Dublin City University School of Computing, Dublin, Ireland

MCM1 Practicum CA694: Literature Review

Video Identification Based on Encrypted Network Traces and Attack Mitigation

Deborah Djon deborah.djon3@mail.dcu.ie 22261972
Darragh Connaughton darragh.connaughton5@mail.dcu.ie 13469978
Supervisor: Geoff Hamilton

I. Introduction

Video streaming is gaining popularity. This is evident in the rise of streaming services such as YouTube, Netflix, and Disney+. Simultaneously the Hypertext Transfer Protocol Secure (HTTPS) is adopted as a standard for protecting internet communications. Encrypted network traffic protects users' privacy by rendering content unreadable to all but the holder of the decryption key. However, stakeholders, such as internet service providers (ISP), may wish to keep sensitive content out of their network. Studies show that ISPs can achieve this goal, despite encryption [1], [2]. Although encryption technologies such as Transport Layer Security (TLS), virtual private networks (VPN), and The Onion Router (TOR) render video identification through deep packet inspection (DPI) ineffective, videos can still be identified using traffic analysis side-channel attacks (SCA). Major streaming providers use a streaming content delivery technique called Dynamic Adaptive Streaming over HTTP (DASH). DASH improves users' quality of experience (QoE) and the streaming service's Google Page Rank [3]. Although it is effective at improving the users' QoE, DASH's design leaves a unique pattern in streaming traffic traces. Previous studies use traffic analysis to infer video fingerprints [4], [2], [5], [1], [6].

This literature review aims at outlining existing methods for identifying videos in encrypted network traffic and evaluating how they can be improved. In addition, we outline the various adversarial models leveraging the identification technique and potential mitigation techniques. To achieve this goal, we first identify research gaps in the field of video identification using encrypted network traffic, by answering the following research questions:

1. What are adversarial models for performing video identification traffic analysis side-channel attacks on encrypted network traffic?
2. What are video fingerprinting techniques?
3. What methods for video identification?
4. How practical are the proposed solutions?

In the following sections, we first provide relevant background information (section II). We then critically review existing literature on the proposed topic to answer the above research questions (section III-V). In section VI, we explore know mitigation's to this type of attack from both the streaming service providers and local LAN administrators perspective. The last section VII summarises the found research gaps. In addition, it outlines how our practicum aims to fill one or more of these research gaps.

II. Background

a. Dynamic Adaptive Streaming over HTTP

Dynamic Adaptive Streaming over HTTP is an implementation of adaptive bitrate streaming (ABS), which seeks to maintain the user's QoE. With ABS videos are encoded in multiple distinct representations of varying quality levels with each representation segmented across time [7], [2]. The sections are described by an extensible markup language (XML) media presentation description (MPD) file containing the segment link index [8]. A streaming session is a series of sequential client-initiated segment requests. It can be divided into an initial playback buffer hydration stage followed by a

steady-state stage where data is transferred at regular intervals [9].

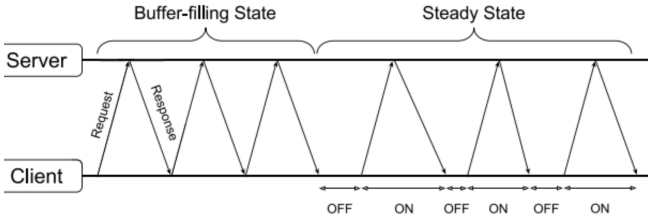

Fig. 1: DASH streaming session states.[8]

Streaming session health signals, such as the client playback buffer state and network bandwidth, are used in conjunction with the video's MPD file to determine the appropriate quality level of the next segment during the segment request process [7], [2].

DASH uses the H.264 industry standard for video compression. H.264 encoding reduces the size of the digital video through the removal of redundancy within and between frames among other techniques [10]. H.264 encoding divides frames in Macroblocks of N x N pixels. Prediction is used to find similarities between Macroblocks within and across adjacent frames [11]. The encoding process results in a unique representation across videos based on video-specific characteristics. There is a correlation between on-disk segments and network traffic patterns when DASH enters into the steady-state phase. The effects of encoding on the segment size is clearly seen below in figure 2 where high and low action scenes have been juxtaposed.

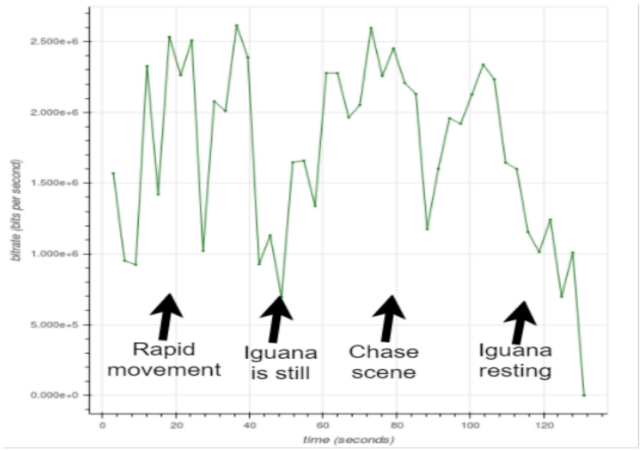

Fig. 2: Bitrate of the "Iguana vs. Snakes" video.[8]

Segmentation leads to content identifiable burst patterns in network traffic. The sequential and segmented nature of the data transmission leaks identifiable burst patterns in the network traffic which can be used to identify videos even in encrypted traffic [2].

III. Video Fingerprinting

Fingerprinting is a process for representing data in a smaller subset of symbols [12, pp. 143–152.]. In streaming traffic analysis, a fingerprint uniquely identifies a video. The following paragraphs describe and compare data collection methods and data formats for video fingerprinting using network traces. Further, different fingerprinting techniques are described and contrasted.

a. Data

1. Data Collection Methods

The authors in papers [1], [5], [13], [14], use a combination of browser automation tools, predominantly Selenium [15], coupled with a network capture utility, predominantly some variation of Tshark [16], to collect data. Data gathered using this approach will have variances in bitrate, caused by variable network conditions, across each video trace in line with the traffic captured by an adversary. Neither of the authors in [5], [1] reference YouTube premium nor a mechanism which would allow them to skip advertising at the start of the video. Failure to properly mitigate this factor means that each trace may contain a variable number of seconds of dummy data at the beginning.

The authors in [2] and [8] used a web-crawling in conjunction with online download utility to download YouTube videos. Each video was then streamed using a local implementation of an open source DASH server. Network conditions between a local client and a local DASH server are likely to be more stable as this is a controlled environment and thus likely to experience a greatly reduce range of potential network conditions, leading to a close to constant bitrate variance across each video trace. To counter this, the author's in [8] ensured the bitrate variance of each video trace was statistically similar to that of YouTube. In contrast, author's in [2] opted for a constant bitrate across each video but streamed each video in three bitrates. Training a video identification model on data that does not reflect the variable bitrate effects of DASH streaming may render the model ineffective in a real-world application. This is because the model is overfitting to closely match the train-

ing data and is not robust enough to find patterns in noisy data.

Video traffic traces can be collected from various sources, YouTube is a prominent source among the reviewed papers. However, other sources such as Netflix, and Facebook have also been explored [17], [8] The videos consist of content spanning several genres such as film, sport, games, or music.

2. Data Formats

As for the data formats, there are several data types and formats suitable for fingerprinting. One approach is extracting information about the application data unit (ADU) size. When streaming data using a DASH, ADUs are encapsulated in HTTP and TLS wrappers. Reed et al. retrieve the length of that ADU and use it for fingerprinting. [13] and [17] use this method. They attain the ADU information from metadata that is sent to the client from the streaming service provider to which they have access to. Their method is not applicable in cases, where there is no access to this metadata [8].

Most other studies including [5], [1], [18], [19], [8] [6] and [2] collect several minutes worth of video network traces in the form of number bits transmitted. The variances in the bitrate caused by DASH lead to drastically varying fingerprints for the same video and thus pose a challenge to video fingerprinting. Mitigation techniques for this issue are grouping the bit sequence into segments and regarding the difference between these differences. From the literature, it is not clear what the best segment size is. [2] perform experiments in a single bit-rate as well as a simulated variable bit-rate setting. In the fixed-bitrate setting, the segment length of four seconds resulted in higher model accuracy. However, they only test out the six-second segment length in the variable bitrate scenario and do not explain their reasoning. [5] adapted the six-second period length according to [2]. [18] reach an accuracy of 90 percent with a sub-second segment size of 0.5. More research must be conducted for identifying the best segment size.

[20] proposes grouping the bits into peaks rather than segments of a fixed time scale. This method is more flexible and robust to varying network conditions as different network traces of the same video may not perfectly align.

3. Dataset Statistics

The authors use many different data set sizes. Some authors, such as [1] and [5], use rather smaller datasets with around 50 classes (videos) for which they collect 50-100 traces. [18] have 100 videos with 200 traces each. All three authors capture traces for the first two to three minutes of the videos. The small data set is generally problematic for machine learning models, such as the ones used by [5], and [18] because they reacquire a large dataset to return reliable results [21], [22].

Also problematic about the small dataset is, that it does not allow for an open-world simulation. Most authors assume a closed-world scenario where it is assumed the attacker knows which videos can be watched. This can lead to a high false-positive rate in an open-world scenario [8]. Further research is needed to evaluate whether the proposed methodologies work in an open-world setting, where most of the videos are not known. m [13] have an extremely large database of 330,364 fingerprints from 42,027 videos stored in a 1.37 GB file. However, each fingerprint is based on a 20-second time interval starting at the beginning of the video. This may be too short for scenarios where you cannot perfectly determine when a video begins, and are trying to detect video segments far outside the first 20 seconds of a video.

b. Differential fingerprinting

To counteract variances in the streaming traces used for fingerprinting, some studies use a differential fingerprinting approach. Here, the fingerprints are not based on absolute bits transmitted per period, but on the differences between successively transmitted bits per period. This ensures lesser variance of fingerprints that result from different network traces of the same video.

The authors of [5] compare multiple differential fingerprint techniques. The first technique is the simple different fingerprints (SDF), shown in equation 1. In a sequence of bits per period a, the simple differential fingerprint creates a sequence of differences between one element a_i and it's predecessor a_i. Although this method reduces the variance between different fingerprints of the same video, it could be further reduced.

The differential fingerprint (DF) method further reduces the variance between fingerprints by dividing each sequence element difference by one of the

sequence elements. The differences are now scaled by the sequence element regarded, a_i, which further normalises the fingerprint. [2] also use the DF.

Lastly, the absolute differential fingerprint (ADF) is another fingerprinting method that eliminates negative values present in the SDF.

$$r_i = a_i - a_{i-1} \quad (1) \qquad r_i = |a_i - a_{i-1}| \quad (2)$$

$$r_i = f_{diff}(a_i, a_{i-1}) = \frac{a_i - a_{i-1}}{a_{i-1}} \quad (3)$$

In their studies, [5] compare the effectiveness of the three techniques described above. As shown in figure 3 the authors compared the accuracy of their model using SDF, ADF, and DF in non-VPN, VPN, and combined scenarios. ADF consistently underperformed. This makes sense, considering that using absolute values removes information in the form of the sign of the difference between two bits per period values. SDF and DF perform similarly well, although SDF is with an acuracy of 90 percent in the non-VPN setting about two percent better than DF. It is not clear if this difference is significant. Further experiments are needed to confirm the performance difference.

Dataset	SDF	ADF	DF
Non-VPN	90%	83%	88%
VPN	59%	52%	59%
VPN v Non-VPN	75%	68%	74%
Traffic (VPN v Non-VPN)	99%	98%	99%

Fig. 3: DASH streaming session states.[5]

c. Neural Network Fingerprinting

The authors of [1] use a convolutional neural network (CNN) for fingerprinting and detecting YouTube videos. They train the CNN with several traces of the same videos to automatically extract the features defining the video fingerprint. An advantage of automatic fingerprinting is that the model can identify features that may be overseen by a human-defined manual fingerprinting method. With their experiment, they attain about 12 percent higher accuracy than [5] in the non-VPN setting. This raises the question if a fingerprinting technique is necessary when using a CNN for automatic feature extraction. However, in the VPN setting, both [1] and [5] reach the same accuracy.

d. Segment Size

As described in section a "Data" some authors utilise the application data unit for fingerprinting. Wu et. al. aim at predicting the resolution of a YouTube video. For this, they use the ADU size as a fingerint that marks a certain resolution [17]. Reed et. al also make use of the ADU size. They create a database containing six-dimensional fingerprints that represent 30 consecutive ADUs.

IV. Video Identification

There are several methods for identifying fingerprints. Some are based on machine learning whereas others use purely statistical and algebraic methods. The following outlines methods used in previous works.

a. Dynamic Time Wrapping

Gu et. al. Lastly, use a modified version of dynamic time warping (DTW) for video identification. The task is to match the aggregate traffic pattern to the stored fingerprints. DTW solves this series matching problem by aligning the two sequences regarded by wrapping the temporal axis. They modify DTW to allow for partial matching [2]. They attain a high accuracy of 90 percent but do not use encryption.

With the proposed methodology, the authors achieve an accuracy of 90 percent. They are, however, most likely using unencrypted data, as Ethernet traffic is not encrypted. The authors do not state that they use any form of encryption. This opens the question of what the accuracy would be with encrypted network streams and if the accuracy can be improved through other methods, such as machine learning. The researchers use a closed-world scenario in which all videos needed for video identification are known prior. It is to be questioned what happens in an open-world scenario similar to [4], where a new video that has no fingerprint is compared to the videos in their fingerprint database. It is also questionable whether their fingerprinting technique is the most effective. As described in the next section, [5] describe a simple differential fingerprint (SDF) as being more effective.

1. Kd-Tree Search

Wu et al. implement a system for detecting Netflix videos in real-time traffic. This is different from all other papers that solely identify videos in an offline setting. The authors solve the problem of identifying videos based on a stream using kd-tree search. Using a 30 ADU window of data they compare the data to the six-dimensional keys in their database and get a number of potential matches in return.

b. Machine Learning

Several authors make use of machine learning for detecting fingerprints. The convolutional neural network is the most common one, being used by [5], [1], [18] and [17]. Their architectures are similar, with around five conv 1D and Max-pooling layers respectively. They also reach similar accuracies with a maximum of about 90 percent. [17] et al., compare different machine learning models, including CNN, random forest classification, gradient boosting regression, and XG boost classifier. Random forest performed the best with a maximal accuracy of around 80 percent. However, their objective is slightly different than the other papers (fingerprinting video quality, not the video itself); thus, comparing the accuracies is difficult. It is interesting to note that the authors of [1] may be working with a relatively small dataset but are able to archive high accuracy with this technique. This can have several reasons, including thwe data may have few underlying features and is therefore easy to classify, the evaluation metric is not appropriate or the model is overfitting. The first reason is supported by the fact that other models that do not use machine learning, but basic statistical models, yet, they can achieve similar accuracies. Understanding why the model performs well despite the small dataset is a research gap that needs to be explored. Potential research avenues are testing the model on a larger dataset and determining if the accuracy remains high and evaluating the model with more metrics such as recall and precision.

c. Website Fingerprinting

Video streaming traffic has specific attributes outlined in section a "Dynamic Adaptive Streaming over HTTP". However, considering the sequentiality of network traffic, looking at other fingerprinting methodologies that use encrypted network traffic can lead to new insights. For instance, the authors of [23] showed that a Long Short-Memory (LSTM) neural network worked best for their task of website fingerprinting. LSTM neural networks are generally suitable for temporal data. [4] also have promising results with stacked denoising auto-encoders (SDAE).

V. Adversarial Models

The adversary's approach to capturing live network traffic data for use in the classification system can be broadly categorised into two categories; on-path and off-path adversarial.

a. On-Path Attacks

In an on-path attack, an attacker has strategically positioned themselves as or on one of the intermediary hosts between the client and the server. Author's in [1], [5], [18] assume a man-in-the-middle (MITM) as either an ISP or a situation where the adversary convinces a client it is the default gateway using an address resolution protocol cache poisoning attack. The benefit of this approach is having direct access to the network data coupled with visibility into the network layer information (IP). Drawbacks include the practicality of performing the attack for an adversary within the victim's local area network (LAN). Performing an ARP poisoning attack floods the network with a recognisable pattern which may alert networking monitoring tools of the attacker's presence.

b. Off-Path Attacks

In an off-path attack, an attacker is able to discern a victim's network traffic data without having direct access to the network data. Access to and contention over a shared network resource used is a prerequisite for each of the approaches studied.

In both [24] and [25], the attacker is able to gather the victim's network traffic data without requiring a foothold on the victim's LAN. Instead, low-bandwidth, high-frequency probes are sent at constant intervals to the victim's LAN public router, with each probe being added to the router's queue upon arrival. Fluctuations in the round trip time (RTT) leak timing and volume information pertaining to the traffic passing through the router. This type of timing side-channel attack takes advantage of the fact that router scheduling policies are often more concerned with increasing through-

put and minimising delay than with preserving privacy. Throughput and fairness maximising routing in low traffic rate environments allows an attacker to completely reveal the victims traffic arrival pattern [26].

[24] were able to discern network traffic patterns of a victim in Illinois from New Jersey, whilst [25] were able to discern network traffic data from a victim in Quebec from New Jersey. Both authors assume a closed-world setting in which a LAN contained a single networked device with a single outbound connection. [24] sanitised the environment further by disabling the browser cache, automatic updates, and unnecessary plugins on the victim's browser. The degree to which additional network devices, browser sessions, or other network communications reduce the identifiable pattern of the victim's network traffic has yet to be explored.

A similar routing queue timing attack was performed in [8] but using from within the LAN. The adversary controls a malicious JavaScript advertisement running in an active browser on an adjacent network device to the victim. In a similar manner as [25], the attackers seek to saturate the shared network resource by sending a constant stream of data to the attacker-controlled service. Variance in the transmission time leaks traffic data of other users of the router.

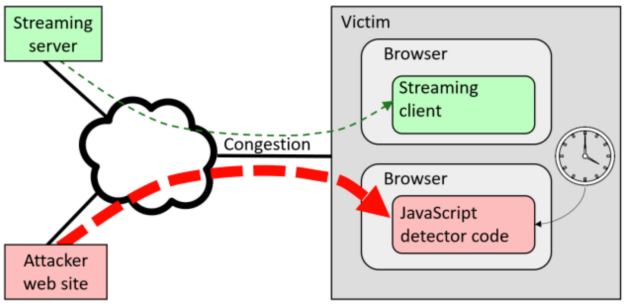

Fig. 4: Cross-site attack[8]

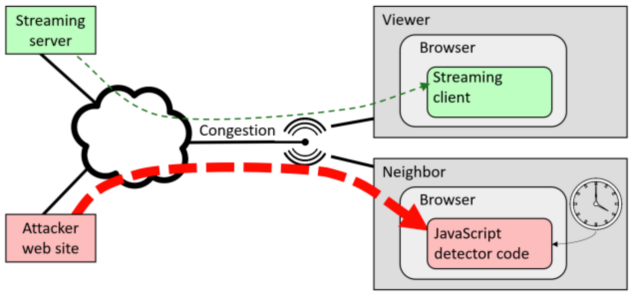

Fig. 5: Cross-device attack[8]

VI. Defences

a. Identifiable Pattern Obfuscation - Noise Generation

Streaming service providers can take steps to remove the identifiable network pattern in videos streamed using DASH. The authors in [2] used the introduction of noise to the packets, transmitting data out of sequence, variable segment length across the stream session, and bulk streaming multiple segments as means of hiding the network traffic pattern [27]. The Buffer Fixed-Length Obfuscation (BuFLO) was proposed [27] to obfuscate network traffic pattern by enforcing a fixed packet sizes and fixed packet transmission intervals. Both methods were able to significantly reduce the accuracy of a classifier trained to identify YouTube videos but incurred an 400 percent bandwidth overhead in pursuit of this goal. The bandwidth penalty likely nullifies any efficiency gained using DASH.

The authors in [18] challenged an assumption made by both of the previous authors, that packets leak identifiable information equally. By adding extra noise to packets deemed significant, the authors were able to reduce the classifier accuracy to that of random guess, even when trained on the obfuscated data, all while incurring lower bandwidth overhead than previous solutions. The disadvantage of targeted noise is the processing required to determine the significant packets, as they are unique to the video. Although manageable for cataloged streaming service providers like Netflix, the colossal scope of YouTube's backlog renders this method impractical.

b. Privacy Preserving Routing Schedules

The off-path timing attack which infers users' network traffic patterns through the introduction contention at the router level can be mitigated using privacy preserving routing schedules. The following mitigations are not compatible with the cross-site attack shown in 5 as resource contention in that attack scenario is introduced at the device's Network Interface, not at the LAN router.

The accumulate and preserve scheduling policy achieves privacy preservation using a two phased approach. During the initial accumulate phase, incoming data is buffered for a period of T time units. Traffic is subsequently served in the later

phase, but on a per user basis rather than in the order of arrival. Both the buffering period and distorted order of transmission effectively erases the fine grained timing information [28].

Time division multiple access (TDMA) is a router schedule which achieves maximum privacy preservation through the strict allocation of time slots for each network device. Traffic pertaining to a particular device can only be processed during the allocated time slot. The strict adherence to prescribed time slots leads to idle time which is a waste of networking resources. The performance cost incurred by idle time growth is proportional to the number of networked devices [28].

The authors in [29] propose variation of TDMA which allows privacy seeking users to opt in to a TDMA like mode, but allows devices indifferent to privacy preservation to continue using throughput and fairness maximising routing schedules. Indifferent clients are given priority over those who wish to remain private, with necessary rate limiting measures taken to prevent starvation of the privacy inclined individuals.

The performance cost incurred implementing privacy preservation threatens the user's QoE, a core streaming metric. This trade off has yet to be explored in the literature.

VII. Conclusion

Off-path adversarial models in the literature assume an isolated user on the network interacting with a single internet service. Tolerance to network noise varies across adversarial models. A comparison of each model's performance in the face of incremental noise is a potential future research question.

Throughput maximising, delay minimising router schedules proved vulnerable to timing analysis attacks. Exploration of privacy maximising router scheduling policy's ability to effectively mitigate the timing attack is another potential avenue of exploration. Specifically, the research question would explore the trade-off between privacy preservation and the streaming session's QoE.

Analytical approaches such as the solutions proposed in [2], [6] are effective and yield high accuracy. However, [2] does not use encrypted data, and [6] is not stable. Both CNN methods are promising. However, they are yet to be tested on a larger data set. In addition, CNN the architectures can be improved using the insights from [30]. Lastly, it is also possible that other deep learning methods, such as an LSTM neural network, are suitable.

References

[1] M. U. S. Khan, S. M. A. H. Bukhari, T. Maqsood, M. A. B. Fayyaz, D. Dancey, and R. Nawaz, "SCNN-Attack: A Side-Channel Attack to Identify YouTube Videos in a VPN and Non-VPN Network Traffic," *Electronics*, vol. 11, no. 3, p. 350, Jan. 2022.

[2] J. Gu, J. Wang, Z. Yu, and K. Shen, "Traffic-Based Side-Channel Attack in Video Streaming," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 972–985, Jun. 2019.

[3] M. Michalos, S. Kessanidis, and S. Nalmpantis, "Dynamic adaptive streaming over HTTP," *Journal of Engineering Science and Technology Review*, vol. 5, pp. 30–34, Jun. 2012.

[4] V. Rimmer, D. Preuveneers, M. Juarez, T. Van Goethem, and W. Joosen, "Automated Website Fingerprinting through Deep Learning," in *Proceedings 2018 Network and Distributed System Security Symposium*, 2018.

[5] W. Afandi, S. M. A. H. Bukhari, M. U. S. Khan, T. Maqsood, and S. U. Khan, "Fingerprinting Technique for YouTube Videos Identification in Network Traffic," *IEEE Access*, vol. 10, pp. 76 731–76 741, 2022.

[6] L. Yang, S. Fu, Y. Luo, and J. Shi, "Markov Probability Fingerprints: A Method for Identifying Encrypted Video Traffic," in *2020 16th International Conference on Mobility, Sensing and Networking (Msn 2020)*. Los Alamitos: Ieee Computer Soc, 2020, pp. 283–290.

[7] K. Ragimova, V. Loginov, and E. Khorov, "Analysis of YouTube DASH traffic," in *2019 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, 2019, pp. 1–5.

[8] R. Schuster, V. Shmatikov, and E. Tromer, "Beauty and the burst: Remote identification of encrypted video streams," in *Proceedings of the 26th USENIX Conference on Security Symposium*, ser. SEC'17. USA: USENIX Association, 2017, pp. 1357–1374.

[9] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2019.

[10] A. Mankodia, S. Bhatt, R. Soni, H. Mevada, and V. Dwivedi, "H.264: An emerging standard for advanced video coding," Apr. 2008.

[11] J.-W. Chen, C.-Y. Kao, and Y.-L. Lin, "Introduction to H.264 advanced video coding," in *Asia and South Pacific Conference on Design Automation, 2006.*, 2006, pp. 6 pp.–.

[12] A. Z. Broder, "Some applications of Rabin's fingerprinting method," in *Sequences II*, R. Capocelli, A. De Santis, and U. Vaccaro, Eds. New York, NY: Springer New York, 1993, pp. 143–152.

[13] A. Reed and M. Kranch, "Identifying HTTPS-Protected Netflix Videos in Real-Time," in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*. Scottsdale Arizona USA: ACM, Mar. 2017, pp. 361–368.

[14] A. Reed and B. Klimkowski, "Leaky streams: Identifying variable bitrate DASH videos streamed over encrypted 802.11n connections," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. Las Vegas, NV, USA: IEEE, Jan. 2016, pp. 1107–1112.

[15] "Selenium," Software Freedom Conservancy.

[16] R. Jacobs, "Tshark | tshark.dev."

[17] H. Wu, X. Li, G. Wang, G. Cheng, and X. Hu, "Resolution Identification of Encrypted Video Streaming Based on HTTP/2 Features," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 19, no. 2, pp. 1–23, May 2023.

[18] H. Li, B. Niu, and B. Wang, "SmartSwitch: Efficient Traffic Obfuscation Against Stream Fingerprinting," in *Security and Privacy in Communication Networks*, N. Park, K. Sun, S. Foresti, K. Butler, and N. Saxena, Eds. Cham: Springer International Publishing, 2020, vol. 335, pp. 255–275.

[19] X. Zhang, J. Hamm, M. K. Reiter, and Y. Zhang, "Statistical Privacy for Streaming Traffic," in *Proceedings 2019 Network and Distributed System Security Symposium*. San Diego, CA: Internet Society, 2019.

[20] R. Dubin, A. Dvir, O. Pele, and O. Hadar, "I Know What You Saw Last Minute—Encrypted HTTP Adaptive Video Streaming Title Classification," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 3039–3049, Dec. 2017.

[21] F. Konietschke, K. Schwab, and M. Pauly, "Small sample sizes: A big data problem in high-dimensional data analysis," *Statistical Methods in Medical Research*, vol. 30, no. 3, pp. 687–701, Mar. 2021.

[22] D. Rajput, W.-J. Wang, and C.-C. Chen, "Evaluation of a decided sample size in machine learning applications," *BMC Bioinformatics*, vol. 24, no. 1, p. 48, Feb. 2023.

[23] A. Shusterman, L. Kang, Y. Haskal, Y. Meltser, P. Mittal, Y. Oren, and Y. Yarom, "Robust Website Fingerprinting Through the Cache Occupancy Channel," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 639–656.

[24] X. Gong, N. Kiyavash, and N. Borisov, "Fingerprinting websites using remote traffic analysis," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, ser. CCS '10. New York, NY, USA: Association for Computing Machinery, Oct. 2010, pp. 684–686.

[25] S. Kadloor, X. Gong, N. Kiyavash, T. Tezcan, and N. Borisov, *Low-Cost Side Channel Remote Traffic Analysis Attack in Packet Networks*, Jun. 2010.

[26] X. Gong and N. Kiyavash, "Quantifying the Information Leakage in Timing Side Channels in Deterministic Work-Conserving Schedulers," May 2014.

[27] 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 *2012 IEEE Symposium on Security and Privacy*, 2012, pp. 332–346.

[28] S. Kadloor, N. Kiyavash, and P. Venkitasubramaniam, "Mitigating Timing Side Channel in Shared Schedulers," *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1562–1573, Jun. 2016.

[29] A. Beams, S. Kannan, and S. Angel, "Packet Scheduling with Optional Client Privacy," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. Virtual Event Republic of Korea: ACM, Nov. 2021, pp. 3415–3430.

[30] Q. Zhu and X. Zu, "Fully Convolutional Neural Network Structure and Its Loss Function for Image Classification," *Ieee Access*, vol. 10, pp. 35 541–35 549, 2022.