/2 preface packets compared to the small portion of DoH data is high and complicates reliable DoH detection. Therefore, the determination of the minimal number of packets in the flow carrying enough DoH packets is needed to recognize DoH from other HTTPS traffic reliably.

1) DETERMINING THE MINIMAL NUMBER OF PACKETS THRESHOLD

There are multiple ways to determine the minimal number of packets in flow needed for reliably distinguishing DoH from regular HTTPS traffic. One of the approaches would be to add the threshold to hyperparameters and find the best value during the hyperparameter tuning phase. Another way would be to use an unsupervised clustering method to group samples automatically and measure the ability to separate samples into two classes by measuring the intra-class similarity. Both approaches were used.

At first, the experiments with threshold tuning were performed. The best-performing XGBoost algorithm was used, and the minimum packet threshold was incrementally increased. The model was retrained each time the threshold was raised. As expected, the model's accuracy increases with a higher threshold, as shown in Figure 4. The model reaches first stable performance around 116 packets and then maintains similar accuracy.

In the second approach, K-Means [34] clustering method was used with k-means++ initialization to determine the minimal packet lengths. It was done by using the algorithm's ability to group samples into a predefined number of k clusters (in this case $k = 2$, the same as the number of classes) based on their similarity without the need to label the samples. Applying the algorithm to all flows without filtration would lead to a different capability of distinguishing the samples reliably than when applied to filtered flows.

The dataset without filtration was clustered and then the required number of packets in flows was increased. The intra-class similarity was measured by calculating the purity score of the resulting clusters. Since the dataset is highly imbalanced in favor of the non-DoH class, random undersampling was applied before each clustering to balance the dataset and correctly compute the purity score. Undersampling rather than over-sampling was chosen since there were enough data to work with and it works only with real data records without the need for any data augmentation.

The results are depicted in Figure 5. It can be seen that the minimal number of packets in flow highly influences the DoH recognition capability of the clustering method. More packets in flows make the recognition more reliable, which is consistent with previous observations. The 90% of purity index of both clusters is reached when flows have at least 112 packets.

In addition, the estimation of the average number of packets in the DoH connection needed for fetching one website was performed. The DoH connections of the top 10 000 visited websites from the *Majestic Million* dataset were investigated. The average number of unique domain references by these websites was 20, leading to an equal number of DNS resolutions. Measured in terms of packets, this gives roughly 120 packets in a DoH flow corresponding to one website visit.

All experiments showed satisfactory DoH detection performance with similar values ranging from 112-116 packets, while flows with more packets are generally more reliably recognized. It was decided to set the filtration threshold

[16]The number of parallel connections created by a browser is not specific for DoH and can often be configured [36].

TABLE 8. Performance metrics of classifiers on reduced dataset—it contains only flows with ≥120 packets (See Table 6 for comparison.).

Algorithm	F1	Precision	Recall
XGBoost	**0.996**	**0.995**	**0.997**
Random Forest	0.995	0.993	0.997
Ada-boosted DT	0.994	0.991	0.996
KNN	0.993	0.991	0.996
Naive Bayes	0.927	0.887	0.971

TABLE 9. Confusion matrix for XGBoost classifier evaluated on the reduced dataset—it contains only flows with ≥120 packets.

		Predicted	
		DoH	**HTTPS**
Actual	**DoH**	4169	13
	HTTPS	20	30 786

to ≥120 packets since such a number of DoH packets is generated during an average single web page visit. According to Crichton et al. study [37], users mostly visit more than one site in a short period and spend on average 1.7 hours browsing in 35.9 browser tabs daily. Even when considering DNS caching mechanisms, there is a high probability of generating longer DoH flows than the estimated 120 packets. In practice, the threshold thus could be set to even higher numbers of packets in a flow, which would improve the model accuracy even more.

2) FINAL CLASSIFIER EVALUATION AFTER FILTRATION

The final experiments of the detection module accuracy were performed on the flows that fulfilled the requirement for the minimal number of packets—the flows with less than 120 packets were filtered out. The filtration reduced the number of DoH flows to 16 728 (8.74% of the original dataset) and 123 222 (9.34%) non-DoH flows. Despite a more than 90% reduction in flows, the number of unique DoH servers and their IPs remained the same.

The accuracy of classifiers reached on the reduced dataset is shown in Table 8. The classification methodology remained unchanged, with the data split into the train (75%) and the test part (25%) using stratified sampling. Nevertheless, hyperparameter tuning was not performed and the same hyperparameters as in the previous evaluation (described in Section V-C2) were used. The best performing algorithm was again the XGBoost, achieving F1 of 0.996.

The detailed results of the XGBoost algorithm are provided in Table 9. It can be seen that the number of false positives is significantly reduced. Since the number of misclassifications is relatively low, they could be investigated manually. The 20 false positives were generated mainly by telemetry and analytical services embedded in the websites by advertisement. These types of flows show similar traffic shape characteristics as DoH, which justifies their misclassifications. The false negatives can be identified as anomalous, with at least one feature value standing out or at the edge of the decision boundary.

E. VERIFICATION

The next step in the DoH detection pipeline is active server verification. Active verification is the most accurate way used in the past by multiple researchers [22], [38].

The active probing module generates DoH requests for `example.com` domain to `/dns-query` endpoint via both HTTP GET and HTTP POST methods, as specified in [2]. The servers hosted on shared infrastructures usually require a domain name for a successful connection. Therefore, the active verification can use a domain name from TLS handshake when extracted by monitoring appliances, available passive DNS services (providing IP-domain mappings), or direct IP queries. Passive DNS and direct IP queries are the only options in this case where the lack of extracted domain name is present.

The active probing approach always comes with ethical consideration and the correct setting of the probing rate. When using an aggressive rate, the source IP address could be marked as malicious and blocked by service providers resulting in reduced verification efficiency. Moreover, active requests always consume some target resources. Nevertheless, in this case, the HTTPS servers are probed; thus, they should be scaled adequately for HTTP requests. Moreover, by storing even non-DoH server IP addresses, it was ensured that the active probing module checks each IP address/domain only once in 24 hours and thus limiting the target resource utilization to a minimum.

F. IP RULE EXTRACTION

The verification results are then processed, and the servers' IP addresses are extracted and stored in the database IP DoH Server Filtration Rules, as seen in Figure 1. Both results, the confirmed DoH and confirmed non-DoH servers, are stored in the database and used for direct detection in the IP-based detection step. The database then limits the amount of ML-based detection and verification, thus increasing the whole system's performance.

As Gracia et al. [5] discussed, the detection of DoH by blocklists is inefficient due to fast blocklist obsolescence. Consequently, the non-DoH addresses are stored only for 24 hours, and the DoH resolvers are regularly checked (every 24 hours) to maintain the timeliness of the stored information.

VI. EXPERIMENTAL RESULTS

The whole proposed DoH detector depicted in Figure 1 was evaluated using a separate Evaluation dataset (see Section IV). Using an utterly separate dataset captured in the same network environment with different properties not seen during the design phase is similar to actual deployment, with the benefits of the ground truth labels. Thus, the correctness of the evaluation performance can be guaranteed.

The evaluation simulated the deployment of the system in the same network environment. The dataset pcap files were processed with the NetExp tool (as in the case of the Design dataset). Since the NetExp creates bidirectional

TABLE 10. Confusion matrix representing the final evaluation of the system.

		Predicted	
		DoH	HTTPS
Actual	DoH	252 641	1147
	HTTPS	0	1 142 329

flow records, its extraction also represents a flow stitching phase. The flows were processed in the original order by the detection pipeline. The XGBoost classifier was used as proposed in Section V-D2, which was trained on the train part of the Design dataset. Since the Evaluation dataset is more than one year old, active verification was not performed; instead, the ground truth labels provided within the dataset were used. The evaluation process was set to classify one flow at a time without any parallelization and other performance optimization techniques.

The final evaluation dataset contained 254 788 DoH flows and 1 142 329 non-DoH flows. Nevertheless, there are only 25 078 DoH and 137 327 non-DoH flows with more than 120 packets that would pass the filtration step. The confusion matrix of the overall system evaluation is shown in Table 10. The system achieved F1 of 0.998 with a precision of 1.0 and a recall of 0.995. No false positives have been observed, which is expected due to the verification stage. However, there are 1147 false-negative flows.

Despite the false negatives, the system was capable of identifying all DoH servers' IP addresses in the dataset—thus, the created blocklist was complete. All false negatives occurred due to the latency of the DoH server identification. From 16 different resolvers, 15 were selected by ML for verification during the first occurrence of flow with more than 120 packets, the missing one was selected after several occurrences. However, almost all false negatives (except one) were then created by the filtration module. Before the first flow with more than 120 packets, the filtration module automatically marked shorter connections as non-DoH. On average, the system falsely marked nine short DoH flows before a longer DoH flow was received and successfully identified as DoH.

Overall, the dataset contained flows toward 76 870 unique IP addresses. The system labeled 23 unique DoH server IP addresses (some of the DoH resolvers had more than one address) and allowlisted another 87 IP addresses as false positives. It is worth noting that those false positives created by ML are expected and handled by the Verification and IP-based Detection stages. Thus, the whole detection system will not raise false alarms. In total, only 110 flows were forwarded to the verification stage. Assuming a large network with 10 000 flows per second, the Evaluation dataset represents 139.7 seconds of traffic, meaning the active verification would create ∼0.7 active requests per second, which can be considered very low and acceptable for real-world deployment.

A. EVALUATION OF THE DATA DRIFT

The network traffic is often susceptible to data drift [39] (sometimes called concept drift)—a phenomenon in which the underlying distribution of the data changes in time due to novel services or updates of the network infrastructure. Thus, the robustness of the novel method to data drift was evaluated. The Real World traffic captured in real backbone network [13] was used for that. This traffic was captured a year apart from the design data on an entirely different network setup. We used the proposed detector with ML trained on the design dataset and evaluated it on the Real World traffic dataset with the same methodology as in the previous section. The performance results are shown in Table 11.

TABLE 11. Confusion matrix for data drift susceptibility of the system.

		Predicted	
		DoH	HTTPS
Actual	DoH	1 549 380	42 210
	HTTPS	0	123 050

The proposed detector achieved an F1 score of 0.987 and an accuracy of 97.5%. Given the different training and testing network environments and the year gap between training data and testing data, which might be considered an extreme case, the detector showed stable performance with only ∼ 0.01 of F1 score drop and 2.4 percentage point drop in accuracy. For comparison, other work also dealing with encrypted network traffic and data drift of Malekghaini et al. [39] experienced a drop of up to 40 percentage points when trained and evaluated on year-apart data.

VII. DISCUSSION

The proposed system performs similarly or better than the other related works (see Table 1), while trained and validated on a more comprehensive DoH dataset containing traffic towards more DoH resolvers including real-world traffic where different resolvers and their configuration highly influences the DoH traffic shape [9]. Due to the rich dataset and thorough analysis, this is the first work that points out the short DoH flow phenomena and designs the detection pipeline accordingly. Moreover, the majority of previous works suffer from reliance on specialized hard-to-obtain features. The tailored features improve the accuracy, but they also limit mass deployment due to the necessity of installing specialized network probes.

Our proposal works with easy-to-obtain features available in almost all network monitoring infrastructures, even those with NetFlowV5. The only related work that also works with features extractable from traditional NetFlow is the Konopa et al. [19], who achieved an accuracy of 94.4% in DoH detection. Compared to them, the proposed system achieved 99.9% of accuracy while evaluated on the biggest DoH dataset available [13]. The proposed system also addresses the desirable elimination of false positives. The increased precision is due to the use of three heterogeneous classifiers, each of which has advantages and limitations that together create a more robust and precise DoH detector.

While the goal was to minimize false positives, the detector can still produce some when DoH and non-DoH services are both hosted behind a single IP address. Such cases cannot be

distinguished by IP addresses but by domain names. Since the majority of flow monitoring infrastructures support only NetFlowV5 [27] and do not support the extraction of domain names from TLS handshakes, it might be considered a limitation, even though it is caused by the underlying network monitoring infrastructure. When extraction of domain names is supported, modification of the presented architecture is trivial (block list would use domain names instead of IP addresses), and the false positives would be mitigated.

The detection system also suffers from false negatives, mainly caused by the detection latency. Nonetheless, the majority of false negatives could be mitigated by back-labeling of flows automatically labeled as non-DoH by the filter. It would require additional temporal storage of short flows that would wait for at least several seconds for the potential labels because a long one often follows short DoH flows (as discussed in Section V-D).

Furthermore, skilled adversaries can bypass detection by limiting the number of packets in their DoH connections. Nevertheless, such packet limitation would significantly impact the DoH performance. Moreover, DoH only encrypts DNS, and more skilled adversaries would rather use other stealthy communication, such as HTTPS-based VPN or other multipurpose tunneling tools.

Despite all mentioned limitations (mainly caused by the limited availability of traffic features), the proposed DoH detector proved to be accurate and also computationally efficient. Most work is done using simple list-based filtering and classification using a pre-computed model. Only a fraction of inputs requires verification through active probing. Moreover, the detector's input consists of standard NetFlow records available on various network devices.

VIII. CONCLUSION

Detection of DNS over HTTPS protocol is a novel but an essential requirement in highly restricted or private networks, where DNS inspection is critical for maintaining security. Lack of DoH detection severely decreases situational awareness and gives attackers and users a powerful and easy-to-use approach for bypassing security policies and network intrusion detection systems. So far, the DoH detection proposals used traffic features that only highly specialized monitoring devices could extract. The use of already deployed flow monitoring appliances supporting widely-used NetFlowV5 was not concerned by state-of-the-art detectors. Therefore, this study proposes a novel high-accurate DoH detector that can be directly deployed on NetFlowV5 (and more recent) flow monitoring infrastructures.

Three heterogeneous detection approaches (IP-based, Machine Learning, and active probing) were used to overcome the challenges posed by the limited amount of information in standard flow data and achieved similar or better accuracy than previous proposals. The validation used an evaluation dataset, which was different from the dataset used during the design, thus simulating the deployment in the same environment. Moreover, the real-world dataset from a completely different network captured at different time was used to test the robustness of the approach.

The proposed DoH detection achieved an excellent performance, with no false positives on the evaluation dataset, only slight degradation on the real-world dataset, and can thus be deployed on even large monitoring infrastructures, potentially detecting DoH from millions of devices. Moreover, the proposed feed-forward detection loop can be used in other use cases, not limited to DoH only.

REFERENCES

[1] C. Grothoff, M. Wachs, M. Ermert, and J. Appelbaum, "Toward secure name resolution on the Internet," *Comput. Secur.*, vol. 77, pp. 694–708, Aug. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404818300403

[2] P. E. Hoffman and P. McManus, *DNS Queries Over HTTPS (DoH)*, document RFC 8484, Oct. 2018.

[3] K. Hynek, D. Vekshin, J. Luxemburk, T. Cejka, and A. Wasicek, "Summary of DNS over HTTPS abuse," *IEEE Access*, vol. 10, pp. 54668–54680, 2022.

[4] K. Bumanglag and H. Kettani, "On the impact of DNS over HTTPS paradigm on cyber systems," in *Proc. 3rd Int. Conf. Inf. Comput. Technol. (ICICT)*, Mar. 2020, pp. 494–499.

[5] S. García, J. Bogado, K. Hynek, D. Vekshin, T. Čejka, and A. Wasicek, "Large scale analysis of DoH deployment on the Internet," in *Computer Security—ESORICS*, V. Atluri, R. Di Pietro, C. D. Jensen, and W. Meng, Eds. Cham, Switzerland: Springer, 2022, pp. 145–165.

[6] D. Vekshin, K. Hynek, and T. Cejka, "DoH insight: Detecting DNS over HTTPS by machine learning," in *Proc. 15th Int. Conf. Availability, Rel. Secur.*, Aug. 2020, pp. 1–8.

[7] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proc. IEEE Int. Conf. Dependable, Autonomic Secure Comput., Int. Conf. Pervasive Intell. Comput., Int. Conf. Cloud Big Data Comput., Int. Conf. Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech)*, Aug. 2020, pp. 63–70.

[8] Y. M. Banadaki, "Detecting malicious DNS over HTTPS traffic in domain name system using machine learning classifiers," *J. Comput. Sci. Appl.*, vol. 8, no. 2, pp. 46–55, Aug. 2020.

[9] K. Jerabek, O. Rysavy, and I. Burgetova, "Measurement and characterization of DNS over HTTPS traffic," 2022, *arXiv:2204.03975*, doi: 10.48550/ARXIV.2204.03975.

[10] B. Alahmadi, L. Axon, and I. Martinovic, "99% false positives: A qualitative study of {SOC} analysts' perspectives on security alarms," in *Proc. 31st USENIX Secur. Symp. (USENIX Secur.)*, Aug. 2022. Accessed: Dec. 16, 2022. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi

[11] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, *CIRA-CIC-DoHBrw-2020*. Accessed: May 2022. [Online]. Available: https://www.unb.ca/cic/datasets/dohbrw-2020.html

[12] K. Jeřábek and S. Stuchlý, "DNS Over HTTPS network traffic," IEEE Dataport, 2021. [Online]. Available: https://ieee-dataport.org/documents/dns-over-https-network-traffic, doi: 10.21227/96ea-2055.

[13] K. Jeřábek, K. Hynek, T. Čejka, and O. Ryšavý, "Collection of datasets with DNS over HTTPS traffic," *Data Brief*, vol. 42, Jun. 2022, Art. no. 108310.

[14] H. Jha, I. Patel, G. Li, A. K. Cherukuri, and S. Thaseen, "Detection of tunneling in DNS over HTTPS," in *Proc. 7th Int. Conf. Signal Process. Commun. (ICSC)*, Nov. 2021, pp. 42–47.

[15] L. F. G. Casanova and P. Lin, "Generalized classification of DNS over HTTPS traffic with deep learning," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2021, pp. 1903–1907.

[16] M. Behnke, N. Briner, D. Cullen, K. Schwerdtfeger, J. Warren, R. Basnet, and T. Doleck, "Feature engineering and machine learning model comparison for malicious activity detection in the DNS-over-HTTPS protocol," *IEEE Access*, vol. 9, pp. 129902–129916, 2021.

[17] R. Mitsuhashi, Y. Jin, K. Iida, T. Shinagawa, and Y. Takai, "Detection of DGA-based malware communications from DoH traffic using machine learning analysis," in *Proc. IEEE 20th Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2023, pp. 224–229.

[18] T. A. Nguyen and M. Park, "DoH tunneling detection system for enterprise network using deep learning technique," *Appl. Sci.*, vol. 12, no. 5, p. 2416, Feb. 2022, doi: 10.3390/app12052416.
[19] M. Konopa, J. Fesl, J. Jelínek, M. Feslová, J. Cehák, J. Janeček, and F. Drdák, "Using machine learning for DNS over HTTPS detection," in *Proc. 19th Eur. Conf. Cyber Warfare Secur.*, 2020, p. 205.
[20] T. Zebin, S. Rezvy, and Y. Luo, "An explainable AI-based intrusion detection system for DNS over HTTPS (DoH) attacks," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 2339–2349, 2022.
[21] J. Bushart and C. Rossow, "Padding ain't enough: Assessing the privacy guarantees of encrypted DNS," in *Proc. 10th USENIX Workshop Free Open Commun. Internet (FOCI)*. USENIX Association, Aug. 2020, pp. 1–8. [Online]. Available: https://www.usenix.org/conference/foci20/presentation/bushart
[22] J. Wu, Y. Zhu, B. Li, Q. Liu, and B. Fang, "Peek inside the encrypted world: Autoencoder-based detection of DoH resolvers," in *Proc. IEEE 20th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Oct. 2021, pp. 783–790.
[23] J. Steinberger, L. Schehlmann, S. Abt, and H. Baier, "Anomaly detection and mitigation at Internet scale: A survey," in *Proc. AIMS*. Berlin, Germany: Springer-Verlag, 2013, pp. 49–60.
[24] M. Belshe, R. Peon, and M. Thomson, *Hypertext Transfer Protocol Version 2 (HTTP/2)*, document RFC 7540, May 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7540
[25] P. Mockapetris, *Domain Names—Implementation and Specification*, document RFC 1035, Nov. 1987. [Online]. Available: https://www.rfc-editor.org/info/rfc1035
[26] H. Federrath, K.-P. Fuchs, D. Herrmann, and C. Piosecny, "Privacy-preserving DNS: Analysis of broadcast, range queries and mix-based protection methods," in *Computer Security—ESORICS*, V. Atluri and C. Diaz, Eds. Berlin, Germany: Springer, 2011, pp. 665–683.
[27] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, "Flow monitoring explained: From packet capture to data analysis with NetFlow and IPFIX," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 2037–2064, 4th Quart., 2014.
[28] J. Luxemburk and T. Čejka, "Fine-grained TLS services classification with reject option," *Comput. Netw.*, vol. 220, Jan. 2023, Art. no. 109467, doi: 10.1016/j.comnet.2022.109467.
[29] Cisco Systems. (Sep. 2007). *NetFlow Export Datagram Format*. Accessed: Dec. 2022. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/3-6/user/guide/format.html
[30] B. Claise, *Cisco Systems NetFlow Services Export Version 9*, document RFC 3954, Oct. 2004, doi: 10.17487/RFC3954.
[31] P. Aitken, B. Claise, and B. Trammell, *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information*, document RFC 7011, Sep. 2013. [Online]. Available: https://www.rfc-editor.org/info/rfc7011
[32] S. Sabilla, R. Sarno, and K. Triyana, "Optimizing threshold using Pearson correlation for selecting features of electronic nose signals," *Int. J. Intell. Eng. Syst.*, vol. 12, no. 6, pp. 81–90, Dec. 2019.
[33] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities," *J. Internet Services Appl.*, vol. 9, no. 1, pp. 1–99, Jun. 2018, doi: 10.1186/s13174-018-0087-2.
[34] J. Han, J. Pei, and H. Tong, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2022.
[35] R. Hofstede, V. Bartos, A. Sperotto, and A. Pras, "Towards real-time intrusion detection for NetFlow and IPFIX," in *Proc. 9th Int. Conf. Netw. Service Manage. (CNSM)*, Oct. 2013, pp. 227–234.
[36] P. G. Smith, *Professional Website Performance: Optimizing the Front-End and Back-End*. Hoboken, NJ, USA: Wiley, 2012.
[37] K. Crichton, N. Christin, and L. F. Cranor, "How do home computer users browse the Web?" *ACM Trans. Web*, vol. 16, no. 1, pp. 1–27, Feb. 2022.
[38] V. Veselý and M. Žádník, "How to detect cryptocurrency miners? By traffic forensics!" *Digit. Invest.*, vol. 31, Dec. 2019, Art. no. 100884. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287618303359
[39] N. Malekghaini, E. Akbari, M. A. Salahuddin, N. Limam, R. Boutaba, B. Mathieu, S. Moteau, and S. Tuffin, "Deep learning for encrypted traffic classification in the face of data drift: An empirical study," *Comput. Netw.*, vol. 225, Apr. 2023, Art. no. 109648. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128623000932

KAMIL JERABEK is currently pursuing the Ph.D. degree with the Faculty of Information Technology, Brno University of Technology. He is a part of the Network and Distributed Systems Research Group. He has participated in several research projects in national and international. His research interests include computer networks, cyber security, big data, data analysis, and machine learning.

KAREL HYNEK is currently pursuing the Ph.D. degree with the Faculty of Information Technology, Czech Technical University in Prague, with a focus on network security and network monitoring. He has worked on several national and international research projects related to security. He is a Key Member of the Network Traffic Monitoring Research Team, Faculty of Information Technology.

ONDREJ RYSAVY received the Ph.D. degree in computer science from the Brno University of Technology, Brno, Czech Republic, in 2005. He is currently an Associate Professor with the Department of Information Systems, Brno University of Technology. His work is focused on improving network security through data analysis by application of data mining, statistics, and distributed computing. His research interests include computer networking, particularly network monitoring, network security and forensics, and network architectures.

IVANA BURGETOVA received the Ph.D. degree in computer science from the Brno University of Technology, in 2009. Currently, she is a part of the Network and Distributed Systems Research Group and the Information and Database Systems Research Group. Her research interests include data mining, data processing, and bioinformatics.