Understanding and Improving Video Fingerprinting Attack Accuracy under Challenging Conditions

August Carlson
Linköping University
Linköping, Sweden

David Hasselquist*
Linköping University
Sectra Communications
Linköping, Sweden

Ethan Witwer
Linköping University
Linköping, Sweden

Niklas Johansson
Linköping University
Sectra Communications
Linköping, Sweden

Niklas Carlsson
Linköping University
Linköping, Sweden

Abstract

The threat of video fingerprinting attacks poses significant privacy concerns. These attacks can identify streamed videos with high accuracy despite the use of encryption, leveraging both heuristic-based and deep learning techniques. However, the real-world effectiveness of such attacks remains underexplored, as most research assumes ideal conditions. In this paper, we address the challenges posed by variable network conditions and live-streaming latency, which complicate the attacker's ability to collect useful training data. First, we evaluate several deep learning model architectures against video data under diverse network conditions, including two adaptations of existing website fingerprinting attacks tailored to video that we show boast notable improvements over the base attacks and previous state-of-the-art video fingerprinting attacks. Second, we introduce two augmentation techniques and demonstrate that they substantially enhance attack performance in suboptimal conditions, without knowledge of the victim's live latency. Finally, we analyze the effects of data limitations such as observation time, dataset size, and training time. Overall, our work provides new insights into the impact that several real-world challenges have on attack accuracy, presents new and improved attacks, and details two augmentation techniques that can further boost the performance of the new attacks. Combined, these significant advancements highlight the urgent need for effective defense mechanisms.

CCS Concepts

• **Networks** → **Network privacy and anonymity**; • **Security and privacy** → **Web protocol security**; **Network security**.

Keywords

traffic analysis, video fingerprinting, challenging conditions

*Corresponding Author: David Hasselquist, david.hasselquist@liu.se

WPES '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-1239-5/24/10
https://doi.org/10.1145/3689943.3695045

ACM Reference Format:
August Carlson, David Hasselquist, Ethan Witwer, Niklas Johansson, and Niklas Carlsson. 2024. Understanding and Improving Video Fingerprinting Attack Accuracy under Challenging Conditions. In *Proceedings of the 23rd Workshop on Privacy in the Electronic Society (WPES '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3689943.3695045

1 Introduction

People and society at large are increasingly reliant on online video: streaming now accounts for the majority of Internet traffic [58] and represents a major source of many people's daily news [36, 44, 49] and other information. At the same time, it has been demonstrated that *video fingerprinting* attacks can identify streamed videos with high accuracy. Even simple heuristic-based attacks are rather effective [23, 55, 56], and deep learning techniques further improve the prospects for an attacker [1, 4, 40, 59]. This has concerning implications for individuals who desire privacy as well as organizations and governments that require secure communications.

Despite the clear threat that video fingerprinting attacks pose, there is, as of today, limited understanding of how successful such attacks can be in real-world scenarios, as most video fingerprinting research assumes ideal attack settings. Though some authors have analyzed, for example, the inherent difficulty of collecting datasets that afford high classifier accuracy when targeting users that have different setups and network conditions [10, 32, 34], these works focus specifically on website fingerprinting. Due to the nature of streaming traffic and its dependence on client and network state, video fingerprinting presents particular challenges that both amplify and transcend those considered in existing studies.

Almost all video players employ some form of adaptive bitrate streaming, a performance optimization in which the requested video content varies based on, for example, the client's buffer status and available network capacity. The result is a unique traffic pattern – which is highly dependent on network conditions – when the *same video* is watched multiple times. This represents a substantial challenge for an adversary, especially one that is unable to collect training data under the same conditions as the victim. Furthermore, in the case of live streaming, variations in live latency (the delay between content distribution and playback) can cause temporal offsets between the adversary's training data and monitored streams.

Contributions: We take three major steps to address these unconsidered challenges. First, to better understand the impact that

variable network conditions and the victim's live latency have on attacks, we evaluate several different deep learning model architectures against video data collected in various network conditions. These include the previous state-of-the-art video fingerprinting attack; two noteworthy website fingerprinting attacks; and custom adaptations of these attacks, which feature video-tailored input formats and boast notable improvements over the base attacks and existing video fingerprinting attacks.

Second, to address the shortcomings of the attacks in difficult conditions, we present two augmentation approaches that significantly increase performance (1) in suboptimal and variable network conditions and (2) when the adversary does not have knowledge of the victim's live latency. The augmentations can be used both separately or in combination, and we show that they provide significant benefits when used along with our adapted attacks. They also offer broad insights into how video fingerprinting models can be improved to account for protocol-specific features, which may be useful when targeting different streaming standards or developing, for example, zero-shot classifiers for live streaming.

Finally, we complement our evaluations by considering the impact of data limitations, such as observation time, dataset size, and training time. These results highlight that the attacks only need limited samples over relatively short time durations (compared to a typical video streaming session) to achieve high attack accuracy and that the accuracy of our adapted attacks increases considerably with the number of epochs used during training, allowing us to significantly outperform the previous state-of-the-art.

Outline: Section 2 provides background on traffic analysis and video streaming. Section 3 presents the challenges and limitations examined in the paper. Sections 4 and 5 introduce the datasets and attacks used in the paper, followed by Section 6, which presents evaluations of the attacks in a variety of challenging conditions. Section 7 describes two augmentation approaches to mitigate the impact of the challenging conditions, and Section 8 portrays the effects of data limitations. Section 9 concludes the paper.

2 Background and Related Work

2.1 Traffic Analysis

In the context of computer networks, *traffic analysis* refers primarily to the deduction of information from encrypted data streams. One prominent application involves determining the contents of a stream, a technique known as fingerprinting.

Website Fingerprinting: A particularly well-studied example is *website fingerprinting* (WF), in which a local, passive adversary attempts to deduce which websites a targeted user is visiting by monitoring their connection to an anonymous communications service, such as a VPN or Tor [15]. The adversary typically collects network traffic traces consisting of packet metadata, such as directions, timestamps, and sizes, and uses them to train a classifier. In the rudimentary *closed-world* evaluation setting, the traces collected correspond to websites of interest, and the user is assumed to only ever visit those websites. Conversely, in the more realistic *open-world* setting, traces must be gathered for a *monitored set* of websites and an *unmonitored set*, with examples of websites that the adversary is not interested in. After training, the adversary classifies traces collected from the victim's connection.

While primitive WF attacks [27, 29, 30, 47] involved heuristic algorithms and basic machine learning techniques applied to a plethora of manually crafted features, modern attacks [7, 14, 33, 54, 57, 60, 62] are based on deep learning and operate on low-level representations of raw network traces. Rimmer et al. [57] were the first to use a Convolutional Neural Network (CNN) to create a WF attack that surpassed the previous state-of-the-art. Since then, CNNs and other deep learning models have become a major focus, with continual improvements thanks to evolving deep learning architectures. Two noteworthy examples are Deep Fingerprinting (DF) [62], which achieves high accuracy using only sequences of packet directions; and the more recent Robust Fingerprinting (RF) [60], which improves over DF by employing time series of packet counts. In this work, we evaluate the viability of these two architectures against video streams and adapt them to improve their effectiveness.

In tandem with the development of better attacks, researchers have begun to investigate the real-world challenges that hinder their viability [10, 32, 34] and presented techniques to enhance classifier performance in difficult scenarios [5, 42]. Much research effort has also been invested in defending against the WF threat: several different defense approaches, such as random distortion [20, 35, 50], trace regularization [9, 18, 31], imitation [45, 69, 70], adversarial machine learning [21, 43, 53], and traffic splitting [13, 28] have been evaluated, each with its own unique tradeoff between overhead and protection [41]. Similarly, the continual arms race of attacks and defenses has led to the introduction of frameworks that allow for dynamic selection and synthesis of defenses [22, 48, 51, 72].

Other Applications: Traffic analysis can also harm user privacy in scenarios beyond website fingerprinting. These including identifying specific subpages visited within a website [24, 61]; determining messaging traffic details and app usage [6, 12, 19, 39, 65]; extracting VoIP transcripts and voice assistant activity [2, 3, 8, 37, 71]; fingerprinting live streams via associated chat traffic [25]; and more.

2.2 DASH Streaming

The most ubiquitous method of streaming video is Dynamic Adaptive Streaming over HTTP (DASH) [64], with major players such as Netflix [56] and YouTube [76] using variations of the standard. To host a video for DASH streaming, a service provider encodes it several times, each with a different target bitrate (quality). They then break up the produced representations into fixed-duration *segments*. A Media Presentation Description (MPD) containing details about the available segments at each quality is requested by clients wishing to stream the video; subsequently, segments are requested individually. The quality of the requested segments may change over time due to rules based on, e.g., throughput estimation and buffer state – this is known as *adaptive bitrate streaming* (ABR).

Critically, videos are typically encoded using a target *average* bitrate for performance reasons; this is known as *variable bitrate encoding* (VBR). As a result, all segments have the same duration, but their sizes depend on their content: segments corresponding to still images are small, while those containing significant motion or details are larger. As exemplified in Figure 1, this causes any given video to have a unique, identifying sequence of segment sizes, enabling video fingerprinting attacks. In this paper, we evaluate all attacks against DASH traffic with 2-second segments.

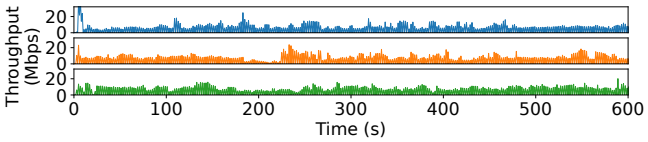


Figure 1: Throughput of 3 video streams using DASH.

Figure 2: Video fingerprinting threat model.

2.3 Video Fingerprinting

Video fingerprinting involves a local, passive adversary situated between a client and video server, whose goal is to identify the videos being watched over the encrypted connection. This scenario is depicted in Figure 2. While training data can be collected in advance for real-time classification of on-demand video, we focus on the more challenging scenario of live streaming. The experiments we perform mimic an adversary who watches multiple live streams to collect training data while *simultaneously* monitoring a victim's traffic, saving traces for later analysis. We remark that our contributions are also applicable to the video-on-demand setting, and we leave development of zero-shot classifiers for live streaming to future work, noting that the insights garnered from our augmentations may serve as a valuable component of such efforts.

Video fingerprinting differs from WF primarily because video traffic has rather distinct characteristics: one noteworthy difference is the substantial increase in the length of streams that are encountered in video fingerprinting, a result of the fact that loading a website typically only takes a few seconds, whereas watching a video is usually done during a much longer period of time. Also, since video streams involve individual segment transmissions, they differ markedly from website traces and can be far more distinctive. Perhaps more critically, video fingerprinting can in certain cases be in a closed world: if a user employs a direct connection to a video server, the set of possible videos can be narrowed down by observing, e.g., the destination IP address and DNS lookups. This would likely prove fruitful against providers that have a bounded and fairly stable video catalog; only make videos available for a limited time period (such as due to capacity, cost, or copyright reasons); or (when considering live streaming) have few concurrent streams. Video metadata may also be useful: for example, live streams with no viewers do not need to be considered by an adversary.

Similarly to WF attacks, many video fingerprinting attacks [16, 17, 23, 38, 55, 56, 73, 74] use heuristic algorithms and manually crafted features based on the periodic pattern of video traces. Though often highly effective, it has recently been shown that these attacks are not robust against defended traffic or in variable bandwidth conditions [26]. Attacks based on deep learning [1, 4, 40, 59], while opaque and less interpretable than simple heuristic attacks, show greater promise in these settings: the state-of-the-art attack, Beauty and the Burst (BnB) [59], achieves nearly perfect performance in a small closed world and excels even against defended traffic [26]. It is also effective with good bandwidth conditions in a larger open world [67]. Due to the attack's success and its foundational role for subsequent deep-learning based video fingerprinting work, we compare it against our adapted attacks throughout the paper.

Though a few defenses [11, 26, 66, 75] against video fingerprinting have been proposed, existing work has not emphasized the construction or circumvention of defenses. Instead, recent progress in the video fingerprinting community has been related to the variance of video traffic. For example, several authors [23, 76] have used heuristics to account for the effects of ABR on undefended traffic. We demonstrate how *augmentation* can improve results against video traces collected in different network conditions.

3 Challenges and Data Limitations

We begin by describing the real-world challenges that a video fingerprinting attacker may face. We focus on video-specific challenges while also considering some general fingerprinting challenges and their manifestations in the context of video fingerprinting.

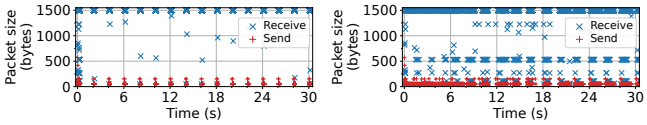
3.1 Primary Video Fingerprinting Challenges

C1) Adaptive Streaming and Varying Network Conditions: Almost all video streaming services today are implemented using DASH or a variant thereof, where performance is optimized by dynamically adjusting the quality of requested segments based on current network conditions and client capabilities (ABR).

While DASH ensures that users with good network conditions can obtain video data at the highest quality and clients with suboptimal connections can still attain seamless playback at a lower quality, the heterogeneity of video qualities due to quality switches throughout a streaming session presents a unique challenge to the video fingerprinting adversary, which WF attacks do not need to consider. First, as a consequence of the flexibility that DASH enables, the sequence of segment sizes may be quite different depending on network conditions. The adversary can therefore not rely on a unique segment size sequence to fingerprint a video stream.

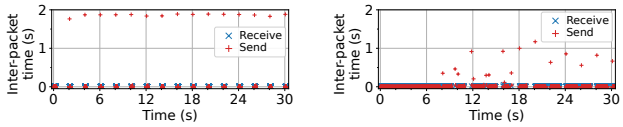
Second, variable bandwidth conditions can cause segment downloads to take longer, sometimes resulting in spillage, where a segment download time exceeds the segment duration and one segment download overlaps the next. This makes it more difficult to recognize when segment downloads start and end, as illustrated in Figures 3 and 4 using sample video traces. For example, looking at the timing of packets of different sizes (Figure 3), we observe clear bursts in the case of high constant bandwidth, regardless of whether we consider large received packets (from server to client) or smaller sent packets (requests and acknowledgments from client to server), whereas packets are less clearly distributed when bandwidth varies over time, making it hard to detect when a particular segment download starts and ends. This is also apparent when looking at the inter-packet delays for the constant bandwidth scenario (Figure 4(a)), where small values are typically observed within segment transmissions and long gaps are present between them. In contrast, in the variable bandwidth case (Figure 4(b)), there is no clear pattern, as many segment downloads are merged.

C2) Hard to Collect Training Data Under Same Network Conditions as Victim: Since an attacker can seldom run a video player from the same vantage point as the victim, network conditions typically differ between the training traces collected by the attacker and those gathered from a victim's connection. Thus, it is



(a) Constant bandwidth scenario (b) Variable bandwidth scenario

Figure 3: Comparison of packet sizes over time.



(a) Constant bandwidth scenario (b) Variable bandwidth scenario

Figure 4: Comparison of inter-packet delays over time.

important to understand the impact that such differences have on the accuracy of attacks. Due to the ABR properties of DASH, this challenge is especially demanding in the video streaming context.

C3) Different Live Latency: In the case of live streaming, clients will typically not share the same live latency (offset between content distribution and playback time). As a result, in contrast to the case of fingerprinting in the website and video-on-demand contexts, there is no exact starting point of a trace that can be used to align the attacker's training data with data collected from a victim's connection. As shown in this paper, this presents a unique challenge that can significantly impact attack accuracy.

3.2 Impact of Video-Related Data Limitations

Good training data is important to fairly and thoroughly evaluate attacks. Here, a central difference between the video fingerprinting and WF communities is their approach to limiting training data. WF attacks have been designed to achieve strong accuracy with few training samples [46, 63, 68]. In contrast, video fingerprinting attacks place a greater emphasis on the *length* of network traces, with the amount of streaming time required to correctly characterize a network trace playing a critical role [55, 56]. To understand the robustness of attacks, it is thus important to study the impact of *both* sample size and trace duration on video fingerprinting accuracy.

L1) Observation Periods: In contrast to website downloads, which typically happen over a short time period, clients may watch a video stream for an extended period of time. However, not all clients select to view for the same duration, and an attacker may not want to devote more resources than necessary to identify which stream a client is watching. To understand the impact that observation time has on prediction accuracy, we evaluate each considered model using traces of varying playback duration.

L2) Number of Training Samples: Similar to the website context, the number of training samples can play an important role in the accuracy of different models. To measure the effect of this, we evaluate each model under a range of sample sizes.

L3) Limited Computing Resources: With the best attacks being based on deep learning and involving significant training, it is important to understand how many epochs are needed to achieve acceptable accuracy and at what point increasing the number of epochs is detrimental to the accuracy of attacks due to overfitting. We provide insights into the influence of training time by evaluating each model with varying numbers of epochs.

3.3 Other Challenges

There are also a few general challenges not considered here.

Defenses: To protect the privacy of video streams, the client and/or server may employ defenses [11, 26, 66, 75]. In this paper, we do not consider defenses for several reasons. First, our focus is to understand the inherent vulnerabilities in video streaming traffic under challenging conditions without the influence of additional defensive measures. Second, the effectiveness of defenses varies widely, and incorporating them would introduce variability that could obscure the core findings. Finally, by excluding defenses, we provide a baseline analysis that can serve as a foundation for future studies that may incorporate and evaluate defenses.

Open World: Video fingerprinting can occur in a closed world, since a client may connect directly to a video server, resulting in a restricted set of potential videos. However, if an anonymous communications service such as a VPN or Tor [15] is used, or in the case of larger streaming services and those with volatile catalogs, the adversary may not have this advantage. Thus, the closed world is a realistic assumption for smaller video platforms with relatively static video catalogs or, in the case of live streaming, if the number of concurrent active streams is limited. We focus on the closed world in this work, as limiting the inherent difficulty of the evaluations better captures the extent to which our addressed challenges affect attacks. We note, though, that the insights we present may also prove valuable for future open-world studies.

4 Datasets

To evaluate how different network conditions may impact a video fingerprinting attacker, we use the *LongEnough* dataset and an extended version of the *LongEnough-variable* dataset [26]. *LongEnough* contains network traces of 100 live-streamed videos with a constant bandwidth of 100 Mbps. Each video is streamed for up to 10 minutes at 10 different starting points (60-second shifts), with 10 samples per shift, resulting in $100 \times \sum_{i=1}^{10} i \times 10 = 55\,000$ minutes $\approx$ 917 hours of streamed video traffic. *LongEnough-variable* is collected similarly but under variable bandwidth conditions. Each sample is gathered using a unique variable bandwidth pattern based on a real-world LTE trace [52], which is scaled by a factor of 8 to better align the peaks with the constant bandwidth case.

In this paper, we expand on the previously collected *LongEnough-variable* dataset [26] by collecting variable bandwidth traces scaled by factors of 1, 2, and 4. Each scale contains an additional 917 hours of streamed traffic. Regardless of the starting point of the stream, the last minute of data collection always contains the same video content; therefore, to obtain the maximum number of samples per video (i.e., 100), we extract and use only the last minute of each stream in our analysis. We note also that the videos in both *LongEnough* and the extended version of *LongEnough-variable* are streamed at up to three different available qualities: 1000 kbps (1K), 2000 kbps (2K), and 4000 kbps (4K). For a detailed description of the variable bandwidth conditions and dataset collection process, we refer the reader to Appendix B in Hasselquist et al. [26].

5 Evaluated Attacks

To evaluate the impact of the aforementioned challenges, we use five attacks. First, we evaluate the WF attacks DF [62] and RF [60]: since

they operate on low-level traffic representations, they are also applicable to video streams. Given the distinct nature of video streams, though, they do not reach maximum potential by default. To increase the attacks' suitability for video streams, we thus present two adaptations that retain the same model architectures. We also test the state-of-the-art video fingerprinting attack, Beauty and the Burst [59], using Hasselquist et al.'s implementation [26]. We use a 0.9-0.1 train/test split and five-fold cross-validation for all attacks.

5.1 WF Attacks

Deep Fingerprinting (DF): The first CNN-based WF attack that we evaluate is DF, presented by Sirinam et al. [62]. DF uses a 1×5 000 input matrix with a *directional* traffic representation: each index represents a single packet from the client's perspective (outgoing +1, incoming −1), for the first 5 000 packets in a trace. Shorter traces are padded with zeros, and longer traces are truncated.

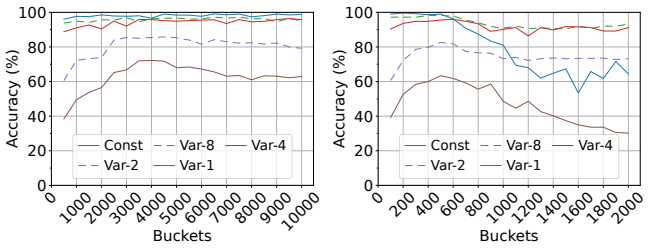
Robust Fingerprinting (RF): The second WF attack we evaluate is RF by Shen et al. [60]. RF incorporates the proven efficiency of CNNs for WF attacks while outperforming existing techniques by employing a new input format. Shen et al. [60] refer to this as the Traffic Aggregation Matrix (TAM), a matrix that separates packets into two different rows based on their directions and accumulates the number of packets within specific time intervals, or *buckets*, corresponding to the matrix columns. For WF, the size of the TAM has been tuned to divide 80 seconds of traffic into a 2×1 800 matrix, meaning that the buckets are 80/1 800 = 0.044 seconds long.

5.2 Video-Adapted Versions of WF Attacks

Video-Adapted DF (vDF): As shown by Hasselquist et al. [26], 5 000 packets capture only a few seconds of a video trace, and DF's performance can be significantly increased by using a larger input size. However, while Hasselquist et al. opt to increase the number of packets, we keep the model architecture of DF while completely replacing its input format. We employ a solution similar to RF's TAM: instead of using the first 5 000 packets, video traces are divided into 5 000 buckets, which each contain a sum of packet sizes. When considering 60-second traces, each bucket comprises the sum of packet sizes over 60/5 000 = 0.012 seconds.

Observe that, contrary to the original version of DF, we consider the sizes of packets in addition to their directions. Also, we do not limit the attack to 5 000 buckets; instead, we tune the number (and, consequently, duration) of buckets. Figure 5(a) depicts the accuracy of vDF as a function of the number of buckets each 60-second trace is split into for both the *LongEnough* and *LongEnough-variable* datasets. While accuracy remains fairly stable with high bandwidth, it increases, peaks around 4 000 buckets, and drops at bandwidth scales 1 and 2. This may suggest that low granularity can obscure informative traffic patterns, while very fine-grained buckets fail to capture important packet- and segment-level relationships. We select 4 000 buckets for the remainder of the paper, resulting in 60/4 000 = 0.015 second buckets for a 60-second trace.

Video-Adapted RF (vRF): As with DF, we adapt RF's input matrix to better utilize network traces. First, we find through preliminary evaluations that taking packet sizes into account has a positive effect, so we use packet size sums rather than packet counts in the buckets. Second, RF is tuned to encompass 80 seconds of traffic by default; we change this to 60 seconds to tailor the attack to our evaluations with the *LongEnough* and *LongEnough-variable* datasets. Finally, we determine the bucket duration that yields maximum accuracy; Figure 5(b) visualizes accuracy as a function of bucket count. In contrast to vDF, high granularity results in a significant accuracy reduction against constant bandwidth and scale 1, with less volatile performance in intermediate conditions. The reduction in accuracy against constant bandwidth and scale 1 may be accounted for by model architecture differences: vRF appears better equipped to handle high-level patterns than fine-grained information, as the attack is less stable with more buckets when segments are clearly delimited or with many rerequests and quality switches. Regardless, peak accuracy is around 400 buckets in every test, so we use this setting throughout the paper; the buckets are thus 60/400 = 0.15 seconds long for a 60-second trace.



(a) Video-Adapted DF (vDF) (b) Video-Adapted RF (vRF)

Figure 5: Accuracy of vDF/vRF with varying bucket count.

5.3 Video Fingerprinting Attack

Beauty and the Burst (BnB): Schuster et al. [59] propose BnB, a CNN-based video fingerprinting attack. BnB uses the same approach as RF in its representation of network traces, though its input matrix is 3×240 rather than 2×1 800: an extra row is present to combine incoming and outgoing packets, and buckets are 60/240 = 0.25 seconds long. Note that the attack is exhibited with different epoch values based on the service provider that the evaluation dataset is collected from. The values used are within the range [150, 1 400]; based on preliminary testing, we choose 500 epochs. This is significantly greater than the number of epochs used in the other four attacks: the original versions of DF and RF feature an epoch value of 30, which we elect to keep unchanged in most of our experiments to isolate the effects of our changes to the attacks' input formats.

6 Evaluated Conditions

We begin by evaluating the featured attacks in a multitude of varyingly challenging conditions.

6.1 Good Constant Bandwidth Conditions

Before considering each of the primary video fingerprinting challenges (C1–C3) outlined in Section 3.1, we compare attack performance in good conditions. Specifically, we direct our focus to the case in which the attacker can collect both training and evaluation data with the 100 Mbps constant bandwidth of the *LongEnough* dataset. With the exception of the default WF attacks (DF and RF), all attacks have greater than 98% accuracy: 93.2% (DF), 61.8% (RF), 98.2% (vDF), 99.3% (vRF), and 99.9% (BnB).

Main Observations: We note that all attacks perform very well in good conditions, with the exception of default RF - this is likely due to its fine-grained buckets, as discussed previously. Also, our adapted attacks (vDF and vRF) perform considerably better than their corresponding defaults (DF and RF). These results highlight the significant privacy threat that video fingerprinting attacks represent as well as the importance of application-specific attacks.

6.2 Known but Variable Bandwidth (C1)

As outlined in Section 3.1, the mechanics of DASH under variable bandwidth conditions bring forth a unique challenge in video fingerprinting: attacks typically leverage three low-level features – packet times, sizes, and directions – all of which are dependent on session-specific network conditions. First, segment transmission durations (and, thus, when the packets making up segments are delivered) are dependent on the current bandwidth available to the client. Second, packet sizes depend on the size of the segment being transmitted, which itself depends on the available bandwidth and how the client's ABR algorithm responds to it. Segment size also determines how many packets are needed to transmit a segment: two packets with the same index in different traces of the same video may be sent in different directions. Given the significant impact of bandwidth on network traces, it is important to understand how resilient attacks are to bandwidth variations.

We first consider an attacker targeting a user with known bandwidth conditions and assume the attacker can collect training data under similar conditions (e.g., traces with the same high-level bandwidth variability characteristics). As an example, the attacker could approximate the target's bandwidth conditions by measuring the throughput of collected network traces and gather training data accordingly. To benchmark our featured attacks in this setting, we use traces from the same bandwidth conditions for both training and evaluation. Table 1 presents the results for the *LongEnough* and *LongEnough-variable* datasets. Except in the case of default RF's lower accuracy against constant bandwidth, the attacks degrade as bandwidth conditions worsen. We also observe that our adapted versions (vDF and vRF) are most robust to poor network conditions.

Main Observations: Even if the attacker knows the bandwidth conditions of the victim, attack accuracy decreases markedly as network conditions worsen. This is likely because, while the training and evaluation data contain similar numbers of quality switches, their locations are not the same; this suggests that the attacks require more varied training data. Though not immune to this effect, our adapted attacks are most robust to poor network conditions. Also, as we show later in the paper, our adapted attacks can achieve further performance gains by way of additional training, allowing us to perform even better by increasing the number of epochs (Section 8.3) and/or using our augmentation-based extensions (Section 7). For example, vRF's accuracy against Var-1 increases from 58.5% to 86.0% when simply increasing epochs from 30 to 200.

6.3 Unknown and Variable Bandwidth (C2)

Beyond classifying traces when their bandwidth conditions are known, an attacker may be faced with the more challenging scenario of unknown bandwidth conditions. This means that the attacker is not able to guarantee that the training and evaluation data are collected under the same conditions. Such a situation may arise, for example, when bandwidth conditions fluctuate significantly over time or the attacker has multiple targets and simply cannot dedicate additional data collection and training effort to a single one. To test the attacks in this more challenging setting, we train on each dataset and evaluate on both the dataset used in training and all other datasets. Table 2 visualizes these results for our adapted attacks and BnB, as these perform best when faced with C1.

Table 1: Accuracy (%) under different bandwidth conditions.

	WF Attacks		Adapted Versions		
	DF	**RF**	**vDF**	**vRF**	**BnB**
Const	93.2	61.8	98.2	99.3	99.9
Var-8	89.6	79.8	96.6	97.8	97.6
Var-4	69.6	78.2	94.9	95.2	92.5
Var-2	39.4	69.0	85.9	79.8	73.7
Var-1	29.4	45.8	69.8	58.5	51.6

Main Observations: While we observe that non-trivial accuracies can be achieved when training and evaluating on traces from different network conditions, it is clear that the best accuracies are obtained under the same conditions as the targeted client and that performance worsens substantially with bigger differences in network conditions. These results highlight that lack of access to similar network conditions as the attacked client (C2) presents a significant challenge to the attacker. Nonetheless, we find that our adapted attacks (vDF and VRF) are generally more robust to being trained using the wrong bandwidth conditions than BnB, e.g., as seen by higher off-diagonal values. We note that, in such a case, it is typically better to train on traces collected under slightly worse conditions than slightly better conditions: too few quality switches in the training data can reduce the attacks' resilience to evaluation traces with more bandwidth variability, while an excessive amount obscures relationships between adjacent segments. Motivated by these observations, Section 7 presents bandwidth augmentation techniques addressing the challenge presented by not having access to traces collected under similar conditions as the client.

6.4 Offset (C3)

In live streaming, there is no guarantee that all clients are in sync, since live latency depends on, for example, the configuration of the video player and available network capacity. Thus, attacks that rely implicitly or explicitly on the timing of data relative to trace start may struggle. To evaluate the resilience of the attacks to this challenge, we apply offsets to the testing data.

To enable the application of offsets to our dataset, we utilize a 40-second sliding window that we move within the 60 seconds available from each trace. When no offset is applied, the middle 40 seconds are used; the preceding and following 10-second intervals are discarded. An offset of −10 slides the window to the first 40 seconds, and an offset of 10 corresponds to the last 40 seconds.

Figure 6 highlights the accuracy of each attack when *training* on the *LongEnough* dataset with no offset (i.e., using the data the attacker would see in their training sessions) but applying an offset to the *evaluation* data (capturing that the client may not have the same offset). The minimum and maximum attack accuracy are in the bottom right corner of each plot, using the format `min/max`.

Table 2: Accuracy (%) when training and evaluating under different bandwidth conditions.

(a) **vDF**

Training data ↓	Testing dataset				
	Const	Var-8	Var-4	Var-2	Var-1
Const	98.2	9.4	3.4	2.2	1.2
Var-8	96.7	96.6	85.0	39.2	15.0
Var-4	93.7	95.7	94.9	65.0	32.3
Var-2	20.7	91.1	92.6	85.9	54.2
Var-1	4.5	62.2	83.2	77.2	69.8

(b) **vRF**

Testing dataset				
Const	Var-8	Var-4	Var-2	Var-1
99.3	41.9	16.6	1.1	1.0
98.3	97.8	84.0	39.0	14.4
96.1	97.3	95.2	58.9	29.8
84.5	92.7	92.1	79.8	48.3
6.5	58.4	77.4	69.7	58.5

(c) **BnB**

Testing dataset				
Const	Var-8	Var-4	Var-2	Var-1
99.9	94.0	18.5	22.6	2.2
99.6	97.6	42.5	33.0	4.5
9.5	10.0	92.5	6.6	3.4
22.1	23.3	28.2	73.7	10.1
21.9	38.4	30.0	29.0	51.6

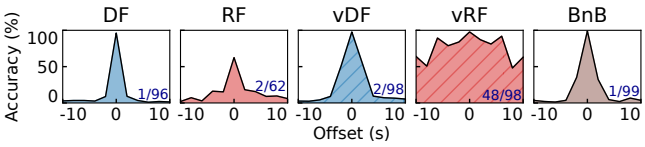


Figure 6: Accuracy when varying the evaluation offset.

Main Observations: All attacks except vRF degrade substantially even at small offsets. vRF is somewhat resilient to offsets, with accuracy staying well above that of the other attacks for offsets as large as 10 seconds: this may be due to the use of coarse-grained windows, which capture higher-lever information about traces. Also, as most clearly seen for vRF, accuracy does not decrease uniformly as offset increases, suggesting that vRF is more robust to temporal offsets of a wide range. However, the large drop in accuracy between the best and worst offsets suggests that data with varying offsets is needed during training. We address this through our offset augmentation techniques (Section 7).

6.5 Variable Bandwidth and Offset (C2 + C3)

Further complicating the challenges these suboptimal conditions present on their own, they might also be encountered at the same time. Since combining our previously examined conditions presents a vast amount of possible combinations and vRF is the only attack with a reasonable resilience to offsets, we only use vRF to test the combination of variable bandwidth and offsets. Figure 7 presents the results when training and testing on our featured datasets.

Main Observations: In general, we observe similar patterns as before: lower bandwidth scales are more challenging, and accuracy decreases non-uniformly as offset increases. It is still better to train on a slightly lower bandwidth scale than a higher one, but we note that the negative impact of large offsets is more pronounced at low bandwidth scales. As we show in Section 7.5, our two augmentation techniques can be combined to handle this challenge.

7 Augmentation-based Attack Extensions and Their Improvements

As seen in the previous section, two of the primary challenges we consider (C2 and C3) result in significant reductions in the accuracy of existing attacks. In the case of C2, it is important to have training data with similar numbers and locations of quality switches as the evaluation data. To solve C3, the attacks must be trained on traces with varying offsets. In light of these observations, we present two augmentation strategies (Sections 7.1 and 7.3) that successfully improve attack performance both separately (Sections 7.2 and 7.4) and in combination (Section 7.5).

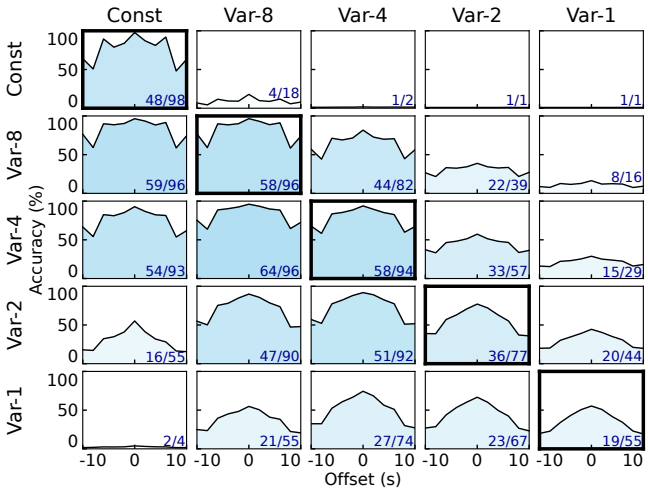


Figure 7: Accuracy of vRF when varying the evaluation offset. Here, training datasets are along the y-axis and evaluation datasets on the x-axis.

7.1 Variable Bandwidth Augmentation

To counteract the decrease in accuracy an attacker might face when targeting a client with an unreliable connection, we develop an augmentation strategy that morphs traces collected under stable conditions (i.e., with minimal quality switches) to imitate their variable bandwidth counterparts. At a high level, the augmentation transforms each trace in *LongEnough* into a sequence of (segment size, segment download start time) tuples. An arbitrary number of new, corresponding traces can then be generated by combining this information with values sampled from observed distributions of quality switches, segment transmission durations, and inter-packet delays. We first describe how we calculate these distributions.

Quality Switches: To calculate the desired frequency of quality switches in generated traces, we count their occurrences, along with the number of segments transmitted, for each bandwidth scale in *LongEnough-variable*. We also determine between which qualities switches should occur based on the observed switches. Figure 8 summarizes this data in the form of state machines, in which an action is taken every time a segment is transmitted.

Segment Transmission Durations: The bursty nature of video traces makes it easy to calculate, in the absence of spillage, how long it takes to transmit each segment. In the case of *LongEnough-variable*, though, poor bandwidth can cause segment transmissions to reach or even exceed 2 seconds. This creates a continuous data stream from which it is impossible to match packets with segments strictly based on packet timestamps. We address this by utilizing the predictable pattern of client-side requests: one or two small packets

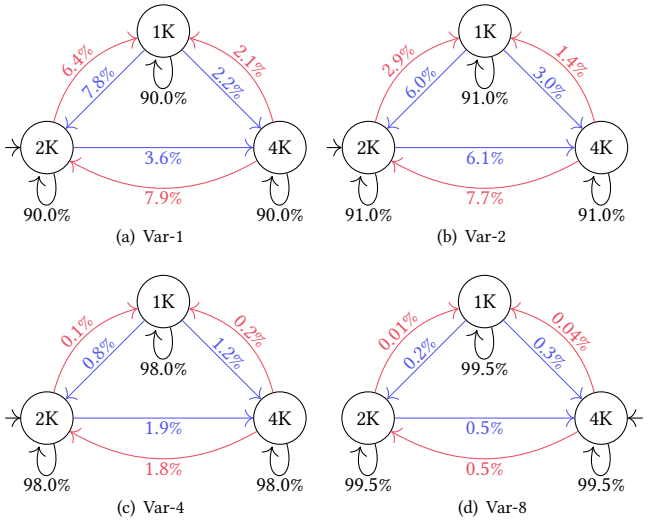


Figure 8: Quality switch state machines.

are sent by the client at the beginning of each segment transmission. This insight allows us to determine all segment transmission durations in the variable bandwidth datasets.

However, determining the start times of segment transmissions based on requests presents additional difficulties, specifically pertaining to rerequests. Rerequests occur when a segment transmission is partially completed but the available bandwidth changes drastically enough to warrant an immediate quality switch. To filter out rerequests, we use knowledge of how many requests a trace should contain. We iteratively remove one appropriate request at a time until the trace contains the correct amount. The request to be removed is chosen by calculating the time between each request and removing the request that is closest to its predecessor.

After filtering out rerequests, we calculate the durations of segment transmissions. We do this by calculating, for each segment, the time between the initial request and the last packet before the next segment request; for the last segment of each trace, we calculate the time between the request and the last packet in the trace. We iterate through each variable bandwidth dataset and use 1 of the 10 samples for each offset, storing the calculated segment transmission durations in a file. This results in $100 \times \sum_{i=1}^{10} i \times 1 \times 30 = 165\,000$ durations (one minute comprises 30 segments) per bandwidth scale.

Normalized Inter-Packet Delays: We also require inter-packet delays for the variable bandwidth datasets; we measure these simultaneously with segment transmission durations to avoid including times between segments. We also normalize the measured delays by, for each segment, calculating each inter-packet delay T as well as their average A and then storing the result of T/A. The last 60 seconds of a trace in the variable bandwidth datasets contain an average of 27 812 packets; thus, we collect approximately $100 \times \sum_{i=1}^{10} i \times 1 \times 27\,811 \approx 152\,960\,500$ normalized delays.

Augmentation Implementation: The augmentation is performed by iterating through the constant bandwidth traces. For each trace, the size of every segment transmission is accumulated based on the bursty nature of the traces. The timestamp of the first packet in each segment transmission is also stored, resulting in a list of (segment size, start time) tuples for each trace.

Table 3: Accuracy (%) of bandwidth augmentation and differences compared to training on constant bandwidth.

Attack	Training dataset	Testing dataset			
		Var-8	Var-4	Var-2	Var-1
vDF	Const	9.4	3.4	2.2	1.2
	Aug-X	77.4	74.0	57.6	32.2
		+723.4%	+2 076.5%	+2 518.2%	+2 583.3%
vRF	Const	41.9	16.6	1.1	1.0
	Aug-X	92.6	80.9	58.9	27.5
		+121.0%	+387.3%	+5 254.5%	+2 650.0%
BnB	Const	94.0	18.5	22.6	2.2
	Aug-X	28.7	31.4	4.7	2.8
		-69.5%	+69.7%	-79.2%	+27.3%

The tuples are then iterated through on a trace-by-trace basis, allowing for new, corresponding traces to be generated. During each iteration, the current size is first scaled based on a quality that is chosen according to the state machines presented in Figure 8. After the size S has been scaled, the number of packets N needed to transmit the desired amount of data is calculated via the formula $\lceil N = S/1\,448 \rceil$. The number 1 448 is derived by taking the maximum transmission unit (MTU) and subtracting the portion occupied by TCP overhead (O_{TCP}): $MTU - O_{TCP} = 1\,448$, giving the maximum amount of segment data that can fit in each packet.

This information is used to construct an artificial segment packet-by-packet. A random segment transmission duration T_S is selected from the collection gathered from the variable bandwidth dataset and used to calculate the average inter-packet delay: $T_P = T_S/N$. For each new packet, an inter-packet delay is chosen by scaling T_P using the normalized delays: a normalized value V is sampled and used to calculate the actual delay via the operation $T_P \leftarrow T_P \cdot V$. The start time for the current iteration determines when to send the segment. If the previous segment was completely transmitted before the current start time, we simply start transmitting at the stored start time. Otherwise, we start transmitting the segment when the previous segment transmission has completed.

7.2 Bandwidth Augmentation Evaluation

To determine the efficacy of our bandwidth augmentation, we generate four different augmented versions of the constant bandwidth dataset, one to target each bandwidth scale. Table 3 highlights the differences between training on the constant bandwidth dataset and appropriate augmented datasets (Aug-X) when evaluating on each variable bandwidth dataset. From here on, we consider only our adapted attacks, as they are most suitable for all addressed challenges, and BnB for comparison, since it is the prior state-of-the-art.

In all cases, though especially at lower bandwidth scales, our adapted attacks improve markedly as a result of the bandwidth augmentation. vDF's accuracy increases by 723.4% against Var-8 (9.4% to 77.4%), and this difference rises as bandwidth scale decreases, reaching 2 583.3% against Var-1 (1.2% to 32.2%). Similarly, vRF's accuracy is 121.0% higher against Var-8 with augmentation (41.9% to 92.6%), and it is boosted by 5 254.5% against Var-2 (1.1% to 58.9%) and 2 650.0% against Var-1 (1.0% to 27.5%). This shows that ensuring

Table 4: Accuracy (%) when training on the constant bandwidth and augmented datasets and evaluating on all datasets.

(a) **vDF**

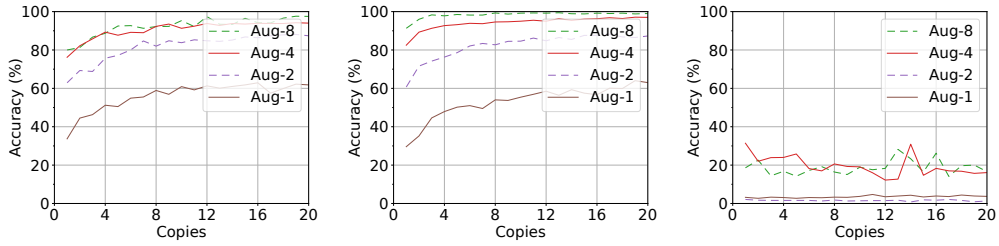
Training data ↓	Testing dataset				
	Const	Var-8	Var-4	Var-2	Var-1
Const	97.8	9.4	3.4	2.2	1.2
Aug-8	73.3	77.4	58.9	30.5	11.9
Aug-4	86.4	87.4	74.0	44.6	21.8
Aug-2	60.2	84.8	82.6	57.6	31.7
Aug-1	66.2	80.5	75.4	55.2	32.2

(b) **vRF**

Testing dataset				
Const	Var-8	Var-4	Var-2	Var-1
99.3	41.9	16.6	1.1	1.0
93.5	92.6	71.8	38.0	13.0
93.5	92.9	80.9	47.9	20.7
89.1	90.6	85.4	58.9	29.6
79.9	90.0	82.9	55.0	27.5

(c) **BnB**

Testing dataset				
Const	Var-8	Var-4	Var-2	Var-1
99.9	94.0	18.5	22.6	2.2
23.4	28.7	11.5	5.1	2.9
4.1	5.2	31.4	8.6	3.8
3.9	3.3	55.8	4.7	3.1
5.3	2.7	65.8	4.6	2.8



(a) vDF (b) vRF (c) BnB

Figure 9: Accuracy when varying the number of copies per sample for bandwidth augmentation.

similar numbers and locations of quality switches are present in the training and evaluation data significantly improves attack efficacy.

However, the results for BnB are mixed. Against Var-8, the highest bandwidth scale, its accuracy *decreases* by 69.5% with augmentation (94.0% to 28.7%). Its accuracy also decreases against Var-2, by 79.2% (22.6% to 4.7%), and improves only marginally against Var-1. The best accuracy is achieved when targeting Var-4, with a 69.7% improvement (18.5% to 31.4%). Nevertheless, it is clear that BnB is unable to take full advantage of the augmentation, likely because its very coarse-grained buckets and simpler model architecture are unable to completely represent the dynamics of quality switches.

In general, our adapted attacks (especially vRF) have rather high accuracy against Var-8, and accuracy decreases as bandwidth scale lowers. To further characterize this and to better understand the impact of the attacker's choice of augmentation option (i.e., Aug-X), Table 4 showcases the results of training on the constant bandwidth and augmented datasets and evaluating on every dataset. The same high-level trend holds: accuracy decreases at lower bandwidth scales. Interestingly, our adapted attacks achieve better accuracy when training for a lower bandwidth scale than the evaluation dataset vs. a higher one, but the opposite is true for BnB against Var-8 and Var-2. Our adapted attacks are remarkably stable when training for lower bandwidth scales, while BnB's accuracy decreases substantially except in the case of Var-4. Finally, an attacker that can collect a network trace is also expected to be able to select a nearby augmentation option, suggesting that the values reported along the diagonal (and summarized in Table 3) are, in most cases, the best predictor of what an attacker would see in practice.

We note that, due to the augmentation algorithm's reliance on randomly selected parameters, any given trace is unlikely to result in the same augmented trace when used multiple times. This enables increasing the size of the training dataset without monitoring additional network traces. Figure 9 shows the results of applying the augmentation to each trace multiple times to increase the size of the dataset. While we observe diminishing returns, these results show that up to a 10-15% further improvement in accuracy can be realized by generating 20 augmented versions of each trace.

Furthermore, we highlight that it is unlikely that an augmented trace will contain *precisely* the same quantity and timing of quality switches as any evaluated trace, but our results suggest that the mere presence of sufficient quality switches in varying positions is beneficial to the models. Similarly, since we observe promising results when training for lower bandwidth scales than the targeted dataset vs. higher ones, it appears most important to have a certain minimum number of quality switches in the augmented traces, but too many may hide relationships between nearby segments.

7.3 Offset Augmentation

We next consider the accuracy decrease an attacker may suffer when using differently offset network traces. We implement an augmentation strategy for this scenario by duplicating network traces and applying various offsets to them, iterating through each trace and modifying packet timestamps. To enable the application of offsets in our setting, only 40 seconds of the last 60 seconds are used by the attacker. This allows us to evaluate the accuracy of the attacks on offsets as big as 10 seconds in both directions, a reasonable upper limit for a low-latency live stream.

We test four models that are based on the same approach. The first model ($K = 1$) performs no duplications or offset applications. The second model ($K = 2$) duplicates all traces once and applies a 10/3 second offset to the two resulting traces, once in either direction. That is, one trace's packet timestamps will be incremented by 10/3 seconds and the other's will be decremented by 10/3 seconds. The third model ($K = 3$) duplicates the original traces three times: the four resulting traces will have 2- and 6-second offsets in both directions. The fourth and final model ($K = 4$) duplicates all traces seven times and applies 70/9, 50/9, 30/9- and 10/9-second offsets in both directions. These values are chosen such that 2^{K-1} offsets evenly split the available 20-second interval, from −10 to 10 seconds, into $2^{K-1} + 1$ partitions.

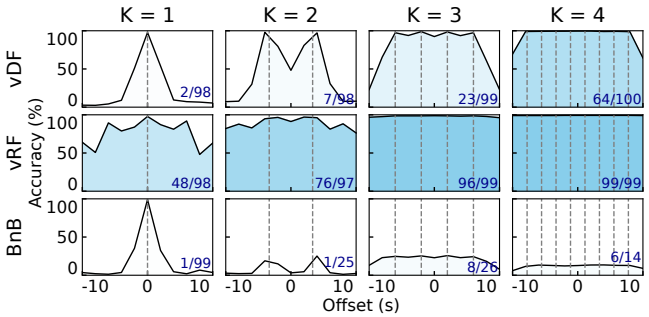


Figure 10: Accuracy of offset augmentation when varying the evaluation offset.

7.4 Offset Augmentation Evaluation

To test the efficacy of the different models, we generate offset datasets by duplicating the constant bandwidth dataset 10 times and applying a different offset in the range of −10 to 10 seconds, with increments of 2 seconds, to each version. Figure 10 shows the results of training after applying augmentation to the constant bandwidth dataset and evaluating the attacks on the constant bandwidth dataset (no offset) and all offset datasets.

Across all attacks and offset augmentation models, there is a distinct peak in accuracy at offsets close to those included in the training data (marked with dashed lines in Figure 10). vDF achieves very high accuracy at the peaks, with larger dips between them when $K = 1$ and 2; however, these fill in as more offsets are included in the training data, and the attack reaches almost perfect accuracy except with maximal offsets. Interestingly, vRF's baseline accuracy does not decrease uniformly as offset increases: due to its inherent resilience to offsets (demonstrated in Section 6.4), it performs even better than vDF, achieving at least 96% accuracy at all offsets when $K = 3$ and 99% accuracy when $K = 4$. This demonstrates the benefit of including traces with varying offsets in the training data.

BnB, on the other hand, does not respond well to offsets, even with augmentation: while it also achieves peak accuracy at offsets close to those included in the training data, maximum accuracy decreases as K increases. Thus, incrementing K has the effect of stabilizing the accuracy across different offsets while reducing it to very low values. This may result from the combination of the attack's low-granularity input format and sensitivity (as shown through our bandwidth augmentation experiments); in any case, we conclude that our adapted attacks are most robust against differences in live latency, particularly when utilizing our offset augmentation.

7.5 Combining Variable Bandwidth Augmentation and Offset Augmentation

Since both evaluated challenges might be encountered at the same time in the real world, we consider combining our proposed augmentation techniques. To evaluate the combination, we apply offset augmentation in the same way described previously to the resulting traces of the variable bandwidth augmentation. Since vRF benefits the most from offset augmentation, we only evaluate this attack using both augmentation strategies. Similarly, we only use the most effective version of the offset augmentation, $K = 4$. Figure 11 presents the results of training on doubly augmented data and evaluating on different offsets applied to each bandwidth scale.

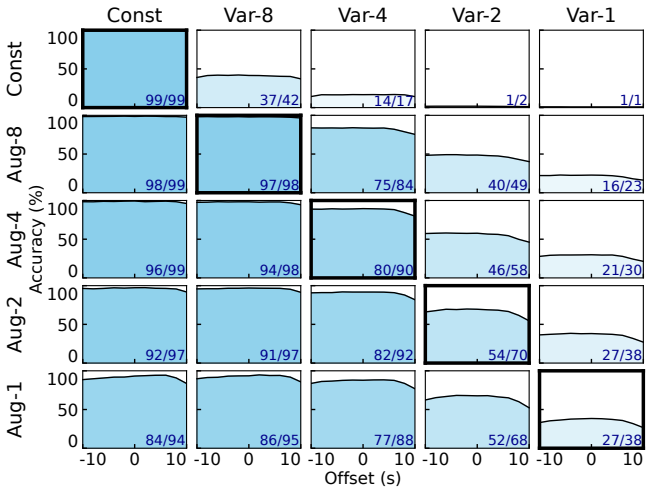


Figure 11: Accuracy of vRF with both offset and bandwidth augmentation when varying the evaluation offset and bandwidth conditions. Here, training datasets are along the y-axis and evaluation datasets on the x-axis.

As in our offset augmentation experiments, vRF achieves similar accuracy across all offsets. Accuracy is rather high in most cases, and combining both augmentation strategies appears to afford the attack additional benefits related to C2: in most cases, even the *minimum* accuracy is slightly higher than the results reported in Table 4 using only bandwidth augmentation. We attribute this to vRF's partial resilience to offsets without augmentation, which likely allows it to better leverage training traces with both the same and similar offsets as the trace being classified. Though accuracy is, as before, low when training for higher bandwidth scales than the victim's, vRF performs remarkably well, even with limited epochs, when training for identical or worse conditions.

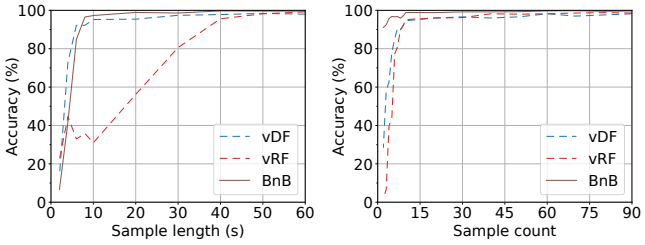
8 Impact of Video-Related Data Limitations

Finally, we evaluate and discuss the effects of the video-related data limitations outlined in Section 3.2.

8.1 Limited Observation Periods (L1)

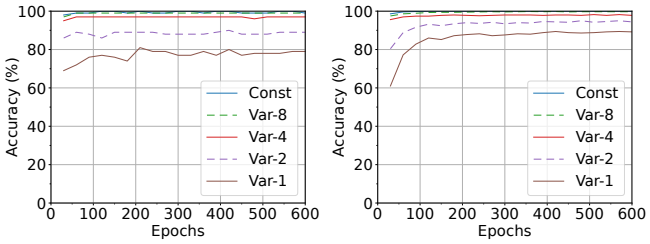
Since the length of a video stream determines the amount of data the attacker can use to characterize the corresponding network trace, it is of interest to study and quantify the impact of limiting the length of network traces. To do this, we iteratively reduce the constant bandwidth dataset to include decreasingly long samples – from 60 to 2 seconds – and measure the accuracy of our adapted attacks and BnB for each sample length.

Since trace length is known to the attacker prior to training and evaluation (i.e., the attacker determines how long to monitor connections for), we opt to tailor the input matrix for the adapted attacks to each considered length. We do this by maintaining the same matrix dimensions but changing the sizes of the buckets. In vRF, for example, the new buckets are $60/400 = 0.15$ seconds long when targeting 60-second traces and $30/400 = 0.075$ seconds against 30-second traces. We take the same approach for all sample lengths with both vDF and vRF.



(a) Sample length (b) Samples used in training

Figure 12: Accuracy when introducing data limitations.



(a) vDF (b) vRF

Figure 13: Accuracy when varying the number of epochs.

Figure 12(a) shows the results. It is clear that vRF is outperformed by both vDF and BnB when sample length decreases, particularly for sample lengths shorter than 40 seconds, where accuracy begins to deteriorate significantly; this is likely because we tune the buckets, and vRF does not perform as well with high-granularity inputs, as shown in Section 5.1. Both vDF and BnB maintain high accuracy down to sample lengths of 20 seconds, with vDF performing the best for even shorter samples. Regardless, longer observation periods provide clear benefits with all attacks. This is in contrast to WF and some other traffic analysis domains, where the majority of useful features can be captured within a short time window.

8.2 Number of Training Samples (L2)

Gathering more traces for training represents another straightforward way for an attacker to increase the amount of training data. We thus evaluate the effect the number of samples has on attack accuracy. The constant and variable bandwidth datasets are of size $100 \times 10 \times 10$ – as with trace length, we evaluate the effect of less training data by iteratively decreasing the dataset size. We test from $100 \times 10 \times 10$ down to $100 \times 10 \times 1$, with decrements of 1, and we further decrease the dataset size to $100 \times 3 \times 1$ to more conclusively capture the effect of dataset limitations.

Figure 12(b) shows the results. A significant reduction of the sample count is evidently required to have a mentionable impact on the accuracy of any of the attacks, but when the sample count becomes extremely small, the accuracy of all attacks except for BnB quickly deteriorates. BnB displays an impressive resilience to this data limitation scenario, due in large part to the high number of epochs selected, which allows the model to utilize each sample more times in training and thus be more accurate when samples are limited. As a rule, though, better accuracy can be achieved with all attacks by gathering more training samples.

Table 5: Accuracy (%) of training and evaluating on the same dataset with 200 epochs. We include the absolute accuracy improvements (in green) over the baseline results reported in Table 1 for 30 epochs, and the last two columns summarize the "winning" model and its increases in accuracy over BnB.

	vDF(200)		vRF(200)		Winner	vs. BnB
Const	99.7	Δ1.5	100.0	Δ0.7	vRF(200)	Δ0.1
Var-8	99.4	Δ2.8	99.7	Δ1.9	vRF(200)	Δ2.1
Var-4	97.1	Δ2.2	97.9	Δ2.7	vRF(200)	Δ5.4
Var-2	91.2	Δ5.3	93.7	Δ13.9	vRF(200)	Δ20.0
Var-1	79.1	Δ9.3	86.0	Δ27.5	vRF(200)	Δ34.4

8.3 Impact of More Epochs (L3)

Thus far, we have elected not to tune the epochs of our adapted attacks to enable better comparisons with their default versions. Compared to BnB, an attack aimed at characterizing video traces, this value is very low. This raises the question of what impact increasing the epoch count would have on our adapted attacks as well as how severe the problem of overfitting is in video fingerprinting as opposed to WF, since the risk of overfitting is the reason for the low epoch value used by the website attacks [60, 62].

Figures 13(a) and 13(b) visualize attack accuracy as the number of epochs increases for vDF and vRF, respectively. Regardless of network conditions, the accuracy of both attacks increases with the number of epochs. We also see no signs of overfitting, as attack accuracy does not decrease even with a very large epoch count. To verify this, we also evaluate epoch values of 5 000 and 10 000, finding no noteworthy difference in accuracy compared to 600 epochs. This shows that overfitting is not a problem for vDF or vRF with our closed-world datasets, regardless of network conditions.

The Case for vRF with 200 Epochs: We find that using more epochs is generally beneficial to our adapted attacks, including when faced with challenges C2 and C3. To highlight the strengths of the attacks, we next show example results using 200 epochs for both vDF and vRF. This value is motivated by the diminishing returns of using more epochs and the desire to limit runtime.

Table 5 presents the accuracies for vDF and vRF with 200 epochs when training and evaluating under the same conditions (C1). Here, we include comparisons (green delta values) with the case of 30 epochs. As shown, accuracy improves across the board, with the most noteworthy difference being the massive increases for vRF in poor conditions: in all cases, our adapted attacks (represented by vRF) improve substantially over the previous state-of-the-art BnB.

While vRF with 200 epochs is not immune to performance degradation when having to train on different network conditions than those of the victim (C2), we find that it is more robust than the other attacks. To see this, we refer to Table 6, which is presented in the same format as Table 2 for direct comparison.

We also see that our adapted attacks, especially vRF, perform very well with 200 epochs in combination with our two augmentation techniques, addressing challenges C2 and C3, respectively. This is illustrated by the big improvements reported in Table 7 and the overall high accuracy seen in Figure 14. Here, it should again be noted that vRF is the only attack able to excel in the presence of offsets. These results make a clear case for using vRF for video fingerprinting under both good and challenging conditions.

Table 6: Accuracy (%) when training and evaluating under different bandwidth conditions when using 200 epochs for our adapted attacks. These results provide a direct comparison with those reported in Table 2.

(a) **vDF(200)**

Training data ↓	Testing dataset				
	Const	Var-8	Var-4	Var-2	Var-1
Const	99.7	14.8	4.0	1.1	1.4
Var-8	99.5	99.4	85.8	40.7	13.8
Var-4	75.1	97.2	97.1	70.4	37.6
Var-2	29.7	94.3	94.9	91.2	59.3
Var-1	4.3	70.9	85.9	82.0	79.1

(b) **vRF(200)**

Testing dataset				
Const	Var-8	Var-4	Var-2	Var-1
100.0	80.4	40.5	13.9	1.0
99.7	99.7	91.7	44.8	19.8
99.0	99.1	97.9	71.2	35.9
89.2	96.8	97.1	93.7	66.0
6.8	66.5	86.9	87.9	86.0

Table 7: Accuracy (%) of bandwidth augmentation and improvements over training on the constant bandwidth dataset when using 200 epochs for our adapted attacks. These results provide a direct comparison with those reported in Table 3.

Attack	Training dataset	Testing dataset			
		Var-8	Var-4	Var-2	Var-1
vDF(200)	**Const**	14.8	4.0	1.1	1.4
	Aug-X	87.6	81.2	65.5	38.5
		+491.9%	+1 930.0%	+5 854.5%	+2 650.0%
vRF(200)	**Const**	80.4	40.5	13.9	1.0
	Aug-X	97.5	91.2	69.7	37.6
		+21.3%	+125.2%	+401.4%	+3 660.0%

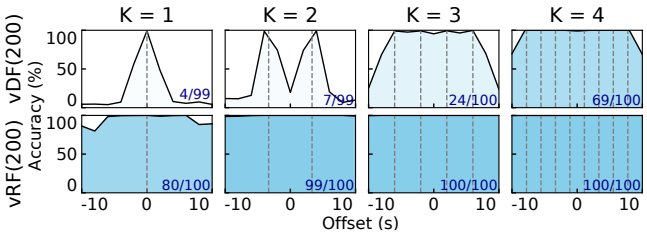


Figure 14: Accuracy of offset augmentation with varying evaluation offset and 200 epochs for our adapted attacks. These results provide a direct comparison with Figure 10.

9 Conclusions

Despite the clear threat of video fingerprinting, understanding of the effectiveness of these attacks in real-world scenarios has been limited, as most research assumes ideal attack conditions. We have addressed this gap by focusing on the unique challenges posed by streaming traffic, particularly its dependence on client and network states. Our work represents the first comprehensive study of video fingerprinting attacks under challenging conditions.

Novel Attacks: By adapting two state-of-the-art website fingerprinting attacks to video traffic and developing custom input formats, we achieve notable improvements over existing video fingerprinting techniques. The attacks are shown to benefit from more training, with the best attack (vRF) surpassing the state-of-the-art video fingerprinting attack (BnB) in almost all cases and establishing a new benchmark for CNN-based video fingerprinting attacks.

Challenging Conditions and Impacts on Attack Accuracy: By evaluating different deep learning model architectures against video data collected under various network conditions, we quantify the impact of adaptive bitrate streaming and live latency variations on attack accuracy. Our study provides valuable insights into real-world limitations that an attacker is likely to encounter and highlights the need to evaluate attack performance under more challenging conditions than prior research has.

Augmentation-Based Attack Extensions: To enhance attack performance under challenging conditions, we introduce two augmentation approaches that significantly boost accuracy in variable network environments and when the adversary lacks knowledge of the victim's live latency. These augmentations, applicable both separately and in combination, demonstrate remarkable performance gains. In addition to addressing our considered challenges, the randomness introduced in our bandwidth augmentation algorithm mitigates the impact of limited training samples.

Impact of Data Limitations: Our analysis also highlights the critical importance of data limitations, such as observation time, dataset size, and training time. We show that high attack accuracy can be achieved with limited samples over short durations and that increasing the number of training epochs significantly enhances the performance of our video-adapted attacks.

Selecting a Winner: Finally, while vRF outperforms the other attacks in all of the primary scenarios we consider, we observe some exceptions when network conditions are good but there are few or short training samples: under such conditions (1) BnB is most robust to cases with very few samples and (2) vDF is most robust to short video samples. However, in all other cases, vRF is the clear winner, as it outperform the other attacks when network conditions are variable but known (C1) and in unknown bandwidth conditions with and without augmentation (C2); also, it is much more robust to offsets (C3) both with and without augmentation. Moreover, it is clear that our augmentation techniques combined with our video-adapted attacks, particularly vRF, always provide significant accuracy benefits when faced with challenging conditions. These considerable advancements over the prior state-of-the-art highlight the significant privacy threat posed by video fingerprinting attacks and stress the pressing need for effective defense techniques.

Code and Datasets: Our attack code and extended dataset can be found at https://github.com/trafnex/video-augmentation.

Acknowledgments

We thank Matthias Beckerle and Tobias Pulls for discussions that contributed significantly to the work presented in this paper. We are also thankful to our shepherd Thorsten Strufe and the anonymous reviewers for their feedback. This work was partially supported by the Swedish Foundation for Strategic Research (SSF) and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[1] Waleed Afandi, Syed Muhammad Ammar Hassan Bukhari, Muhammad U. S. Khan, Tahir Maqsood, and Samee U. Khan. 2022. Fingerprinting Technique for YouTube Videos Identification in Network Traffic. *IEEE Access* 10 (2022).

[2] Dilawer Ahmed, Aafaq Sabir, and Anupam Das. 2023. Spying through Your Voice Assistants: Realistic Voice Command Fingerprinting. In *Proc. USENIX Security.*

[3] J.S. Atkinson, M. Rio, J.E. Mitchell, and G. Matich. 2014. Your WiFi Is Leaking: Ignoring Encryption, Using Histograms to Remotely Detect Skype Traffic. In *Proc. IEEE Military Communications Conference (MILCOM).*

[4] Sangwook Bae, Mincheol Son, Dongkwan Kim, CheolJun Park, Jiho Lee, Sooel Son, and Yongdae Kim. 2022. Watching the Watchers: Practical Video Identification Attack in LTE Networks. In *Proc. USENIX Security.*

[5] Alireza Bahramali, Ardavan Bozorgi, and Amir Houmansadr. 2023. Realistic Website Fingerprinting By Augmenting Network Traces. In *Proc. ACM Computer and Communications Security (CCS).*

[6] Alireza Bahramali, Ramin Soltani, Amir Houmansadr, Dennis Goeckel, and Don Towsley. 2020. Practical traffic analysis attacks on secure messaging applications. In *Proc. Network and Distributed System Security (NDSS).*

[7] Sanjit Bhat, David Lu, Albert Kwon, and Srinivas Devadas. 2018. Var-CNN: A data-efficient website fingerprinting attack based on deep learning. In *Proc. Privacy Enhancing Technologies (PETS).*

[8] Dario Bonfiglio, Marco Mellia, Michela Meo, Dario Rossi, and Paolo Tofanelli. 2007. Revealing Skype traffic: when randomness plays with you. *SIGCOMM CCR* (2007).

[9] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. 2014. A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses. In *Proc. ACM Computer and Communications Security (CCS).*

[10] Giovanni Cherubin, Rob Jansen, and Carmela Troncoso. 2022. Online website fingerprinting: Evaluating website fingerprinting attacks on tor in the real world. In *Proc. USENIX Security.*

[11] Thilini Dahanayaka, Guillaume Jourjon, and Suranga Seneviratne. 2022. Dissecting traffic fingerprinting CNNs with filter activations. *Computer Networks* (2022).

[12] Shuaifu Dai, Alok Tongaonkar, Xiaoyin Wang, Antonio Nucci, and Dawn Song. 2013. NetworkProfiler: Towards automatic fingerprinting of Android apps. In *Proc. IEEE INFOCOM.*

[13] Wladimir De la Cadena, Asya Mitseva, Jens Hiller, Jan Pennekamp, Sebastian Reuter, Julian Filter, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. 2020. TrafficSliver: Fighting Website Fingerprinting Attacks with Traffic Splitting. In *Proc. ACM CCS.*

[14] Xinhao Deng, Qilei Yin, Zhuotao Liu, Xiyuan Zhao, Qi Li, Mingwei Xu, Ke Xu, and Jianping Wu. 2023. Robust multi-tab website fingerprinting attacks in the wild. In *Proc. IEEE Security and Privacy (S&P).*

[15] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. Tor: The Second-Generation Onion Router. In *Proc. USENIX Security.*

[16] Meijie Du, Minchao Xu, Kedong Liu, Weitao Tang, Lijuan Zheng, and Qingyun Liu. 2023. Long-Short Terms Frequency: A Method for Encrypted Video Streaming Identification. In *Proc. Computer Supported Cooperative Work in Design (CSCWD).*

[17] Ran Dubin, Amit Dvir, Ofir Pele, and Ofer Hadar. 2017. I Know What You Saw Last Minute—Encrypted HTTP Adaptive Video Streaming Title Classification. *IEEE Trans. on Information Forensics and Security (TIFS)* (2017).

[18] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. 2012. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *Proc. IEEE Security and Privacy (S&P).*

[19] Yanjie Fu, Hui Xiong, Xinjiang Lu, Jin Yang, and Can Chen. 2016. Service usage classification with encrypted internet traffic in mobile messaging apps. *IEEE Trans. on Mobile Computing* (2016).

[20] Jiajun Gong and Tao Wang. 2020. Zero-delay Lightweight Defenses against Website Fingerprinting. In *Proc. USENIX Security.*

[21] Jiajun Gong, Wuqi Zhang, Charles Zhang, and Tao Wang. 2022. Surakav: Generating Realistic Traces for a Strong Website Fingerprinting Defense. In *Proc. IEEE Security and Privacy (S&P).*

[22] Jiajun Gong, Wuqi Zhang, Charles Zhang, and Tao Wang. 2023. WFDefProxy: Real World Implementation and Evaluation of Website Fingerprinting Defenses. *IEEE Trans. on Information Forensics and Security (TIFS)* (2023).

[23] Jiaxi Gu, Jiliang Wang, Zhiwen Yu, and Kele Shen. 2018. Walls Have Ears: Traffic-based Side-channel Attack in Video Streaming. In *Proc. IEEE INFOCOM.*

[24] David Hasselquist, Martin Lindblom, and Niklas Carlsson. 2022. Lightweight fingerprint attack and encrypted traffic analysis on news articles. In *Proc. IFIP Networking.*

[25] David Hasselquist, Christian Vestlund, Niklas Johansson, and Niklas Carlsson. 2022. Twitch Chat Fingerprinting. In *Proc. IFIP Network Traffic Measurement and Analysis Conference (TMA).*

[26] David Hasselquist, Ethan Witwer, August Carlson, Niklas Johansson, and Niklas Carlsson. 2024. Raising the Bar: Improved Fingerprinting Attacks and Defenses for Video Streaming Traffic. In *Proc. Privacy Enhancing Technologies (PETS).*

[27] Jamie Hayes and George Danezis. 2016. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *Proc. USENIX Security.*

[28] Sébastien Henri, Gines Garcia-Aviles, Pablo Serrano, Albert Banchs, and Patrick Thiran. 2020. Protecting against Website Fingerprinting with Multihoming. In *Proc. Privacy Enhancing Technologies (PETS).*

[29] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. 2009. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier. In *Proc. ACM Workshop on Cloud Computing Security.*

[30] Andrew Hintz. 2002. Fingerprinting websites using traffic analysis. In *Proc. Workshop on Privacy Enhancing Technologies (PETS).*

[31] James K Holland and Nicholas Hopper. 2022. RegulaTor: A Straightforward Website Fingerprinting Defense. In *Proc. Privacy Enhancing Technologies (PETS).*

[32] Rob Jansen, Ryan Wails, and Aaron Johnson. 2024. A Measurement of Genuine Tor Traces for Realistic Website Fingerprinting. *arXiv:2404.07892* (2024).

[33] Zhaoxin Jin, Tianbo Lu, Shuang Luo, and Jiaze Shang. 2023. Transformer-based Model for Multi-tab Website Fingerprinting Attack. In *Proc. ACM Computer and Communications Security (CCS).*

[34] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. 2014. A critical evaluation of website fingerprinting attacks. In *Proc. ACM Computer and Communications Security (CCS).*

[35] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. 2016. Toward an efficient website fingerprinting defense. In *Proc. European Symposium on Research in Computer Security (ESORICS).*

[36] Antonis Kalogeropoulos, Federica Cherubini, and Nic Newman. 2016. The Future of Online News Video. *Digital News Project* (2016).

[37] Vengatanathan Krishnamoorthi, Niklas Carlsson, Emir Halepovic, and Eric Petajan. 2017. BUFFEST: Predicting Buffer Conditions and Real-time Requirements of HTTP(S) Adaptive Streaming Clients. In *Proc. ACM Multimedia Systems Conference (MMSys).*

[38] Feng Li, Jae Won Chung, and Mark Claypool. 2018. Silhouette: Identifying YouTube Video Flows from Encrypted Traffic. In *Proc. ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV).*

[39] Jianfeng Li, Shuohan Wu, Hao Zhou, Xiapu Luo, Ting Wang, Yangyang Liu, and Xiaobo Ma. 2022. Packet-Level Open-World App Fingerprinting on Wireless Traffic. In *Proc. Network and Distributed System Security (NDSS).*

[40] Ying Li, Yi Huang, Richard Xu, Suranga Seneviratne, Kanchana Thilakarathna, Adriel Cheng, Darren Webb, and Guillaume Jourjon. 2018. Deep Content: Unveiling Video Streaming Content from Encrypted WiFi Traffic. In *Proc. IEEE Network Computing and Applications (NCA).*

[41] Nate Mathews, James K Holland, Se Eun Oh, Mohammad Saidur Rahman, Nicholas Hopper, and Matthew Wright. 2023. SoK: A critical evaluation of efficient website fingerprinting defenses. In *Proc. IEEE Security and Privacy (S&P).*

[42] Asya Mitseva and Andriy Panchenko. 2024. Stop, Don't Click Here Anymore: Boosting Website Fingerprinting By Considering Sets of Subpages. In *USENIX Security.*

[43] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. 2021. Defeating DNN-Based Traffic Analysis Systems in Real-Time With Blind Adversarial Perturbations. In *Proc. USENIX Security.*

[44] Nic Newman, Richard Fletcher, Kirsten Eddy, Craig T. Robertson, and Rasmus Kleis Nielsen. 2023. Digital News Report. *Reuters Institute for the study of Journalism* (2023).

[45] Rishab Nithyanand, Xiang Cai, and Rob Johnson. 2014. Glove: A Bespoke Website Fingerprinting Defense. In *Proc. ACM Workshop on Privacy in the Electronic Society (WPES).*

[46] Se Eun Oh, Nate Mathews, Mohammad Saidur Rahman, Matthew Wright, and Nicholas Hopper. 2021. GANDaLF: GAN for data-limited fingerprinting. In *Proc. Privacy Enhancing Technologies (PETS).*

[47] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. 2016. Website Fingerprinting at Internet Scale. In *Proc. Network and Distributed System Security (NDSS).*

[48] Mike Perry and George Kadianakis. 2020. Circuit Padding Developer Documentation. https://github.com/torproject/tor/blob/main/doc/HACKING/CircuitPaddingDevelopment.md.

[49] Pew Research Center. 2023. Social Media and News Fact Sheet. https://www.pewresearch.org/journalism/fact-sheet/social-media-and-news-fact-sheet/.

[50] Tobias Pulls. 2020. Towards Effective and Efficient Padding Machines for Tor. *arXiv:2011.13471* (2020).

[51] Tobias Pulls and Ethan Witwer. 2023. Maybenot: A Framework for Traffic Analysis Defenses. In *Proc. ACM Workshop on Privacy in the Electronic Society (WPES).*

[52] Darijo Raca, Jason J Quinlan, Ahmed H Zahran, and Cormac J Sreenan. 2018. Beyond throughput: A 4G LTE dataset with channel and context metrics. In *Proc. ACM Multimedia Systems Conference (MMSys).*

[53] Mohammad Saidur Rahman, Mohsen Imani, Nate Mathews, and Matthew Wright. 2021. Mockingbird: Defending against deep-learning-based website fingerprinting attacks with adversarial traces. *IEEE Trans. on Information Forensics and Security (TIFS)* (2021).

[54] Mohammad Saidur Rahman, Payap Sirinam, Nate Mathews, Kantha Girish Gangadhara, and Matthew Wright. 2020. Tik-Tok: The Utility of Packet Timing in

Website Fingerprinting Attacks. In *Proc. Privacy Enhancing Technologies (PETS)*.
[55] Andrew Reed and Benjamin Klimkowski. 2016. Leaky streams: Identifying variable bitrate DASH videos streamed over encrypted 802.11n connections. In *Proc. IEEE Consumer Communications & Networking Conference (CCNC)*.
[56] Andrew Reed and Michael Kranch. 2017. Identifying HTTPS-Protected Netflix Videos in Real-Time. In *Proc. ACM Conference on Data and Application Security and Privacy (CODASPY)*.
[57] Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom Van Goethem, and Wouter Joosen. 2018. Automated Website Fingerprinting through Deep Learning. In *Proc. Network and Distributed System Security (NDSS)*.
[58] Sandvine. 2024. 2024 Global Internet Phenomena Report. https://www.sandvine.com/global-internet-phenomena-report-2024.
[59] Roei Schuster, Vitaly Shmatikov, and Eran Tromer. 2017. Beauty and the Burst: Remote Identification of Encrypted Video Streams. In *Proc. USENIX Security*.
[60] Meng Shen, Kexin Ji, Zhenbo Gao, Qi Li, Liehuang Zhu, and Ke Xu. 2023. Subverting Website Fingerprinting Defenses with Robust Traffic Representation. In *Proc. USENIX Security*.
[61] Meng Shen, Yiting Liu, Liehuang Zhu, Xiaojiang Du, and Jiankun Hu. 2020. Fine-grained webpage fingerprinting using only packet length information of encrypted traffic. *IEEE Trans. on Information Forensics and Security (TIFS)* (2020).
[62] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. 2018. Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning. In *Proc. ACM Computer and Communications Security (CCS)*.
[63] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew Wright. 2019. Triplet Fingerprinting: More Practical and Portable Website Fingerprinting with N-shot Learning. In *Proc. ACM Computer and Communications Security (CCS)*.
[64] Iraj Sodagar. 2011. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE MultiMedia* (2011).
[65] Vincent F. Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. 2018. Robust Smartphone App Identification via Encrypted Network Traffic Analysis. *IEEE Trans. on Information Forensics and Security (TIFS)* (2018).
[66] Alexander Vaskevich, Thilini Dahanayaka, Guillaume Jourjon, and Suranga Seneviratne. 2021. Smaug: Streaming media augmentation using CGANs as a defence against video fingerprinting. In *Proc. IEEE Network Computing and Applications (NCA)*.
[67] Tim Walsh, Trevor Thomas, and Armon Barton. 2024. Exploring the Capabilities and Limitations of Video Stream Fingerprinting. In *Proc. IEEE Security and Privacy Workshops (SPW)*.
[68] Chenggang Wang, Jimmy Dani, Xiang Li, Xiaodong Jia, and Boyang Wang. 2021. Adaptive Fingerprinting: Website Fingerprinting over Few Encrypted Traffic. In *Proc. ACM Conference on Data and Application Security and Privacy (CODASPY)*.
[69] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. 2014. Effective Attacks and Provable Defenses for Website Fingerprinting. In *Proc. USENIX Security*.
[70] Tao Wang and Ian Goldberg. 2017. Walkie-Talkie: An Efficient Defense Against Passive Website Fingerprinting Attacks. In *Proc. USENIX Security*.
[71] Xinyuan Wang, Shiping Chen, and Sushil Jajodia. 2005. Tracking anonymous peer-to-peer VoIP calls on the internet. In *Proc. ACM Computer and Communications Security (CCS)*.
[72] Ethan Witwer, James K Holland, and Nicholas Hopper. 2022. Padding-only defenses add delay in Tor. In *Proc. ACM Workshop on Privacy in the Electronic Society (WPES)*.
[73] Hua Wu, Zhenhua Yu, Guang Cheng, and Shuyi Guo. 2020. Identification of encrypted video streaming based on differential fingerprints. In *Proc. IEEE Computer Communications Workshops (INFOCOM WKSHPS)*.
[74] Luming Yang, Shaojing Fu, Yuchuan Luo, and Jiangyong Shi. 2020. Markov probability fingerprints: A method for identifying encrypted video traffic. In *Proc. Mobility, Sensing and Networking (MSN)*.
[75] Xiaokuan Zhang, Jihun Hamm, Michael K Reiter, and Yinqian Zhang. 2019. Statistical privacy for streaming traffic. In *Proc. Network and Distributed System Security (NDSS)*.
[76] Xiyuan Zhang, Gang Xiong, Zhen Li, Chen Yang, Xinjie Lin, Gaopeng Gou, and Binxing Fang. 2024. Traffic spills the beans: A robust video identification attack against YouTube. *Computers & Security* (2024).