Open access • Proceedings Article • DOI:10.1109/NOMS.2018.8406218

Fingerprinting encrypted network traffic types using machine learning — Source link

Sam Leroux, Steven Bohez, Pieter-Jan Maenhaut, Nathan Meheus ...+2 more authors

Institutions: Ghent University

Published on: 23 Apr 2018 - Network Operations and Management Symposium

Topics: Encryption, Network administrator, IPsec, Quality of service and Strong cryptography

Fingerprinting encrypted network traffic types using machine learning

Sam Leroux, Steven Bohez, Pieter-Jan Maenhaut, Nathan Meheus, Pieter Simoens, Bart Dhoedt
Department of Information Technology, IDLab, Ghent University - imec
sam.leroux@ugent.be

***Abstract*—Internet applications rely on strong encryption techniques to protect the content of all communications between client and server. These encryption algorithms ensure that third parties are unable to obtain the plain text data but also make it hard for the network administrator to enforce restrictions on the types of traffic that are allowed. In this paper we show that we can train accurate machine learning models which can predict the type of traffic going through an IPsec or TOR tunnel based on features extracted from the encrypted streams. We use small, fast to execute machine learning models that work on small windows of data. This makes it possible to use our approach in real-time, for example as part of a Quality of Service (QoS) system.**

I. Introduction

Modern internet technologies such as video streaming, Voice over IP (VoIP) and Peer-to-Peer (P2P) file sharing systems all require large amounts of bandwidth. To ensure a smooth user experience a Quality of Service (QoS) system can allocate bandwidth for every stream based on the type of traffic. It is however not always easy to discover the type of traffic contained in each stream. The most straight-forward approach would be to look at the source and destination port numbers. This is very fast but not always reliable. Most applications allow the user to change the port numbers and some applications (like Bittorrent clients) can even use random port numbers. Various works have shown that a simple port based fingerprinting technique performs poorly on real world data [1][2][3].

A more robust approach is to look at the actual content of the network packets. Different applications each follow a different protocol that is easy to recognize. Deep packet inspection techniques can examine the payload to extract this information but only if the required data is transmitted as plain text. Virtual Private Networks (VPNs) allow the user to encapsulate network traffic through an encrypted tunnel which hides the content and the actual destination of the network stream. Because of this additional security layer it is hard to enforce any restrictions on the types of traffic that are allowed in the network.

Even more privacy is offered by technologies such as The Onion Router (TOR) [4]. TOR uses a network of relay nodes to provide anonymity to its users. It does so by routing all traffic over three relay nodes before it is routed to its actual destination. Each relay node only knows the previous and the next node, resulting in anonymity for the user. To provide data confidentiality, TOR establishes a TLS connection between the client and every relay node. This produces several layers of encryption, comparable with the layers of an onion, hence the name.

In this paper we describe a machine learning pipeline that is able to discover the type of traffic going through a VPN (IPSec) or TOR tunnel using only features extracted from the encrypted network stream. We distinguish between four types of traffic: Web browsing (HTTP), VoIP (Skype), Video streaming (YouTube) and P2P (Bittorrent). We define features based on the timing and size of the encrypted packets and train three different machine learning models for classification (naive Bayes, logistic regression and random forest) on this data. We show that these simple models are sufficient to predict the traffic type with a high accuracy.

This paper is organized as follows: we start with an overview of related work in Section II. In Section III we describe our machine learning pipeline and in Section IV we present our results both for IPSec and TOR encrypted network traffic. We conclude in Section V and discuss some interesting possibilities for future work.

II. Related work

Network traffic analysis has gathered a lot of interest both from industry and academia. We focus only on passive fingerprinting approaches which do not interfere with the packet streams. Active network fingerprinting techniques on the other hand rely on injecting traffic or on compromised or malicious nodes in the network [5].

Technologies such as TOR and IPSec can protect the content of the communications from eavesdroppers and if used correctly they can also provide anonymity. Even though the actual payload is unavailable for third parties, it is still possible to extract information based on the timing and size of individual packets.

Common features extracted from timing and size information are packet size combined with direction (upstream or downstream) and inter-arrival time. These features are used in [6] to distinguish between five types of network traffic. Features can also be observed at an aggregated level of multiple packets. A burst is such a possible aggregation, and is defined as a sequence of packets sent in one direction that lie between two packets sent in the opposite direction[7].

Both the timing and the sizes of bursts can be useful. Another interesting aggregation proposed in literature is the “Surge Period”. A Surge Period marks the parts of a traffic trace where the channel is continuously busy transmitting packets upstream or downstream. These features are used in [7] combined with the first *n* components of Haar Wavelet Transformations to predict the website a user is visiting through a VPN tunnel. It is shown that these features can be used to distinguish between eleven popular websites with dynamic content.

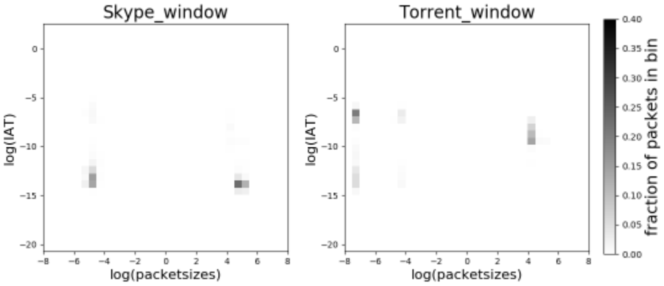
A similar experiment was performed by Liberatore and Levine who used a naive Bayes classifier to predict the visited web page out of a set of 2000 different pages, based on the packet size combined with the direction (downstream or upstream) [8]. They ignored all timing information but found that they were still able to predict which website from a list of 2000 was visited with an accuracy of up to 90%.

Most works that apply machine learning to this problem use relatively simple machine learning techniques based on manually defined features. Deep learning on the other hand allows us to learn the features from data instead of having to engineer them. The Deep packet paper [9] introduces an end-to-end approach to train neural networks both for traffic categorization and application identification.

Some of these techniques can pose severe privacy risks for the end users. There has been some work on practical countermeasures that make it harder to extract privacy sensitive information from the network streams. TOR for example packs all data into 512-byte cells and even includes an experimental feature where random HTTP pipelining is used to make it harder to discover which website was visited [10]. Other techniques pad packets to 2^k bytes or to the MTU [11]. It is even possible to transform the traffic to make it look like traffic from a different application by mimicking the distribution of packet sizes [12]. Unfortunately most of these countermeasures fail at hiding the information completely [13].

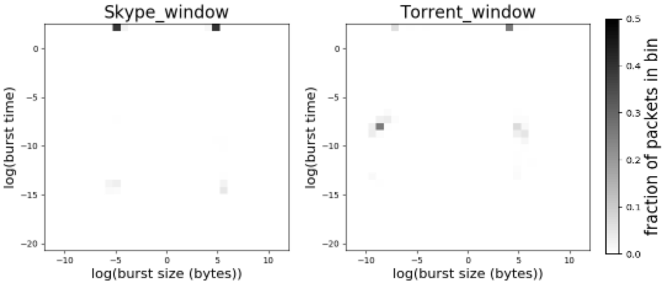
III. Feature extraction

We propose to represent the extracted features as two-dimensional histograms. This allows us to easily visualize the relationship between different features. We defined two feature spaces that each visualize the relationship between two features: packet size-interarrival time and burst time-burst size. We extracted windows containing 1024 packets from each trace and generated the histograms for each window independently. This windowed approach allows us to use the classifier after every 1024 packets that are captured, for example as part of an online QoS system. We use a logarithmic data representation to focus more on the relative magnitude rather than on the actual value of a feature.



(a) Skype (b) Bittorrent

Figure 1: The packet size - interarrival time feature space for a Skype and a Bittorrent window.



(a) Skype (b) Bittorrent

Figure 2: The burst size - Surge period feature space for a Skype and a Bittorrent window.

A. *Packet size - interarrival time*

Packet size and interarrival time (IAT) are the two most straight-forward features that we can extract from encrypted network traffic streams. The size of a packet is directly linked to the contained payload, which follows an application specific profile and can thus be used for classifying the application. We indicate the direction of the packet (downstream or upstream) by the sign, a negative value indicates downstream traffic, a positive value indicates upstream traffic.

The main drawback of using packet sizes as a feature is the ease by which privacy providing protocols can influence this feature by padding the encrypted payload with a pseudorandom number of bytes or by padding all packets to the same size. The timing of packets is less adaptable, since it would have a direct impact on the performance of the web service [14]. We combine both packet size and IAT to create a two-dimensional histogram. An example of this visualization is shown in Figure 1 for a window of Skype and Bittorrent traffic. Each bin in these histograms is colored according to the amount of packets in it. For example, Skype traffic consist primarily of small packets with short interarrival times while Bittorrent traffic typically has larger interarrival times and larger packet sizes.

B. Burst size - Surge period

A burst is defined as a sequence of (non-acknowledgement) packets sent in one direction that lie between two packets sent in the opposite direction. The bandwidth of a burst is the total size of all packets contained in the burst, in bytes, and the burst count is the number of packets within the burst [7][13]. This two-folded, aggregated feature describes the properties of the network traffic at a higher level, because it does not only take packet features into account, but also the correlation between packets. A burst can typically be used to identify a web page fetch.

Together with the burst size we also use the surge period. The surge period marks the parts of a traffic trace where the channel is continuously busy transmitting packets upstream or downstream. Any packet within the surge period should be separated from its predecessor and subsequent packets by a time period no larger than a predefined time window size.

An example of this feature space is visualized in Figure 2. The same reasoning as with the packet size-interarrival time histograms applies here. Figure 2 shows that Skype windows contain mostly longer bursts, and Bittorrent bursts tend to be more of medium length.

IV. Results

In this section we present the results obtained by training three machine learning models for classification (naive Bayes, logistic regression and random forest) on the feature representations described in the previous section. We generated our own dataset of traffic traces. A VoIP trace starts just before starting or receiving a voice call and goes on for a brief moment of the actual call. The Bittorrent traces start when we added a torrent file to the torrent client, and some time of the download. The HTTP browsing covers a Google search and following one of the suggested links, and the YouTube traffic consists of visiting the YouTube homepage, clicking a video link and watching the video play for some time. For each experiment we captured 40 traces (ten for each application) and extracted windows containing 1024 packet each. This results in around 500 windows. We used 20% of these samples as a test set and used the remaining 80% for training and validation.

A. Unencrypted data

As a baseline we trained our models on unencrypted network streams. The results are summarized in Table I. This table shows that all three machine learning models achieve a high accuracy which indicates that these features capture enough information to distinguish the different traffic types, at least on unencrypted data. The logistic regression classifier consistently performs the best and we also find that a combination of both feature spaces results in the highest accuracy for all machine learning models.

Accuracy	Naive Bayes	Logistic Regression	Random Forest
Size - IAT	94.55%	99.09%	97.27%
Burst features	96.36%	99.09%	97.27%
Combined	97.27%	100.00%	99.09%

Table I: Classification accuracy for the different feature spaces for three different machine learning models using data from unencrypted streams.

Accuracy	Naive Bayes	Logistic Regression	Random Forest
Size - IAT	93.55%	95.70%	96.77%
Burst features	87.10%	94.62%	93.55%
Combined	94.62%	95.70%	96.77%

Table II: Classification accuracy for the different feature spaces for three different machine learning models using data from IPsec encrypted streams.

B. IPsec VPN encrypted data

IPsec is a network protocol operating in the Internet Layer of the Internet Protocol Suite that can be used to authenticate and encrypt network traffic. We configured IPsec to use tunnel mode with Encapsulated Security Payload (ESP) to provide both encryption and authentication. In this configuration the original packet is encrypted and encapsulated in a new packet.

Since both packet size and timing determine the feature spaces, it is useful to understand the effect of IPsec on these characteristics. The timing is only influenced by the time needed for the encryption/decryption and authentication check of the packets. In our configuration, IPsec uses the Advanced Encryption Standard (AES), with a 128-bit key for encryption. This encryption is combined with SHA-256 as hash function to provide data integrity and authentication. The delays caused by AES-128 and SHA-256 are negligible. The size of the packets is altered by the protocol specifications for ESP packets and the use of tunnel mode. Tunnel mode introduces and additional IP header (typically 20 bytes) because of the encapsulation. The format of an ESP packet adds another 8 bytes of protocol parameters and 32 bytes for the integrity check value.

The results obtained on IPsec encrypted data are summarized in Table II. We again find that our features contain sufficient information to allow a high classification accuracy.

C. TOR encrypted data

TOR is more complex than IPsec since it uses a network of Tor relay nodes to provide anonymity to its users. Table III shows our results on TOR encrypted data. As expected the accuracy is much lower compared to IPsec but we still achieve an accuracy of up to 86%.

These results are again explained by looking at the impact of TOR on the packet sizes and timing characteristics. The effect on the timing is twofold. First of all, there is the longer path that is taken by Tor because of the three relay nodes through which all packets are routed. This circuit changes regularly [4], resulting in different delays. The effect of TOR on the packet sizes is caused by the TLS encryption added by each of the relay nodes which make up the circuit.

Accuracy	Naive Bayes	Logistic Regression	Random Forest
Size - IAT	80.56%	77.78%	84.72%
Burst features	86.11%	80.56%	86.11%
Combined	83.33%	80.56%	80.56%

Table III: Classification accuracy for the different feature spaces for three different machine learning models using data from TOR encrypted streams.

D. The role of background traffic

So far we have only considered the ideal situation in which all captured packets originate from a single application. In a more realistic scenario however, several background processes may be running and sending/receiving packets as well. We mimicked a more realistic scenario by replaying background packets. We examined the effect of different background packet rates on the classification accuracy for IPsec encrypted traffic. We only report the results for the Random Forest classifier since this classifier consistently performed best on encrypted data. Figure 3 shows that the accuracy drops as we increase the background traffic rate but even for a rate of 300 background packets/s we are still able to achieve an accuracy of 87.5%.

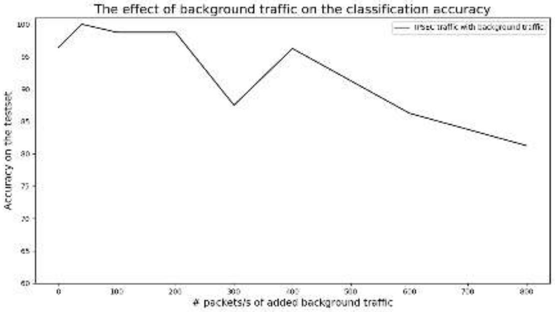


Figure 3: The effect of background traffic on the classification accuracy.

E. Multi label classification:

In a real world application it is possible that a single window contains packets belonging to different applications. A person can watch YouTube videos while downloading content via a Bittorrent client for example. We now extend our approach to multi-label classification where every window can have one up to four labels.

Due to the nature of the chosen feature spaces, the histograms for multi-labeled samples are not simply the addition of their single-labeled components. Although packet sizes might show this behaviour, inter-arrival times do not. They tend to shorten as packets from different applications interleave each other. Also burst from one application may be interrupted by packets from the other application.

We summarized the results in Table IV. We only report the results for the Random Forest classifier since this classifier consistently performed best on encrypted data. In addition to the accuracy we also report the Hamming loss. With multi-label classification it is possible to have partially correct results. The Hamming loss is defined as the fraction of the wrong labels to the total number of labels. A Hamming loss of 4% for example means that on average, 4% of the predicted labels are incorrect.

Accuracy	Accuracy	Hamming loss
Size - IAT	83.87%	4.62%
Burst features	76.83%	6.89%
Combined	83.58%	4.62%

Table IV: Classification accuracy and Hamming loss for the different feature spaces for the random forest classifier using data from TOR encrypted streams.

V. Conclusion and future work

We have shown that size and timing features, at both individual packet level and at aggregated burst level, can be successfully used to fingerprint encrypted network traffic types. We first validated our approach on an unencrypted baseline. IPsec did not alter the features significantly and similar accuracy measures were obtained. TOR on the other hand does have an effect on the timing characteristics of a network stream because of the circuit of relay nodes that it uses in order to provide anonymity. This resulted in a noticeable accuracy drop. We do however believe that more data, captured from different TOR circuits can make our models more robust against the delay introduced by the protocol.

In future work we will look at more real-world settings with a wider variety of network protocols. We will increase our dataset to include traffic from different devices with different operating systems and network stacks.

Acknowledgements

The work presented in this paper was performed by Nathan Meheus during his Master of Science in Computer Science Engineering thesis dissertation. Steven Bohez is funded by a Ph.D. grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

[1] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, “Internet traffic classification demystified: myths, caveats, and the best practices,” in *Proceedings of the 2008 ACM CoNEXT conference*. ACM, 2008, p. 11.

[2] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, “Dynamic application-layer protocol analysis for network intrusion detection.” in *USENIX Security Symposium*, 2006, pp. 257–272.

[3] A. W. Moore and K. Papagiannaki, “Toward the accurate identification of network applications.” in *PAM*, vol. 5. Springer, 2005, pp. 41–54.

[4] R. Dingledine, N. Mathewson, and P. Syverson, “Tor: The second-generation onion router,” Naval Research Lab Washington DC, Tech. Rep., 2004.

[5] P. Winter, R. Kowinter2014spoiledwer, M. Mulazzani, M. Huber, S. Schrittwieser, S. Lindskog, and E. Weippl, “Spoiled onions: Exposing malicious tor exit relays,” in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2014, pp. 304–331.

[6] G. Lu, H. Zhang, M. Qassrawi, and X. Yu, “Comparison and analysis of flow features at the packet level for traffic classification,” in *Connected Vehicles and Expo (ICCVE), 2012 International Conference on*. IEEE, 2012, pp. 262–267.

[7] Y. Shi and S. Biswas, “Website fingerprinting using traffic analysis of dynamic webpages,” in *Global Communications Conference (GLOBECOM), 2014 IEEE*. IEEE, 2014, pp. 557–563.

[8] M. Liberatore and B. N. Levine, “Inferring the source of encrypted http connections,” in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 255–263.

[9] M. Lotfollahi, R. Shirali, M. J. Siavoshani, and M. Saberian, “Deep packet: A novel approach for encrypted traffic classification using deep learning,” *arXiv preprint arXiv:1709.02656*, 2017.

[10] “Experimental defense for website traffic fingerprinting,” https://blog.torproject.org/experimental-defense-website-traffic-fingerprinting, accessed: 2017-09-29.

[11] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, “Touching from a distance: Website fingerprinting attacks and defenses,” in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 605–616.

[12] C. V. Wright, S. E. Coull, and F. Monrose, “Traffic morphing: An efficient defense against statistical traffic analysis.” in *NDSS*, vol. 9, 2009.

[13] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, “Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail,” in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 332–346.

[14] S. Feghhi and D. J. Leith, “A web traffic analysis attack using only timing information,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1747–1759, 2016.