International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 6, December 2023, pp. 6691~6700
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i6.pp6691-6700

A novel deep-learning based approach to DNS over HTTPS network traffic detection

Jan Fesl, Michal Konopa, Jiří Jelínek
Department of Informatics, Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic

Article Info

Article history:

Received Oct 12, 2022
Revised Jun 4, 2023
Accepted Jun 26, 2023

Keywords:

Computer networks
Detection
DNS over HTTPS
Machine learning
Traffic detection

ABSTRACT

Domain name system (DNS) over hypertext transfer protocol secure (HTTPS) (DoH) is currently a new standard for secure communication between DNS servers and end-users. Secure sockets layer (SSL)/transport layer security (TLS) encryption should guarantee the user a high level of privacy regarding the impossibility of data content decryption and protocol identification. Our team created a DoH data set from captured real network traffic and proposed novel deep-learning-based detection models allowing encrypted DoH traffic identification. Our detection models were trained on the network traffic from the Czech top-level domain maintainer, Czech network interchange center (CZ.NIC), and successfully applied to the identification of the DoH traffic from Cloudflare. The reached detection model accuracy was near 95%, and it is clear that the encryption does not prohibit the DoH protocol identification.

Corresponding Author:

Jan Fesl
Department of Informatics, Faculty of Science, University of South Bohemia
Branišovská 31a, České Budějovice, 37001, Czech Republic
Email: jfesl@prf.jcu.cz

1. INTRODUCTION

The usage of the domain name system (DNS) over hypertext transfer protocol secure (HTTPS) (DoH) protocol is currently supported by the most commonly used web browsers. In DoH, the DNS requests/responses [1] are encapsulated in the data of the HTTPS protocol [2]. This approach allows DoH to be easily forwarded through network firewalls or Internet routers. The content of the requests or responses is impossible to identify due to encryption.

Internet service providers (ISPs) commonly use their DNS servers for domain monitoring which are accessed via HTTPS protocol. This is suitable because HTTPS encryption transport layer security (TLS) can since the used TLS version 1.3 completely hide the accessed domain name in the packet header. When the users prefer in their web browsers the usage of DoH protocol, there is practically no possibility to detect which domains are accessed by them. For a specific potentially dangerous user, it can be helpful to mark or drop the DoH traffic packets and force the user to use a normal unencrypted version of the DNS protocol. The robust, fast and reliable DoH packet detection within a huge volume of common network traffic is a non-trivial task.

In our work, we first created a specific DoH dataset retrieved from real DoH network traffic coming from DoH servers placed in the Czech network interchange center (CZ.NIC). Then we identified a specific DoH network pattern, resp. created the two deep-learning-based neural network models. The created models were then used to identify unknown DoH traffic coming from Cloudflare network services. We have performed many experiments due to the creation of machine-learning models able to detect the DoH traffic within normal network traffic. Our final-version convolutional models were applied to the identification of unknown DoH network traffic coming from Cloudflare and achieved an accuracy near 95%. Our paper consists of 4 following sections: i) Current state-of-the-art: this section contains the DoH protocol introduction and a detailed overview of previously created machine-learning models used for DoH detection; ii) Our approach and proposed solution definition: this section contains the algorithm used for captured DoH data preprocessing and there are described the dense and convolutional machine-learning models used for DoH detection and their hyperparameter optimization; iii) Results: we introduced a new data set used for the training of the DoH detection models and there is an overview of the achieved accuracy of the models introduced in the previous section; and iv) Conclusion: there are summarized results of our work and proposed possible new directions for future work.

2. CURRENT STATE-OF-THE-ART

Because DoH is a relatively new technology, there are few papers regarding this topic. The complete source we have found on this topic is [3]. The author mentions the following detection options: i) TLS inspection, ii) profiling of encrypted traffic, iii) checking connections to servers from the list of DoH providers (frequent updates are needed), iv) whitelist of network applications and their proper configurations, and v) statistical processing of metadata (e.g., number of bytes transmitted, connection length, and average packet length).

According to the author, with the introduction of TLS 1.3, where certificates are encrypted and therefore not transmitted in plaintext, as in earlier versions, TLS inspection is much more difficult. Another extension of TLS 1.3 encrypts the SNI field so that it is no longer possible to "look out" for the domain name from the TLS handshake. More information about SNI encryption can be found in Rescorla *et al.* [4].

Another possibility is to use artificial intelligence methods. A very interesting paper on DoH detection using machine learning techniques is [5]. The authors focused on two things: i) simple detection of DoH traffic and ii) identification of a concrete web browser involved in DoH communication. The features mainly concerned statistics of communication delays, sizes of transmitted data and their ratios within one flow. It should be noted that the feature vector did not contain any data on IP addresses and ports, which could be used to easily identify the DoH server and thus DoH traffic. To train and test selected machine learning (ML) models (5-NN, C4.5, random forest, naive Bayes (NB), Ada-boosted decision tree DT), the authors created and then published their dataset, including DoH traffic from currently most used web browsers, Mozilla Firefox and Google Chrome. The results were excellent both in the case of simple DoH detection and in the case of identification of a concrete web browser. An interesting experimental finding of the authors was that the feature that most distinguishes DoH traffic from normal HTTPS traffic is the flow duration, which is much longer in the case of DoH.

Bushart and Rossow [6] and Wang *et al.* [7] investigated the possibility of recognizing websites visited via DoH with DNS message padding (RFC 8467). They created a mechanism called DNS Sequences, which describes the time sequence of DNS response sizes and gaps (in millisecond log scale) between responses when visiting a particular website. They used the k-nearest neighbors (k-NN) classifier to classify DNS-Sequences into websites. They used 10,000 websites from the Tranco list for learning and testing. The presented results show that a classifier would be able to classify about 80% of websites with 90% success (9 out of 10 samples). The authors also suggested possible countermeasures that should radically reduce the success of the recognition of visited websites.

Hynek and Cejka [8] evaluated the possibilities of inferring individual domain names from encrypted DoH connections. The authors based their solution on 2 findings: i) For each website load, it is possible to observe multiple DNS packet bursts - as each loaded asset might have other dependencies and ii) Although the order of DNS responses can be shuffled, DNS packet sizes remains almost unchanged in one web page load. For each response, they defined 3 neighbor zones according to the time distance from the response (time distances were stated experimentally). For each zone separately, they then measured statistics (min, max, mean, median, variance) of the size of the DoH responses that were in the zone. From these statistics, they finally selected a total of 11 as a feature vector. They experimented with several ML models, the best results were achieved using the combination of the AdaBoosted decision tree and the Bagging meta-learning algorithm. According to the authors, their classifier can infer domain names with an accuracy up to 90% on HTTP 1.1 and up to 70% on HTTP 2 protocol.

Classification of DoH traffic into benign or malicious classes was the main goal of the work [9]. The authors created two ensemble learning classifiers. The first one consisted of decision tree (DT), logistic regression (LR) and k-NN, the second one was the RF classifier. The CIRA-CIC-DoHBrw-2020 benchmark dataset [10] was used for training and testing. According to the results, the RF classifier achieved the best possible results (100%) in terms of precision, recall and F1-score while the composed DT, LR and k-NN ensemble classifier performed only slightly worse.

Testing different kinds of machine learning algorithms (beyond artificial neural network (ANN)) on DoH traffic recognition, including its classification into benign vs. malignant, was the focus of the work [11]. Among other things, the authors also focused on feature engineering, in the process of which they identified and removed features that were insignificant for traffic classification. They thus achieved a noticeable reduction in the time required to train the model and its prediction. The average prediction time of all the tested algorithms was below 1 s, allowing their eventual use in practice. As in the work of Wang *et al.* [7], the CIRA-CIC-DoHBrw-2020 dataset was used for training and classification.

Trying to get around complicated feature engineering led the Ding *et al.* [12] to propose an end-to-end anomaly detection model based on a variational autoencoder which incorporates the attention mechanism. They used a bidirectional GRU-based network to automatically learn the feature representations and detect anomalies via reconstruction error. Huang *et al.* [13] deals with the evaluation of several ML models detecting malicious DoH traffic. The authors optimized hyperparameters of several models and compared them with respect to precision, accuracy, recall and F1-Score. They concluded that RF and DT models performed best in comparison with KNN, 1D convolutional neural network (CNN), 2D CNN and long short-term memory (LSTM) models. Training and testing were conducted on CIRA-CIC-DoHBrw-2020 dataset. The CNN-oriented approach was used also in [14].

The testing of different machine learning algorithms for recognizing DoH traffic from classic web traffic was also addressed in [15], [16]. Unlike the other works reported here, the training and testing dataset was generated by the authors themselves - by capturing traffic to the top 20,000 most visited domains from Alexa's list of top 1 million websites. Beyond the DoH traffic detection itself, they also focused on finding techniques that would, in turn, significantly reduce the detection accuracy of the trained ML model used, e.g., on the ISP side.

Casanova and Lin [17] used LSTM and bidirectional long short-term memory (BiLSTM) models for DoH traffic classification. Their aim was to create a generalized, portable model, independent of the target deployment environment. The BiLSTM model achieved better results in terms of accuracy and training and classification time.

Jha *et al.* [18] focuses on the detection of DoH tunneling. The authors built a test environment and created their dataset along with CIRA-CIC-DoHBrw-2020. They observed that many tunneling instances had large packet sizes and long request duration so they used outlier detection models, namely k-NN, SVM, deep factorization machines (DeepFM) and RF. All the mentioned models, except k-NN, achieved excellent results over 99% in terms of FI-score.

In addition to achieving high DoH traffic detection accuracy, Zebin *et al.* [19] and Banadaki [20] focused on the transparency of ML model decision making, which has received increasing attention in recent years in the context of explainable machine learning. The authors constructed a Balanced Stacked Random Forest classifier and used Shapley additive explanations (SHAP) values to illustrate the impact of individual features on the model's decision making. The detection results were also convincing, over 99.9% in terms of FI-score.

Steadman and Scott-Hayward [21] deals with the design of system architecture for detection and mitigation of malicious DNS and DoH communication. The DoHxP architecture is based on SDN. Nguyen and Park [22] proposed a detection system for DoH tunneling attacks based on Transformer to detect a malicious DoH tunneling. The main advantage over conventional supervised machine learning approaches is that training requires significantly less labeled data, the authors report around 20% compared to another research, while achieving F1 over 99%.

Zhan *et al.* [23], Li *et al.* [24] also focused on the detection of DoH tunneling. They divided the detection into 2 phases. In the first, preliminary phase, they tested TLS data and compared it with fingerprints of DoH clients, according to the authors' findings, fingerprints of DoH clients are often unique. In the second phase, flow-based features were fed to the trained ML classifier. The authors tested a total of 3 classifiers: boosted DT, RF and LR, investigating the effect of location and the recursive solver used on detection accuracy. Shatoori *et al.* [25] came up with the idea of packet clumps (sequence of consecutive packets in a network flow) to find patterns in a limited window of traffic, which can reduce detection latency. They examined and analyzed dependency of accuracy and response time on the number of packet clumps in a sliding window.

N-shot learning was used by Zou *et al.* [26]. Their model, called Depl, outputs websites a user visited. Depl uses BiLSTM to extract features of the input data, which consists of sequences of packet sizes. They achieved remarkable results using a very small number of training samples, only 5 samples were enough for an accuracy of around 86% in a closed environment. Al-Fawa'reh *et al.* [27] combined the bi-directional recurrent neural networks (RNNs) and the statistical methods and achieved very good accuracy for a specific dataset.

3. OUR APPROACH AND PROPOSED SOLUTION

Our team has developed a special platform allowing us to capture and visualize the data retrieved from network traffic probes. A graphical representation of the traffic per packet is depicted in Figure 1, the green spots mean the direction from a client to a server and the red spots are the reverse direction. The X-axis represents the time and Y-axis the packet size in bytes.

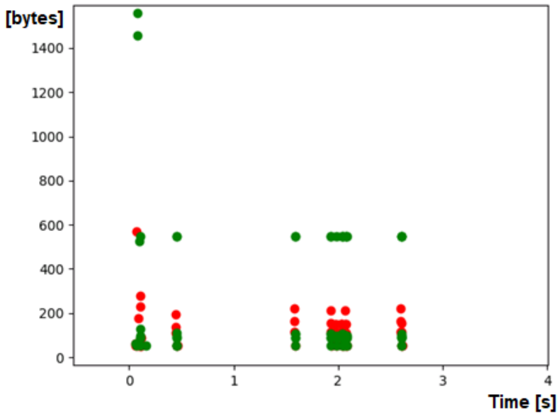


Figure 1. Visualization of DoH network traffic flow. Each spot means a single packet. The red color means the direction from source to destination and the green backwards

In our work, for a DoH connection, we have introduced the term DoH handshake (DoHH). DOHH means one complete DNS REQUEST/RESPONSE over HTTPS. One DoH connection contains a TLS initial handshake and then a consequence of standalone DOHHs, in detail depicted in Figures 2 and 3. This approach showed as suitable and used for the evaluation of practical measurements. The motivation for the introduction of the DOHH mechanism was the effort to divide the communication between the client and the server into logically related parts, which would subsequently allow easier processing and separation of DoH traffic.

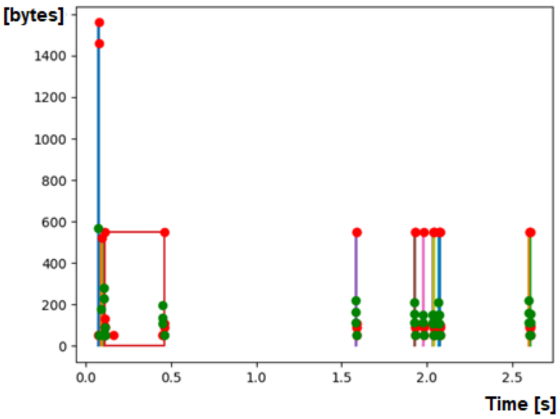


Figure 2. The visualization of a concrete DoH network traffic flow containing detected DoHHs

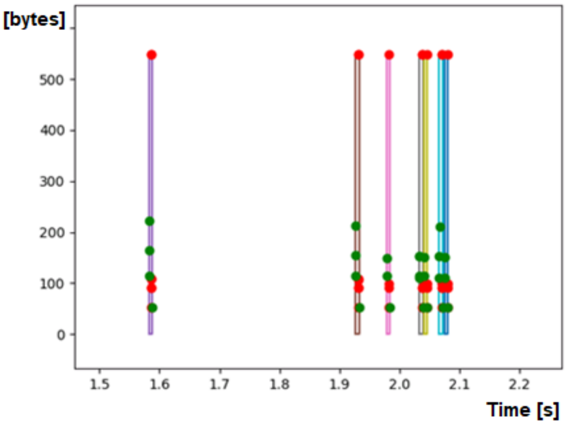


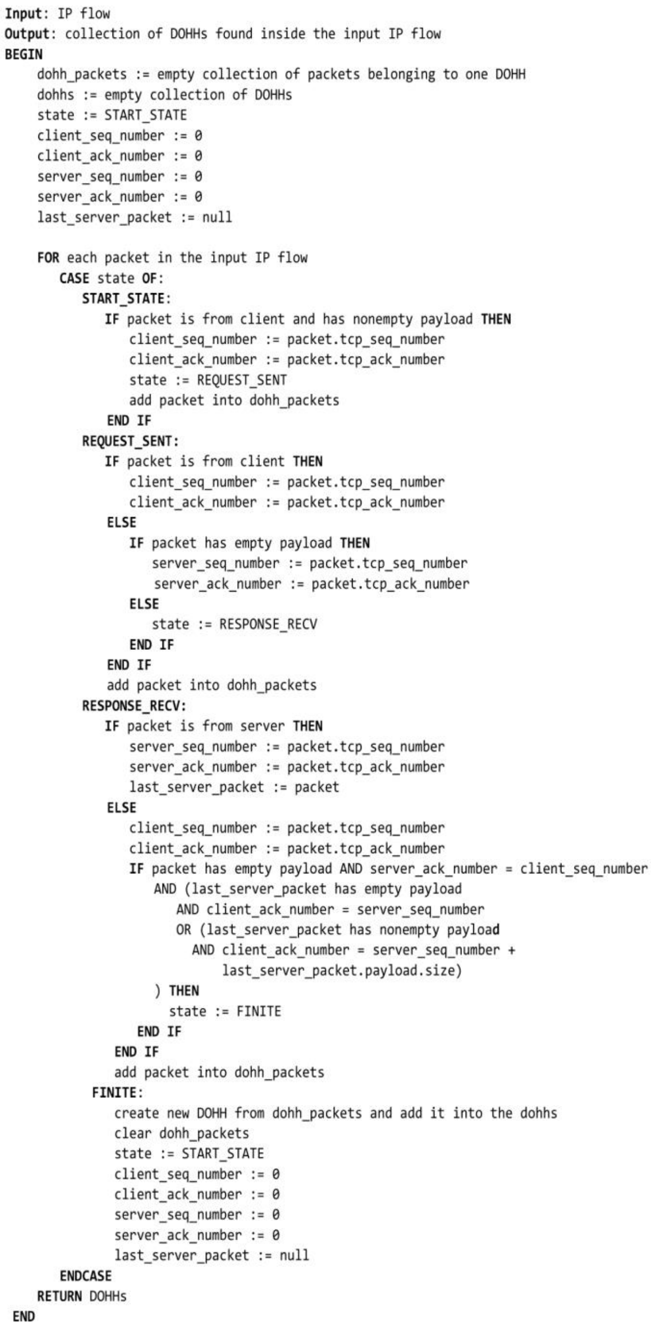
Figure 3. Zoom on the DoH traffic. The DoHH within the specified time interval. Each rectangle represents a single DoHH

The DoHH strives to define and encapsulate the processing of a particular client request (e.g., a DoH request) by the server. A DoHH is a subset of records about individual packets from a particular IP flow. Within the detection process, one particular IP flow is divided into a set of individual DoHH.

Each DoHH begins with sending a client request to the server and ends with the following conditions:

a. The client sent an empty acknowledgement to the server.
b. The ACK (acknowledgement number) of the last packet received from the server is equal to the SEQ (sequence number) of the acknowledgement sent from the client.
c. The ACK of the acknowledgement sent from the client is equal to the SEQ of the last packet received from the server (plus the length of the data if the packet was non-empty).

Based on these assumptions, we created a model of the DoHH detection algorithm. It is based on the assumption that DoH uses the transmission control protocol (TCP as the transport protocol). Each TCP packet contains its specific SEQ and ACK number. Thanks to the comparison of the SEQ and ACK values and measuring the transported payload size, the model is able to get rid of the duplicate packets and find the start and end of each DoHH. The detailed algorithm pseudocode could be seen in Figure 4. The model is realized as a finite state machine that transitions between the states - {*START_STATE*, *REQUEST_SENT*, *RESPONSE_RECV*, *FINITE*} according to the information contained within headers of received TCP packets. The input of this algorithm is an IP flow consisting of the TCP packets and the output of the algorithm is a list of DOHHs.



```
Input: IP flow
Output: collection of DOHHs found inside the input IP flow
BEGIN
    dohh_packets := empty collection of packets belonging to one DOHH
    dohhs := empty collection of DOHHs
    state := START_STATE
    client_seq_number := 0
    client_ack_number := 0
    server_seq_number := 0
    server_ack_number := 0
    last_server_packet := null

    FOR each packet in the input IP flow
       CASE state OF:
          START_STATE:
             IF packet is from client and has nonempty payload THEN
                client_seq_number := packet.tcp_seq_number
                client_ack_number := packet.tcp_ack_number
                state := REQUEST_SENT
                add packet into dohh_packets
             END IF
          REQUEST_SENT:
             IF packet is from client THEN
                client_seq_number := packet.tcp_seq_number
                client_ack_number := packet.tcp_ack_number
             ELSE
                IF packet has empty payload THEN
                   server_seq_number := packet.tcp_seq_number
                   server_ack_number := packet.tcp_ack_number
                ELSE
                   state := RESPONSE_RECV
                END IF
             END IF
             add packet into dohh_packets
          RESPONSE_RECV:
             IF packet is from server THEN
                server_seq_number := packet.tcp_seq_number
                server_ack_number := packet.tcp_ack_number
                last_server_packet := packet
             ELSE
                client_seq_number := packet.tcp_seq_number
                client_ack_number := packet.tcp_ack_number
                IF packet has empty payload AND server_ack_number = client_seq_number
                   AND (last_server_packet has empty payload
                      AND client_ack_number = server_seq_number
                      OR (last_server_packet has nonempty payload
                        AND client_ack_number = server_seq_number +
                            last_server_packet.payload.size)
                   ) THEN
                     state := FINITE
                 END IF
              END IF
              add packet into dohh_packets
           FINITE:
              create new DOHH from dohh_packets and add it into the dohhs
              clear dohh_packets
              state := START_STATE
              client_seq_number := 0
              client_ack_number := 0
              server_seq_number := 0
              server_ack_number := 0
              last_server_packet := null
       ENDCASE
    RETURN DOHHs
 END
```

Figure 4. Transformation of TCP packets into DoHHs. The above-depicted algorithm works as a state machine

3.1. ML-model creation, testing, and optimization

The approach to model design and preprocessing of input data has gradually evolved. The first version of the model was based on the basic paradigms associated with neural networks, and fully interconnected (dense) layers were used as crucial layers. However, the process of designing a specific model is not described by a general methodology, so this phase was supported by the creation of a specialized software tool forming an extension of the Keras API. All presented models are optimized and trained on the GPU NVIDIA GeForce 2080 TI.

The structure of the optimization superstructure can be divided into several layers. The fundamental part represents hyper-parameters and other factors influencing the model's learning (after this only parameters). Appropriate classes are defined for individual parameter types, allowing to specify the allowed range and the methods of selecting a specific parameter value. Each parameter instance of any type is then able to generate this value within the set limits. Specific parameters are the ones related to the description of the model's internal structure, especially its hidden layers. It is possible to define their number, eventually type, and other parameters. Above this layer of the optimizer, a list of parameters used to set up the model is then created. Any adjustable parameter of the model can be selected for optimization and experiments with its value. The final list of parameters forms a mixed-type vector with coordinates dedicated to optimizing their value. Each coordinate value of this vector is set (generated or otherwise selected) in each optimization step. The set model is then trained and tested. The value of loss of accuracy was chosen as a measure of quality when using a test set; we try to minimize this value. Genetic algorithms were chosen to control the optimization process.

3.2. Developed deep neural network models

Using the optimization mentioned above, it was subsequently found that on the given input data (described below), the given classification task can be satisfactorily solved with a small number of hidden layers and neurons in them. For instance, the model described in Figure 5 achieved good results. In addition to fully interconnected dense layers, layers supporting the robustness of network learning were used to improve the model's function (a layer implementing Gaussian noise and a dropout layer randomly zeroing some elements of the previous layer's output

Experiments with the model structure have also shown that networks containing convolution layers generally achieve better results. Several models were tested with this global architecture again, including models containing multiple parallel convolution layers with different convolution kernels. However, the model shown in Figure 6 achieved the best results in testing.

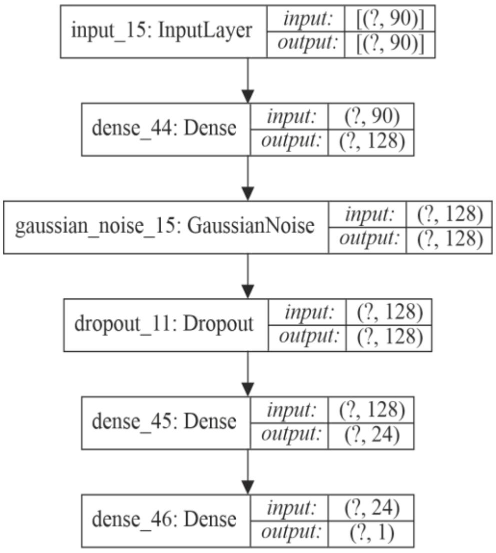


Figure 5. The model is based on fully interconnected (dense) layers

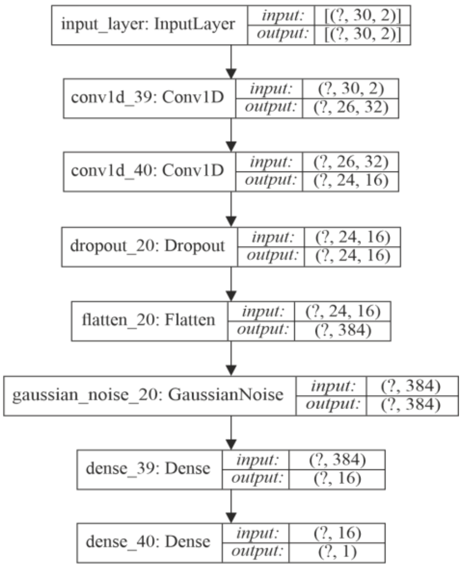


Figure 6. The architecture of the model uses convolutional layers. This model achieved the highest accuracy

4. RESULTS AND DISCUSSION

4.1. Dataset creation

The creation of the data set was based on the following idea. We selected approx. 10,000 real active domains with enabled HTTPS under the national domain CZ. Commonly, DoH traffic can be considered normal HTTPS traffic. The generated dataset contains the DoH and non-DoH sets of packet flows. The DoH packet flows were distinguished according to the specific IP addresses.

The DoH and non-DoH packet flows were generated automatically by a periodical execution of the Firefox-extended support release (ESR) web browser. The network traffic was reported by network probes via the internet protocol flow information export (IPFIX) network protocol. The data set is freely downloadable via [28].

4.2. Input dataset preprocessing

The original idea was to train the neural network model on three basic parameters of each packet. These parameters were the packet size in bytes, packet direction, and time distance from the previous packet. These data were used from the first 30 packets from each data flow (data flow here means continuous communication between two IPs using the same ports). The 90-element vector was then used for training and testing models described above and based on dense and convolutional layers.

Further was also performed with the DOHH-based structure of input data, not data flows. The method of determination of these DOHHs is given above. In this case, data from the first 20 packets in each DoHH were used. The characteristics of these packets were the same as mentioned above.

The data itself came from two different sources. The first one was the catching of DoH queries within the CZ.NIC network on the model infrastructure. In this case, several groups of data sets were generated. The first group was created by modelling a separate communication for each DoH query when there was a separate communication flow for each query (CZ.NIC S sets). The second group (CZ.NIC L sets) was obtained by modelling communication where one communication flow contained more DoH queries. The second data source was data from the model infrastructure's communication with DoH servers located within the domain cloudflare.com (CLOUDFLARE set). For DoHH-based testing, the CZ.NIC N and CLOUDFLARE N sets were generated from the data flow described above. For sets from the CZ.NIC S and N sets, the obtained data were divided into three essential parts: training, validation, and test set. The first two data sets were used in the training of models. The last part, together with data from other sets, was used to evaluate the model's generalization abilities in the tests.

The models were always trained on data from the CZ.NIC network for separate queries against DoH servers (from the CZ.NIC S and N set). The experiments aimed to verify in practice whether models were trained to communicate with a specific server in the CZ.NIC network will be able to detect DoH communication with other servers (e.g., cloudflare.com). That was also a measure of the model quality.

4.3. Experimental results and evaluation

Within the project, a large number of experiments were performed with different model structures. Only selected top-quality outputs are listed below. The best results were obtained with the model shown in Figure 6. The models' evaluation is given in Table 1 and 2 for data from the CZ.NIC network and data from communication with the cloudflare.com domain.

In Table 1, the columns describe the input data used for testing, the number of test examples and the total input vector dimension. In Table 2, the first two columns describe the model type and the number of packets from which the input data was used. The type of model follows, type D1 model was based on dense layers (depicted in Figure 5), while models C were on convolution layers (depicted in Figure 6). The following are four columns for testing evaluation (true positive, true negative, false positive, false negative). In Figure 7, it is possible to see the achieved quality of all models. The quality is computed as

$$q = (true\ positive\ (TP) + true\ negative(TN))/(TP + TN + false\ positive(FP) + false\ negative\ (FN)),$$

from the data contained in Table 2.

Table 1. Input datasets and selected neural network models (Dense vs CNN)

Data	Samples	Input dimension	Model type	Packets used
CZ.NIC S	999	90	Dense	30
CZ.NIC L	94	90	Dense	30
CZ.NIC S	999	60	CNN	30
CZ.NIC L	94	60	CNN	30
CLOUDFLARE	5938	60	CNN	30
CZ.NIC S	999	60	CNN	30
CZ.NIC L	94	60	CNN	30
CLOUDFLARE	5938	60	CNN	30
CZ.NIC S	998	60	CNN	30
CLOUDFLARE	5938	60	CNN	30
CZ.NIC S	998	60	CNN	30
CLOUDFLARE	5938	60	CNN	30
CZ.NIC N	3898	40	CNN	20
CLOUDFLARE N	4095	40	CNN	20

The results show that the D1 model also achieved good results. However, already on the CZ.NIC network data, these results were worse than in the models involving convolutional layers. Therefore, these models were not further tested. Several experiments were performed with type C models to find the model's optimal setting in terms of its parameters and hyperparameters. Somewhat paradoxically, it turned out that the best results were achieved already by the second tested model, C2, which is also shown in Figure 6. Its quality lies in correctly identifying DoH data against the cloudflare.com domain and thus in its robustness.

Table 2. Achieved results for specific datasets and models

Data	Model	TP	TN	FP	FN
CZ.NIC S	D1	488	490	7	14
CZ.NIC L	D1	41	48	4	1
CZ.NIC S	C2	491	493	2	13
CZ.NIC L	C2	42	49	0	3
CLOUDFLARE	C2	36	5806	78	18
CZ.NIC S	C3	499	487	8	5
CZ.NIC L	C3	44	47	2	1
CLOUDFLARE	C3	47	5694	190	7
CZ.NIC S	C4	482	500	9	7
CLOUDFLARE	C4	10	5734	150	44
CZ.NIC S	C5	479	500	9	10
CLOUDFLARE	C5	5	5629	255	49
CZ.NIC N	N1	1919	1923	21	35
CLOUDFLARE N	N1	197	2014	16	1868

The tests were also performed on DoHH-based inputs (the last two rows of the table). The model's quality was very high on data from the CZ.NIC network. However, its generalization ability tested on data from cloudflare.com was only small, even if three parallel CNN layers with different kernel sizes were used. The overall achieved quality of all models is depicted in Figure 7.

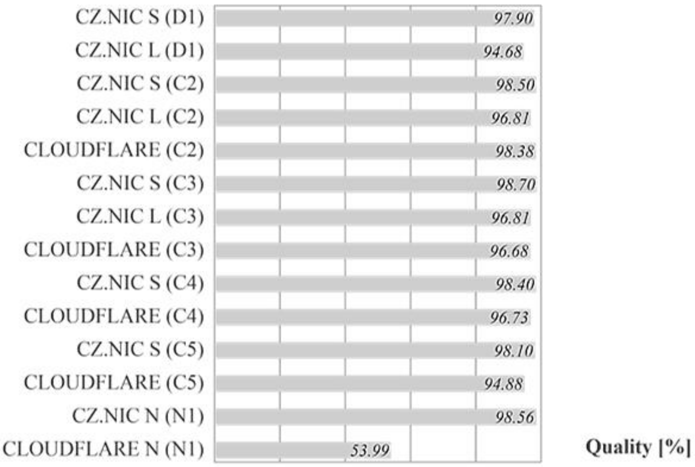


Figure 7. Achieved quality for different neural network models and their comparison

5. CONCLUSION

In our work, we created a new dataset for the DoH network traffic detection. We analyzed the DoH packet flows and introduced a new algorithm for DoHH detection. Based on our data set, we made many practical experiments a proposed dense and CNN-based neural network models. The models were successfully used on two data sets and achieved accuracy higher than 95%.

The models trained on the data coming from provider CZ.NIC were able to detect the DoH connections from another provider, Cloudflare. These results confirmed the ability to generalization of our proposed models. All presented models are part of the free available dataset and can be downloaded via https://gitlab.prf.jcu.cz/root/dohgpu/.

In our future work, we would like to target the graphical visualization of the DoH traffic and the usage of computer vision methods suitable for its identification. This approach has a great advantage in the sense that is near to human thinking and easily checkable. The next gap which is worthwhile for attention is the elimination of the jitter of the TCP packets causing the ambiguity of the DoH patterns.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the organization CESNET for providing the financing and to the University of South Bohemia for providing the technical background necessary for the research evaluation. We also would like to thank Mrs Marie Feslová for her support with the graphic works.

REFERENCES

[1] P. Hoffman and P. McManus, “DNS queries over HTTPS (DoH),” *Internet Engineering Task Force (IETF)*. 2018.

[2] M. Trevisan, F. Soro, M. Mellia, I. Drago, and R. Morla, “Does domain name encryption increase users’ privacy?,” *ACM SIGCOMM Computer Communication Review*, vol. 50, no. 3, pp. 16–22, Jul. 2020, doi: 10.1145/3411740.3411743.

[3] D. Hjelm, “New needle and haystack: Detecting DNS over HTTPS usage,” SANS Technology Institute, North Bethesda, MD, USA, 2019.

[4] E. Rescorla, K. Oku, N. Sullivan, and C. A. Wood, “Encrypted server name indication for TLS 1.3,” *IETF*. 2020.

[5] D. Vekshin, K. Hynek, and T. Cejka, “DoH insight: Detecting DNS over HTTPS by machine learning,” in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, Aug. 2020, pp. 1–8, doi: 10.1145/3407023.3409192.

[6] J. Bushart and C. Rossow, “Padding ain’t enough: Assessing the privacy guarantees of encrypted DNS,” *Prepr. arXiv.1907.01317*, Jul. 2019.

[7] Y. Wang, A. Zhou, S. Liao, R. Zheng, R. Hu, and L. Zhang, “A comprehensive survey on DNS tunnel detection,” *Computer Networks*, vol. 197, Oct. 2021, doi: 10.1016/j.comnet.2021.108322.

[8] K. Hynek and T. Cejka, “Privacy illusion: Beware of unpadded DoH,” in *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Nov. 2020, pp. 621–628, doi: 10.1109/IEMCON51383.2020.9284864.

[9] S. K. Singh and P. K. Roy, “Malicious traffic detection of DNS over HTTPS using ensemble machine learning,” *International Journal of Computing and Digital Systems*, vol. 11, no. 1, pp. 1061–1069, Mar. 2022, doi: 10.12785/ijcds/110185.

[10] UNB, “CIRA-CIC-DoHBrw-2020,” *Canadian Institute for Cybersecurity*, 2020. https://www.unb.ca/cic/datasets/dohbrw-2020.html (accessed Jan. 05, 2022).

[11] M. Behnke *et al.*, “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, doi: 10.1109/ACCESS.2021.3113294.

[12] S. Ding, D. Zhang, J. Ge, X. Yuan, and X. Du, “Encrypt DNS traffic: Automated feature learning method for detecting DNS tunnels,” in *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, Sep. 2021, pp. 352–359, doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00056.

[13] Q. Huang, D. Chang, and Z. Li, “A comprehensive study of DNS-over-HTTPS downgrade attack,” in *FOCI @ USENIX Security Symp.*, 2020, pp. 1–8.

[14] M. Konopa, J. Fesl, J. Jelínek, M. Feslová, J. Cehák, and F. Drdák, “Using machine learning for DNS over HTTPS detection,” Jun. 2020, doi: 10.34190/EWS.20.001.

[15] L. Csikor, H. Singh, M. S. Kang, and D. M. Divakaran, “Privacy of DNS-over-HTTPS: Requiem for a dream?,” in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, Sep. 2021, pp. 252–271, doi: 10.1109/EuroSP51992.2021.00026.

[16] R. Zheng, J. Liu, W. Niu, L. Liu, K. Li, and S. Liao, “Preprocessing method for encrypted traffic based on semisupervised clustering,” *Security and Communication Networks*, vol. 2020, pp. 1–13, Jul. 2020, doi: 10.1155/2020/8824659.

[17] L. F. G. Casanova and P.-C. Lin, “Generalized classification of DNS over HTTPS traffic with deep learning,” in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2021, pp. 1903–1907.

[18] H. Jha, I. Patel, G. Li, A. K. Cherukuri, and S. Thaseen, “Detection of tunneling in DNS over HTTPS,” in *2021 7th International Conference on Signal Processing and Communication (ICSC)*, Nov. 2021, pp. 42–47, doi: 10.1109/ICSC53193.2021.9673380.

[19] T. Zebin, S. Rezvy, and Y. Luo, “An explainable AI-based intrusion detection system for DNS over HTTPS (DoH) attacks,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2339–2349, 2022, doi: 10.1109/TIFS.2022.3183390.

[20] Y. M. Banadaki, “Detecting malicious DNS over HTTPS traffic in domain name system using machine learning classifiers,” *Journal of Computer Sciences and Applications*, vol. 8, no. 2, pp. 46–55, Aug. 2020, doi: 10.12691/jcsa-8-2-2.

[21] J. Steadman and S. Scott-Hayward, “Detecting data exfiltration over encrypted DNS,” in *2022 IEEE 8th International Conference on Network Softwarization (NetSoft)*, Jun. 2022, pp. 429–437, doi: 10.1109/NetSoft54395.2022.9844067.

[22] T. A. Nguyen and M. Park, “DoH tunneling detection system for enterprise network using deep learning technique,” *Applied Sciences*, vol. 12, no. 5, Feb. 2022, doi: 10.3390/app12052416.

[23] M. Zhan, Y. Li, G. Yu, B. Li, and W. Wang, “Detecting DNS over HTTPS based data exfiltration,” *Computer Networks*, vol. 209, May 2022, doi: 10.1016/j.comnet.2022.108919.

[24] Y. Li, A. Dandoush, and J. Liu, “Evaluation and optimization of learning-based DNS over HTTPS traffic classification,” in *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, Oct. 2021, pp. 1–6, doi: 10.1109/ISNCC52172.2021.9615659.

[25] M. M. Shatoori, L. Davidson, G. Kaur, and A. H. Lashkari, “Detection of DoH tunnels using time-series classification of encrypted traffic,” in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, Aug. 2020, pp. 63–70, doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00026.

[26] F. Zou, D. Meng, W. Gao, and L. Li, “DePL: Detecting privacy leakage in DNS-over-HTTPS traffic,” in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Oct. 2021, pp. 577–586, doi: 10.1109/TrustCom53373.2021.00088.

[27] M. Al-Fawa’reh, Z. Ashi, and M. T. Jafar, “Detecting malicious DNS queries over encrypted tunnels using statistical analysis and bi-directional recurrent neural networks,” *Karbala International Journal of Modern Science*, vol. 7, no. 4, Dec. 2021, doi: 10.33640/2405-609X.3155.

[28] J. Fesl, “DoH Traffic dataset,” GitLab. Accessed: Feb. 28, 2023. [Online]. Available: https://gitlab.prf.jcu.cz/root/dohgpu/

BIOGRAPHIES OF AUTHORS

Jan Fesl received an M.S. degree in electrical engineering from Czech Technical University in Prague in 2007 and a Ph.D. degree in Computers and Informatics in 2018. He is a research assistant at the University of South Bohemia, Faculty of Science. His research interests include computer networks, cyber-security, cloud computing, HPC infrastructures, and Distributed and Parallel programming. He can be contacted at email: jfesl@prf.jcu.cz.

Michal Konopa received an M.S. degree in software systems from Charles University in Prague, Czech Republic, in 2010. Currently, he is an assistant at the Department of Computer Science, University of South Bohemia. His main research interests include algorithms and optimization, network protocol analysis, SAT solving, and machine learning. He can be contacted at email: konopm05@prf.jcu.cz.

Jiří Jelínek received the M.S. (Ing.) degree in electrical engineering from Czech Technical University in Prague (CZ), Faculty of Electrical Engineering in 1989, and the Ph.D. (CSc.) degree in electric power engineering from the same University and Faculty in 1992. He is an Associate Professor at the Department of Informatics, Faculty of Science, University of South Bohemia, Czech Republic. His research interests include artificial neural networks and their applications, artificial intelligence and machine learning techniques and applications, knowledge representation, modeling and simulation of multiagent systems, multimedia processing with IT, web technologies, Linux OS, and using modern information technology in the learning process. He can be contacted at email: jjeliinek@prf.jcu.cz.