CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

# Watch out for the mole: Can the obfuscation defences put in place by DAITA be circumvented for video fingerprinting?

Master's thesis in Computer science and engineering

Luna Ridderland, Riya Tagra

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Master's thesis 2026

# Watch out for the mole: Can the obfuscation defences put in place by DAITA be circumvented for video fingerprinting?

Luna Ridderland, Riya Tagra

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Watch out for the mole: Can the obfuscation defences put in place by DAITA be circumvented for video fingerprinting?
Luna Ridderland, Riya Tagra



Supervisor: Romaric Duvignau, Department of Computer Science and Engineering
Advisor: Odd Stranne, Mullvad VPN
Examiner: Romaric Duvignau, Department of Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Watch out for the mole: Can the obfuscation defences put in place by DAITA be circumvented for video fingerprinting?
Luna Ridderland, Riya Tagra
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

# Abstract

In an ever increasingly digital world where user data has become a commodity to be analysed and sold, digital privacy is growing evermore important. Despite the widespread adoption of privacy solutions such as HTTPS and VPNs, information about network traffic is still leaked despite encryption. One such case being the adaptive bitrate protocols used for video streaming, such as MPEG-DASH. In the last years the severity of this vulnerability has become apparent. In a recent study by Björklund and Duvignau, it has proved feasible to use this leak to build an attack on a large scale, covering hundreds of thousands of videos across multiple video streaming providers, capable of identifying streamed video traffic with high accuracy. While there are certain tools that are used to mitigate these types of traffic analysis attacks, one such tool being Mullvad VPN's DAITA, it has not previously been known to what extent it protects against analysis of video traffic. This thesis investigates to what extent DAITA protects against the attack, showing that during testing a protection of 78% was observed. This work has also looked at the underlying traffic patterns of video streaming using DAITA, and found that the degree of protection is likely dependent on DAITA's internal system parameters and on what content is being streamed. Alternative identification logic is tested in an attempt to circumvent the added noise introduced by DAITA. Additionally the feasibility of reducing the size of the required dataset was investigated, which if possible would show that an attack utilizing this vulnerability would be cheaper to deploy on a large scale, making further defensive advancements an important matter.

Keywords: MPEG-DASH, Adaptive Bitrate Streaming, DAITA, Video Fingerprinting, Video Streaming Attacks.

# Acknowledgements

We would like to give our gratitude to our academic supervisor from Chalmers, Romaric Duvignau for his constant support and encouragement throughout the duration of this thesis. We would also like to thank Odd Stranne from Mullvad VPN for his expertise and assistance with DAITA.

Luna Ridderland & Riya Tagra, Gothenburg, 2026-09-17

# Contents

# List of Figures

# List of Tables

# 1

# Introduction

The right to browse, stream and express thoughts freely in the digital sphere is becoming more and more important, as user data and behaviour has become a commodity to be analysed and sold. A large part of the fight for digital sovereignty, is secure and free network communication. The most dominant type of traffic within communication networks is video streaming [1]. On average, 1 hour and 22 minutes a day are spent per person streaming videos online and the video streaming market has approximately 1.4 billion users as of 2024 [2]. Given the prevalence of video streaming, it is of high value for data collecting and analysis services to know what content is streamed and by whom. Applications of such data profiling could for example be corporations seeing users stream certain types of videos and therefore tailoring ads towards those interests, or authoritarian regimes wanting to monitor the population of their country. Such actors have an interest in investing in network and communication analysis. Therefore, protecting and keeping such data anonymous and obfuscated is a continuous process of research and development.

The past few decades have seen an exponential increase in cyber defences to protect communication and networking. Solutions regarding end-to-end encryption have increased with the share of web traffic using HTTPS, as well as the growing adoption of solutions such as Tor and VPNs that make the end-user unlinkable from their IP addresses [3], [4]. While this makes it more difficult for attackers to target users and the content of the communication, it does not always protect against the analysis of other more obscure aspects of the communication. These aspects can include energy consumption, time of transmission, or patterns in the transmission of content such as the size of the packets and the change frequency of the packets. This kind of analysis is called side-channel analysis. [4], [5].

The current international standard protocol for streaming video, is Dynamic adaptive streaming over HTTPS (shortened to DASH), and also known as MPEG-DASH. It is a bitrate adaptive streaming protocol that allows for videos to be streamed over HTTPS and is fully encrypted, which offers good security against attacks such as network sniffing attacks [6]. The protocol is based around dynamically adapting the quality at which the video is transmitted, allowing for network quality changes, resulting in a smooth viewing experience. This dynamic streaming is done by breaking the video up into small segments and adapting the quality based on what the client requests. While this protocol has many benefits and easily adapts to different network conditions, research has shown that these segments are sent in unique pat-

terns, which, upon analysis, leak distinct images of the video being streamed. These images can then be used and cross-referenced against a database of pre-existing patterns, or *fingerprints*, to identify the video without ever looking at the content of the packets themselves [7].

One such exploit is presented by Björklund and Duvignau in their paper *Endangered Privacy: Large-Scale Monitoring of Video Streaming Services* (hereby denoted as *Endangered Privacy*) [5]. It shows a protocol-agnostic system that uses network traffic side-channel analysis to identify videos despite VPNs being implemented and without using network layer information. The attack is based on collecting a database of many such video fingerprints or packet burst patterns from three streaming platforms and uses a k-d tree search to identify videos being streamed. The attack results in a 99.5% accuracy rate. This system uses the largest dataset to date.

One privacy solution that addresses these kinds of side-channel exploits, is Mullvad VPN's DAITA, which is built using Maybenot, a framework for traffic analysis defences [8], [9], [10]. DAITA works by *normalizing* the traffic, that is, making sure each packet being sent is the same size and adding extra packets as padding between the "real" packets. In the case of DASH, this alters the *bursty* pattern.

Researching and pre-empting potential weaknesses before malicious actors do and/or are able to act on them, is a vital part of network security. It is of high importance to not only have accessible solutions to ensure privacy, but also to know their limitations. Part of this kind of pre-emptive security, is to think like a malicious attacker and find weaknesses and vulnerabilities in the system. Therefore, many studies and tests are conducted on new protocols and services to reveal possible weak spots and patch them before they are used maliciously.

## 1.1 Aim

This project aims to understand how Endangered Privacy [5] can be improved to take into account changes made to network traffic produced by DAITA, and allow for the classification of the traffic despite DAITA being used. By analysing network traffic produced by DAITA, the study looks at patterns emerging with different parameter combinations, to understand if any identifiable qualities of the video stream are preserved in the network stream, or if DAITA is able to obscure it completely. This paper not only aims to find weaknesses in DAITA, but also evaluate the large scale implications and viability of the attack. Is the attack only feasible in a lab environment, or is it realistic to implement in real-world scenarios? Parameters that will be taken into account in that respect are for example time, network fault tolerance, resources required etc. This study ultimately seeks to answer the following question:

*How can the attack Endangered Privacy be modified to take into account obfuscation implemented by Mullvad's protocol DAITA, and what is the large-scale applicability of the attack?*

## 1.2 Goals and Challenges

The first goal of the project is to modify the Endangered Privacy attack to limit the scope of its dataset to only include videos from SVT. In the state provided *Endangered Privacy*, the attack requires 192 GB of memory to run which is not feasible due to the hardware the group members have access to. Running the attack with a limited dataset, using videos exclusively from SVT only requires 9.3 GB, which is sufficient for the hardware we have access to. The main challenge here was to understand the source code and modify it successfully, although this is a good introduction to the source code which would be valuable in the later stages of the thesis. This did not cause any trouble and was an easily attainable goal.

The second goal of the project is to classify DAITA video streams into unique deterministic representations. This proved to be a difficult task, as the goal of DAITA is to obfuscate the network traffic, which not only makes the fingerprint less unique (as dummy packets are added methodically), but also randomizes it for every connection. The challenge here was to break down DAITA traffic and find where similarities lie between the same video streamed with different connections, and what makes it unique from other videos. While the methods and frameworks behind DAITA are publicly accessible, the exact algorithms used are closed source which makes it difficult to determine if there is any parameter in particular that leads to vulnerabilities.

The next goal of the project is to analyse and evaluate the applicability of the attack in real world scenarios. Evaluation metrics will include comparisons to Endangered Privacy regarding time, resources required, and the various environments they are successful in (real world vs lab simulated). The goal is to design an attack that is fast and light enough to be applied in the real world, as that is what will reveal the realistic threat that is posed to the public, and highlights the vulnerabilities in DAITA. The challenge that may appear in this development and evaluation is to keep the time and resource complexity at around the same level as Endangered Privacy. This is not expected to be too difficult, as the attack will be based on Endangered Privacy.

An additional goal if time permits would be to look at how DAITA can be further developed and expanded to make it more robust against the attack. In order to reach true development in protecting the user from attacks like this is to of course develop the security devices and services they use to be more robust and defend against a variety of attacks. Therefore in order to create true value, it would be beneficial to find how to defend against the attack.

## 1.3 Scope Limitations

In order to remain within the scope of the project, various limitations have been placed on the methodology and testable areas. The scope of the project will be limited to as follows. The content used to generate the video network traffic will be streamed from SVTplay, which is a Swedish public service streaming platform that is viewed by on average 3.06 million people a week [11]. Using a popular and

free streaming platform makes the results relevant for Swedish users, and makes data collection easily accessible. The success criteria of the project is to be able to simulate the attack successfully identifying the video being streamed while DAITA is active, given that the video exists previously in the database. As the attack that this thesis will be based on is very robust to traffic not in the database, a successfully identified video will indicate the possibility of a working attack, but not the large scale effectiveness. Hence, the focus of the project will lie in applying the attack, and modify it for use with DAITA traffic, rather than testing it on larger datasets. Therefore, the project will only look at a part of the SVT database from the Endangered Privacy source code (for example 1000 videos). As has been previously mentioned, an equally important part to the success of the attack, is the large-scale applicability and efficiency of the attack. Currently, endangered privacy takes about five to six minutes of streaming to successfully be able to apply the attack. It is outside the scope of this project to make the speed of endangered privacy or the attack created in this project much faster, rather the goal is that the attack should not be significantly slower.

## 1.4 Ethical Considerations

As more and more of the daily lives of our civilisation take place in the digital space, secure and private communication becomes increasingly vital. In order to defend that security and privacy, companies such as Mullvad offer services such as VPNs to keep users' communication safe. However as has been historically recorded, many malicious actors analyse and study security protocols that such services are built upon, in order to develop attacks that break them. In order to find weaknesses and vulnerabilities in their services in order to patch them and maintain their security, it is very important to conduct offensive cybersecurity analysis. With a deeper understanding of how a potential hacker may analyse and break into defences already set up, it makes it easier to improve the defences to a system proactively rather than reactively.

This thesis builds on a well-known vulnerability of MPEG-DASH that may be exploited by malicious actors, and seeks to examine to what extent DAITA protects against this. This means that this project is not introducing a new attack or a zero day vulnerability, rather further analysing how a known vulnerability in MPEG-DASH holds up against the DAITA protocol. This kind of work ultimately improves network security and gives users of Mullvad and DAITA more secure communication by analysing its vulnerabilities safely, rather than waiting until a malicious actor finds it and takes advantage of it.

A very important consideration in the kind of offensive cybersecurity analysis that this project aims to conduct, is to ensure full transparency and accountability with the service provider that is being analysed. As this project is conducted in coordination with Mullvad, there is full transparency in what will be analysed and attacked, and how that will be done. There will be no personal user data that will be involved, and the attack will be conducted in a safe lab environment on video stream traffic that has been specifically recorded for testing purposes. After the conclusion of the

project, all the results will be clearly communicated to Mullvad.

## 1.5 Thesis Outline

This report aims to present the relevance, methodology, and results found in the research done on trying to improve Endangered Privacy to be able to circumvent DAITA's obfuscation of network traffic. The report will, in the second chapter, give important technical background to have in order to engage with the research. It will explain how the attack Endangered Privacy works in detail and how it builds its database, breaks down the captured video traffic streams, and the algorithm it uses to identify the capture to a video in the database. Then a presentation of DAITA will be given, along with a discussion of the methodology used by DAITA to add padding packets to the network traffic. The framework used to build DAITA, called Maybenot, will also be explained.

The methodology used to conduct the research will be presented in the third chapter. How preparation will be done, the purpose of conducting tests before implementing the attack, as well as what attack development will look like, will be clarified. The environments used will be shown, as well as the evaluation metrics that will be used on the results.

The next chapter will show the result of implementing our attack on the traffic, as well as a comparison of if and how much it affects the efficacy of the attack on traffic streamed using DAITA. The penultimate chapter will present a discussion of the results, including their significance, as well as an evaluation of the applicability of the attack itself in realistic, non-simulated scenarios. We will also discuss further research that can be done.

The last chapter will conclude the report by summarizing the initial aim of the report, discussing the methodology and the results derived from it, and the conclusions that can be drawn from them. It will also discuss the relevance of the research, and how it aligns with the initial aim of the report.

# 2

# Background

This chapter aims to provide the technical background necessary for the research presented in this report. It will explain the basics of virtual private networking. It will also present how the standard protocol for video streaming works and show its benefits and flaws. It is also important to understand how video fingerprinting against DASH works, which will therefore be explained, as well as how the attack *Endangered Privacy* implements video fingerprinting. Mullvad's defence against traffic analysis attacks, DAITA, which adds obfuscation to network traffic, will also be explained.

## 2.1 Video Streaming Protocols

In order to stream a video, large amounts of data must be transmitted to the viewer, containing image data, sound data, and metadata of the video itself. When one opts to stream a video rather than download it, one does not want to save the entire video locally; rather, the goal is to view it at around the same rate that the video is being sent, so that one does not have to wait for the video to download or use storage space to store it. For these reasons, video streaming is a convenient and efficient method of viewing videos. This does, however, create a problem on how to stream a video in real-time as a viewer is watching it and keep the timing and quality good enough for it to be watchable without introducing stuttering.

One of the early protocols used for streaming videos is Real-Time Transport Protocol (RTP) [12][5]. It defined audio and video content packets and gave stream-session management to address issues regarding video streaming timing. While RTP worked well under managed networks with predictable network quality and throughput, it did not adapt well to modern large-scale networks. Issues included firewalls filtering RTP packets, a lack of support from content delivery networks (CDN), and RTP's requirement of clients needing a unique streaming session, which is resource intensive when scaled up to modern-day requirements. One of the biggest issues with RTP was also that it was unable to adapt to changing network conditions, as it has a fixed bitrate design and can only send videos at a predetermined bitrate [12][5].

As HTTP became a more established protocol for transmitting data over the internet, it came with benefits that combatted the issues seen in RTP. These include support for caches provided by CDNs to help minimize network traffic, as well as HTTP being firewall friendly. HTTP also allows streaming without managing a session

state for the client. These properties made HTTP useful for large-scale and content-heavy network transmission [12]. This made HTTP the most common underlying delivery method of modern video protocols such as Apple's HTTP Live Streaming (HLS), Microsoft's Smooth Streaming Protocol (MS-SSTR), and the aforementioned MPEG-DASH, (Dynamic Adaptive Streaming over HTTP) [12]. The latter is the most commonly used, and has been deemed the international standard protocol for video streaming. In order to build a protocol on HTTP, it must be able to handle changing network conditions and speeds; hence, all the mentioned protocols implement adaptive bitrate (ABR).

### 2.1.1 Adaptive Bitrate Protocols

ABR streaming protocols are based on adjusting the streaming quality of a video in real-time to handle changing network conditions in order to deliver the best quality video possible. It does this by dividing the streamed video into smaller fragments, called *segments*. The segments are usually around 2-15 seconds long and are encoded at multiple different bitrates. The client will then send requests for the segments, and the quality can vary between segments depending on the available bandwidth and its buffer consumption rate [5].

A simple example of adaptive bitrate streaming can be seen in figure 2.1. As presented, this multimedia content is made up of a video that is encoded at 3 different bitrates (5 Mbps, 2 Mbps, and 0.5 Mbps), as well as a trick mode, which is a low frame rate bitstream that is used while rewinding videos. The first audio that is included in the multimedia content is an English dub, encoded at three rates: one surround sound and two Advanced Audio Coding (AAC) at two rates (128 kbps and 48 kbps). The second audio is a French dub with only two forms of AAC encodings (128 kbps and 48 kbps).

As described by the labels and the arrow in figure 2.1, it is evident that this hypothetical client began streaming the video at a bitrate of 5 Mbps and in English with 128-kbps AAC (tag 1 in fig 2.1). During streaming the first segments, the client monitors the real bandwidth of the network. It sees from the monitoring that the real bandwidth is lower than 5 Mbps, so it lowers the video bitrate to 2 Mbps at the next available moment (tag 2 in fig. 2.1). Notice that the audio has not yet changed. After further monitoring, the client sees that the real bandwidth is even lower, so it further reduces the bitrate to 0.5 Mbps and switches the audio stream to the lower rate of 48 Kbps (tag 3 in fig 2.1). When the client sees an increase in bandwidth, it increases the bitrate to 2 Mbps (tag 4 in fig. 2.1). After some time the user rewinds the video, so the client requests trick mode, and the video is played in reverse at a very low frame rate; during this rewind the audio is muted (tag 5 in fig 2.1). Then, once the user has rewound to the desired point, they start playing the video in French audio, at which point the client requests French audio at the highest bitrate (128 Kbps) (tag 6 in fig. 2.1).

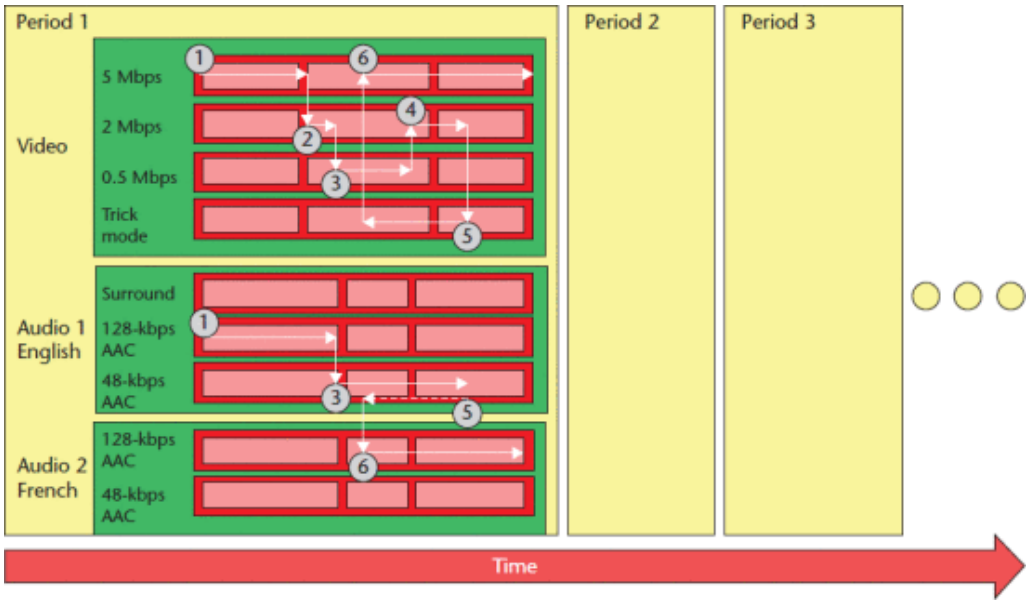



Figure 2.1: Example of ABR streaming. Numbered circles show actions taken by client [12]

### 2.1.2 MPEG-DASH

As mentioned above, MPEG-DASH is built using ABR streaming techniques. When a client requests a video to be streamed, it sends a request to the HTTP server hosting the video. The HTTP server has the video content itself, split into segments, and the manifest file of the video (see figure 2.2). The manifest file, also known as the multimedia presentation description (MPD), provides information about the video. It is sent as an XML file and provides information such as the different audio tracks and subtitles available, as well as the different bitrates the video is being encoded for. When the client receives the manifest file, it has access to all this information about the video, and can select the correct encoding depending on the bandwidth available as well as the settings selected by the user. The client starts fetching the segments using HTTP GET requests, and the stream begins. During the streaming process, the client requests varying encodings of the content depending on changes in the network and preferences selected by the user. The scope of MPEG-DASH is defined as specifically the manifest file and how the segments are encoded.

### 2.1.3 Leak in DASH

The benefits that come with ABR streaming are undeniable, which is why it has such large-scale applications in modern networking. However, in recent years it has been discovered that the network traffic that ABR (thereby MPEG-DASH as well) generates results in a pattern that can be analysed by malicious actors [5]. As described, the first request sent by a client starting to stream is to ask the HTTP server hosting the content to send its manifest file. Then it starts requesting many segments in the manner described in 2.1.1. This fills up the client's buffer, it begins estimating its bandwidth, and requests different bitrates that the segments

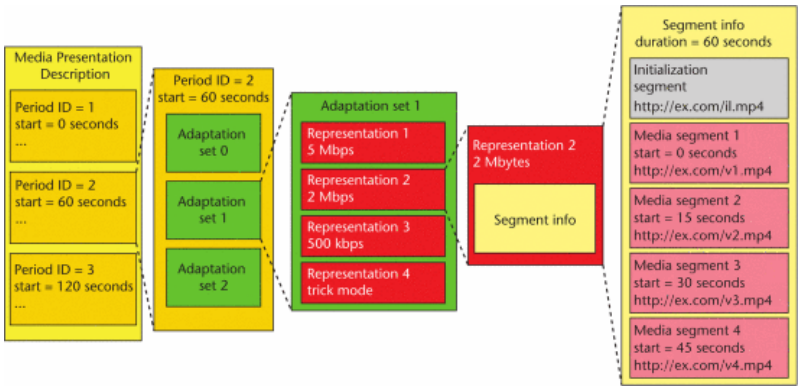



Figure 2.2: MPD Hierarchical Data Model [12]

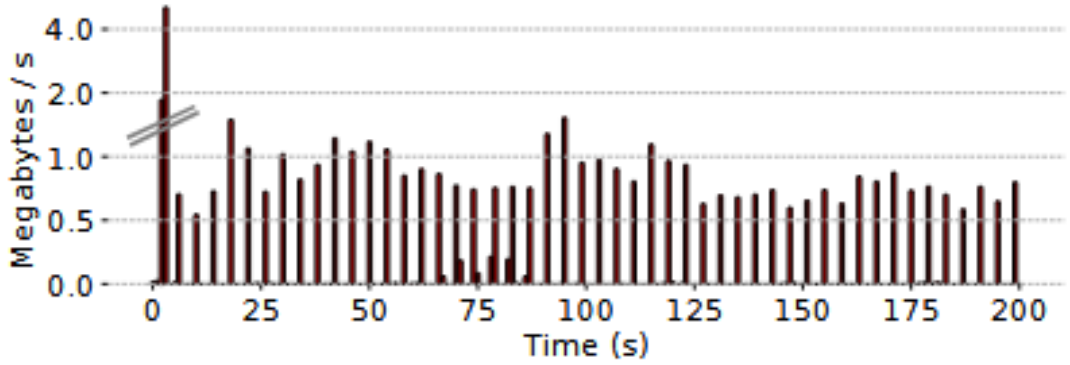



Figure 2.3: *Bursty* behaviour of ABR streaming, with the buffer filling and then transitioning to steady-state [5]

are encoded for. This approximation behaviour creates a *bursty* pattern in the network traffic. This pattern can be seen in figure 2.3. Each column is a new request for new segments at a specific bitrate encoding. The size of each burst/segment is reflective of the kind of content being streamed; for example, scenes with a lot of data, like many colours and quick movements, have more data than scenes that are less complex. The leak is therefore that the kind of data that is transmitted once the stream reaches steady-state based on the network conditions, it starts becoming reflective of the video that is being streamed. This can be attacked using a dictionary attack, where these patterns are saved for every video from a given database, at every available encoding [5].

## 2.2 Endangered Privacy

The attack presented in this thesis uses the attack presented in Endangered Privacy as a starting point for development, changing some aspects of the attack to attempt to better fit the behaviour of DAITA traffic [5]. Endangered Privacy uses the aforementioned leak in MPEG-DASH to classify video traffic using a dataset of video fingerprints collected from the video manifest files. The attack builds on an efficient fingerprint collection method together with a k-d tree for an attack that is

scalable and robust. In their paper, Björklund and Duvignau presented a dataset of 242,364 videos with 3,061,073 different representations from three commonly used video streaming services. In this paper, this is limited to 26,816 videos with 230,582 representations based on the dataset of fingerprints from SVTPlay that was published by Björklund and Duvignau.

With encrypted traffic tunnelled with a VPN, the testing of Endangered privacy found a precision of 1 and a recall of 0.985 for videos streamed from SVTPlay [5]. During the limited testing of 5 videos that was done with DAITA activated they found no matches, although during more extensive testing of this we found that DAITA does not entirely protect against this attack, and this will be further discussed in Chapter 4.

### 2.2.1 Data Organization

In Endangered Privacy, the collected fingerprints are organized into a k-d tree that is used for the lookup stage of collected traffic. To build a k-d tree, the video segments are transformed into points in k-d space. The dataset $S$ is defined as the union of all sizes of segments for all representations. For the set of all representations $I$ where $i$ represents one representation in $I$ let its fingerprint be the set of sizes of segments for a representation $i$, $S_i = (s_{i,1}, s_{i,2}, ...s_{i,n})$. Each representation has its segment sizes normalized in relation to the global minimum and maximum segment size of the dataset. Additionally, a set of normalized fingerprints are defined to make the attack more robust to noisier data when classifying collected traffic, as this gets rid of any constant offset to the traffic. Let $S'_i = (s'_{i,1}, s'_{i,2}, ...s'_{i,n})$ be the set of normalized fingerprint segments where

$$s'_{ij} = \frac{s_{ij} - min(S)}{max(S) - min(S)}$$

Min and Max in this case takes the smallest and greatest segment size in the entire set of segment sizes. The set of the discrete derivatives of the normalized fingerprints $\partial S'_i$ is defined for $i \in I$ such that $\partial S'_i = (\Delta s'_{i,2}, \Delta s'_{i,3}, ...\Delta s'_{i,n})$ where $\Delta s'_{i,j} = s'_{i,j} - s'_{i,j-1}$

A window $W_i^k$ is defined for each representation.

$$W_i^k = \begin{bmatrix} \Delta s'_{i,2} & \Delta s'_{i,3} & ... & \Delta s'_{i,k+1} \\ \Delta s'_{i,3} & \Delta s'_{i,4} & ... & \Delta s'_{i,k+2} \\ \vdots & \vdots & \ddots & \vdots \\ \Delta s'_{i,n_i-k+1} & \Delta s'_{i,n_i-k+2} & ... & \Delta s'_{i,n_i} \end{bmatrix}$$

Each row $W_i^k[\ell]$ in this window corresponds to a point in k-d space and from there the dataset of all videos is defined as

$$\mathcal{D}_k = \bigcup_{i \in I} W_i^k$$

Finally, the database is populated with $\langle W_i^k[\ell], i, \ell \rangle$ triplets. As mentioned previously, $W_i^k[\ell]$ corresponds to the k-d point used to calculate the distance during the lookup of the k-d tree.

### 2.2.2 Identification Logic

Similarly to the fingerprints, the collected video data must be transformed into k-d points in order to allow for classification. Firstly, the tshark wrapper *Burstshark* is run on a network trace to get the sizes of bursts, then the bursts are transformed in the same way that the fingerprinted segment sizes were transformed when building the k-d tree.

Let $X$ be the set of bursts in a capture so that each burst $x_t$ and time $t$ $x_t \in X$. A set of normalized bursts is defined as $X' = (\Delta x'_2, \Delta x'_3, ...\Delta x'_n)$ where $\Delta x'_t = x'_t - x'_{t-1}$ and

$$x'_t = \frac{x_t - min(S)}{max(S) - min(S)}$$

In contrast to previous works utilizing k-d trees [13], [14] Endangered privacy does not rely on a single query to classify an identification as positive due to the increased noise handled by the attack, as that would result in false positives [5]. Endangered privacy instead relies on candidates. A candidate is defined as any non-empty alignment group $A(i,j)_t$ at time $t$ where the alignment group $(i,j)$ correspond to a number of triples satisfying the criteria $i_a = i_b \ \wedge \ \ell_b - \ell_a = t_b - t_a$. That is: they correspond to the same video, and they are aligned relative to each other temporally so that the matched segments' offset is equal to the offset of the observed bursts.

The candidates are then scored and ranked using a scoring function and compared to a confidence threshold $\theta$, and if the score is greater than $\theta$ the candidate is predicted as a match.

## 2.3 DAITA

DAITA is a powerful tool for protecting against these types of attacks [15]. DAITA is built on the Maybenot framework, which allows for obfuscation of side-channel data. Maybenot is a framework for building probabilistic finite state machines that trigger different actions depending on the network. This means that the machines can be implemented for different protocols and networks, and provide obfuscation that looks more realistic rather than simply padding each packet to be a constant length and sent at a constant rate [9]. DAITA is used to add padding to packets to conceal information about packet size. It works by sampling time values, padding windows, and padding counts, and adds padding at those sampled times. This padding variation allows for randomness, making it more difficult for an attacker to train a classifier to identify the new patterns after adding padding [8]. This randomness is configured at each new connection, which makes it very difficult for a sniffer to distinguish real communication from the noise added by DAITA.

Currently, Endangered Privacy is incapable of reliably breaking the defences put in place by DAITA, as it does not account for dummy packets and data pattern distortion. In the previous work, tests with DAITA did not yield any success [5]. During this work, an experimental success rate of about 20% was observed. If the attack is performed while DAITA is active, the fingerprints look different not only

from the ones collected in the database (due to dummy packets and package padding) they also look different to one another when compared to the same video streamed again from a different connection. This makes it difficult to recreate such a database with DAITA-affected fingerprints.

## 2.4 Attack Model

The attack model used is that of an attacker on the network path with a target using a VPN, with DAITA being used for additional obfuscation. As discussed above, this will cause all of the target's traffic to be sent through an encrypted tunnel, hiding any network layer information of the real communication. While this paper limits its dataset to videos streamed from SVTPlay, in a real-world scenario this means that an attacker is unable to identify what service is being used to stream from, the attacker would therefore be required to query the entire set of videos available in the worst case.

Another consideration that has to be made is the inability to separate the video traffic from any other background traffic, although one assumption that is made in Endangered Privacy and this paper is that any background traffic remains fairly constant [5]. This assumption is made as a target streaming a video is likely to be doing that as a primary activity, not generating much traffic in the background, although it does lead to a certain baseline of noise. Using the data representation of $\Delta x'_t$ we can see that adding a constant background traffic $C$ does not affect the resulting values of the identification.

$$\frac{x_t - \min(S)}{\max(S) - \min(S)} - \frac{x_{t-1} - \min(S)}{\max(S) - \min(S)} = \frac{x_t + C - \min(S)}{\max(S) - \min(S)} - \frac{x_{t-1} + C - \min(S)}{\max(S) - \min(S)}$$

# 3

# Methods

This section will discuss the methodology used when this research was conducted. It will discuss the steps taken to prepare for conducting the research and mention the papers that were read to gain a deeper understanding of the different protocols that were analysed. Then it will go on to discuss the tests that were run to understand where the vulnerabilities lay in DAITA. Then a description of the attack development will be given, including how the problem was defined and the steps taken to solve it. Evaluation metrics for results will also be discussed, as well as the environments and modelling tools that were used to conduct the research.

## 3.1 Preparation

The first part of the project was to study different existing attacks on the MPEG-DASH protocol to see the differences in approach and accuracy. The main attack being Endangered Privacy [5] which served as a base from which we developed the attack. Then we captured data of video streams, unprotected, using WireGuard, and using DAITA to see how the behaviour changed and to see if there remains any characteristics about the bandwidth pattern that could be used to recreate any attack input. Videos were captured for 10 minutes, multiple videos were captured with different Mullvad connections, and using the same connection to see to what extent resetting the connection affects difference in the traffic patterns. This provided further insights for us as to what extent the "bursty" traffic pattern was preserved, and to what extent modifying *Endangered Privacy* was feasible. We did this by mapping the data and comparing various parameters such as the number of packets, number of bytes, total transfer time and more.

Several different plots were made to gain an intuition of how the streamed video data behaved with DAITA compared to without it, how the same video would look compared to other videos, and compared to itself using the same and different DAITA configurations (streaming a video multiple times, either reconnecting to the Mullvad server or not). We found that the behaviour seen in DAITA seems to be fairly dependent on the particular configuration parameters used for each connection. The difference between the same video streamed with the same DAITA configuration was not great, as the bursts were very consistent in size as can be seen when looking at a post processed B/s graph processed with a box filter.

Once that had been done, we worked on recreating the attack from Endangered

Privacy and understanding it on a technical level. This was crucial for successfully modifying the attack later on.

## 3.2 Attack Development

A problem that arose for the attack when using DAITA was that the traffic appears to be scaled by some factor, Endangered Privacy observed approximately a doubling of the traffic during its tests with DAITA [5]. This scaling is an issue for the identification logic as can be seen when calculating $\Delta x_t$ for a point and its scaled counterpart. An assumption was made that traffic is scaled by a constant scaling factor. Tests that were conducted later indicated that the scaling is not simply modelled as a multiplicative scaling constant. When a multiplicative scaling factor $\rho$ is assumed in

$$\frac{x_t - \min(S)}{\max(S) - \min(S)} - \frac{x_{t-1} - \min(S)}{\max(S) - \min(S)} = \frac{x_t - x_{t-1}}{\max(S) - \min(S)} = \Delta x_t$$

and

$$\frac{\rho x_t - \min(S)}{\max(S) - \min(S)} - \frac{\rho x_{t-1} - \min(S)}{\max(S) - \min(S)}$$
$$= \frac{\rho x_t - \rho x_{t-1}}{\max(S) - \min(S)} = \rho \frac{x_t - x_{t-1}}{\max(S) - \min(S)} = \rho \Delta x_t$$

As seen in the equation, a constant scaling applied to the traffic results in equal scaling to the discrete derivative of the normalized burst. To get rid of the factor $\rho$ a sliding window $W_{t,t+w}$ was considered where $W_{t,t+w} = (x_t, x_{t+1}, ...x_{t+w})$ for a window size $w \geq 2$. Let

$$\Delta x_t^W = \frac{x_t - \min(W_{t,t+w})}{\max(W_{t,t+w}) - \min(W_{t,t+w})} - \frac{x_{t-1} - \min(W_{t,t+w})}{\max(W_{t,t+w}) - \min(W_{t,t+w})}$$

Assuming that all packets in the window $W_{t,t+w}$ are scaled by a factor of $\rho$ this would create an equality for $\Delta x_t^W$ for the set $X = (x_1, x_2, ...)$ and its scaled counterpart $X^\rho = (\rho x_1, \rho x_2, ...)$ as seen in the equation:

$$\frac{\rho x_t - \min(\rho W_{t,t+w})}{\max(\rho W_{t,t+w}) - \min(\rho W_{t,t+w})} - \frac{\rho x_{t-1} - \min(\rho W_{t,t+w})}{\max(\rho W_{t,t+w}) - \min(\rho W_{t,t+w})}$$

$$= \frac{\rho x_t - \rho x_{t-1}}{\max(\rho W_{t,t+w}) - \min(\rho W_{t,t+w})} = \frac{\rho x_t - \rho x_{t-1}}{\rho \max(W_{t,t+w}) - \rho \min(W_{t,t+w})}$$

$$= \frac{\rho(x_t - x_{t-1})}{\rho(\max(W_{t,t+w}) - \min(W_{t,t+w}))} = \Delta x_t^W$$

As the window method of calculating the normalization is robust against any constant scaling factor independent of its source, it may allow for reductions in the dataset depending on how a service handles segmentation of videos of different qualities. This will be discussed further in chapter 4.

Additionally, two different implementations of this method were tried, one with a typical sliding window that for any point $x_t$ and window $w$ the normalization is

$$\frac{x_t}{max(W_{t,t+w}) - min(W_{t,t+w})} \tag{3.1}$$

The other implementation set $w = k$ and normalized within the already existing k-d points as in an arbitrary normalized k-d point

$$[\frac{x_t - min(W_{t,t+k})}{max(W_{t,t+k}) - min(W_{t,t+k})} \ ... \ \frac{x_{t+k} - min(W_{t,t+k})}{max(W_{t,t+k}) - min(W_{t,t+k})}] \tag{3.2}$$

This results in a minimum and maximum per k-d point that each burst is normalized with, in comparison to the first implementation, where each burst was normalized the same way across each k-d point, but where the bursts in a k-d point had different maxima and minima.

One major difference between the different methods is that the latter presents a ”clamping” behaviour, as all points are forced to take a value between 0 and 1. The difference between the implementations is in part due to the constant level of noise that is assumed in the traffic. As the normalization in the latter method happens after the points have been organized into k-d points, they are not differentiated as that would cause issues in respect as the minimum and maximum point would no longer be representative of the bandwidth. The normalization must therefore take care of the noise in the numerator. In the former method, that is not the case as the derivation happens after the normalization, which eliminates the noise. While a $-min(W_{t,t+w})$ term in the numerator would not cause issues in regard to the level of noise present, the decision to leave it out was made as it was theorized that the clamping behaviour would lead to a loss in resolution of the points. The difference of the methods is illustrated in figure 3.1 and 3.2.

## 3.3 Evaluation

In order to evaluate the efficacy of the attack against DAITA, an analysis of the required resources and time was done. A comparison will be drawn between both the developed attack and *Endangered Privacy* on whether any major changes in performance occurred after the changes made to the attack. This is important because one goal of the thesis was to not worsen the performance compared to *Endangered Privacy* and if possible, to reduce it. One important aspect in the efficiency compared to *Endangered Privacy*, was to see if and how the database size can be reduced. The current size of the database is 9 GB, and takes around 1–2 minutes to respond and after 5–6 minutes of running it can be deduced that the

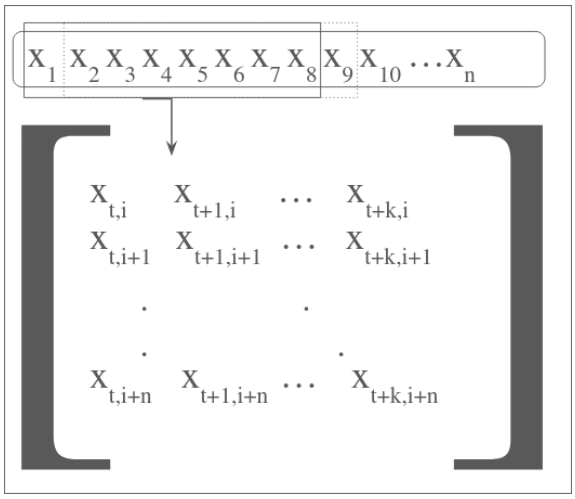



Figure 3.1: The normalization as shown in equation 3.1. The sliding window is shown as step one with a solid line and step two with a dotted line, illustrating where the normalization is made. Arrow implies bursts/segments being arranged into k-d points.

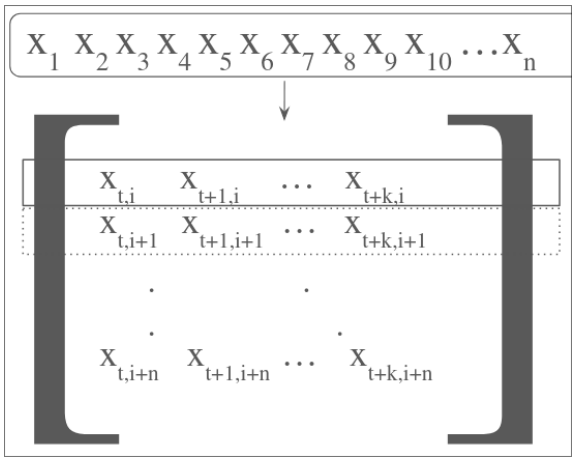



Figure 3.2: The normalization as shown in equation 3.2. The sliding window is shown as step one with a solid line and step two with a dotted line, illustrating where the normalization is made. Arrow implies bursts/segments being arranged into k-d points.

video is not in the database [5]. The attack will also be analysed based on its large-scale applicability. This is important to keep the realistic large-scale applications of the attack intact.

The collection of video streams was done in a virtualized environment using Kali Linux version 2025.2 with 4 GB of memory allocated. Mullvad VPN version 2025.9, and tshark version 4.4.6 were used. The Mullvad locations used were those available in Sweden (Stockholm, Gothenburg, and Malmö) with multihop enabled when collecting DAITA streams, to ensure that DAITA was used in those cases. The streams were collected between September and November 2025.

The attack was run using Burstshark version 0.1.0, and tshark version 4.4.6.

## 3.4 Environments and Modelling

The environment used was the same as in *Endangered Privacy*. The pre-captured video streams from the streaming service SVT were used as the primary dataset in the project. Specifically, SVTplay was chosen as the streaming service as it is free to view, and is a popular streaming service used by the Swedish population. Wireshark/tshark was used to capture streams of videos that were obfuscated using DAITA. This served as the dataset for testing the modified attack. Additionally, the tools developed for data collection and processing for Endangered Privacy: Karl, and Burstshark will be used to collect any fingerprints and extract bursts from packet captures. Burstshark is a tool developed for extracting bursts from network traffic, which is a crucial pre-processing step for the attack to work. Karl is used for retrieving DASH manifests, and extracting the fingerprints from them, for each representation of a video given a URL pointing to the source where the video could be streamed from.

For a weak attacker (an attacker unable to determine what service is being used) all available fingerprint-trees are queried, although in this project all tests are done with a weak attacker as the traffic is tunnelled using a VPN. Additionally, as this project limits the dataset and traffic to videos from SVTPlay only one tree is queried during the testing and evaluation.

# 4

# Results

This chapter aims to present the various results collected in the duration of this research, and discuss the efficacy of the methodology used. The results of this thesis were focused around gaining a deeper understanding of both Mullvad's DAITA protocol and the attack Endangered Privacy, and how applicable it is against DAITA traffic. In this section, both elements shall be discussed, beginning with a re-examination of Endangered Privacy on DAITA, this yielded results different than those seen in the original presentation of the attack [5]. Then, the results of the aforementioned *sliding window attack* will be discussed, along with the performance of various iterative changes that were tested.

## 4.1 Further testing of Endangered Privacy

In the original paper presenting Endangered Privacy [5] it was believed that the attack was unable to identify any videos captured with traffic using DAITA. The testing that was performed in the paper, was done on different videos, each streamed using the same Mullvad connection, and therefore the same DAITA configuration. When the attack was applied to these streams, it resulted in zero identified videos, and the assumption was made that Endangered Privacy was unable to bypass DAITA. This differs from the results found in this thesis.

This thesis attempted to re-perform this same test, but at a larger scale to understand in a more detailed manner how Endangered Privacy performs against DAITA. The tests were run on 50 streams, that consisted of the same five videos streamed 10 times with a different connection each time. After having run the attack on these 50 streams, the conclusion is drawn that Endangered Privacy is indeed able to bypass the obfuscation added by DAITA, on an average at a hit rate of 22%. The tests were performed by streaming five videos, 10 times each, with each stream being reconnected to a new instance of Mullvad, getting a different DAITA configuration. The attack was then run on these streams, and yielded the results that can be seen in figure 4.1. As can be seen, the rates average out to approximately 22%.

These results implied that there may be certain connections that are weaker than others and therefore more susceptible to the attack than others. This would explain the 22% hit rate. As every stream in this data was collected using a new connection, it was difficult to gain any deeper understanding of the possible vulnerability of certain connections.

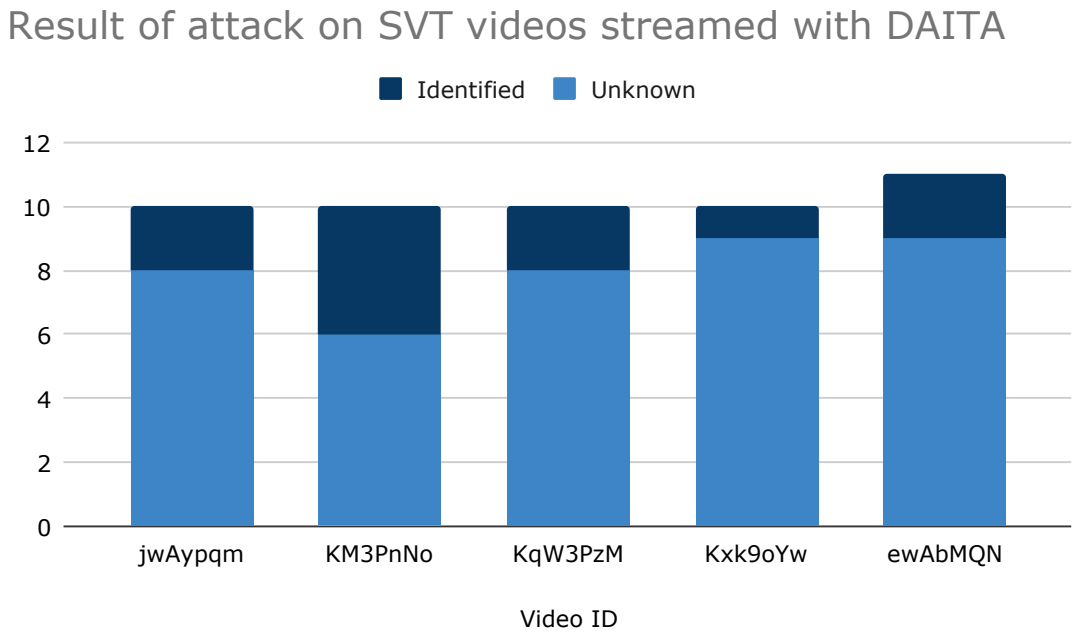



Figure 4.1: Endangered Privacy [5] applied on SVT videos streamed using DAITA. DAITA connection re-established for every stream.

In order to get more data about the performance of specific connections, a new test was conducted that connected to Mullvad using DAITA once, and then five different videos were streamed one at a time, without interrupting the connection. This showed how the same video can react in different connections, while comparing how the same connection protects the different videos. The results of these tests can be seen in table 4.1.

In order to see how the different videos behaved under different connections, plots were drawn to see the bitstreams of the same video under different connections (see figures 4.2 and 4.3). It is important to note that these plots are normalized, in order to more clearly see the similarities, no matter what bandwidth the stream resulted in. They firstly display that the bursty behaviour of MPEG-DASH is preserved, despite DAITA's obfuscation being in place. They also show that there is not a lot of difference between the same video in different connections, and follow a similar pattern.

## 4.2 Sliding Window Attack

This section will present the results regarding the two implementations of the sliding window normalization, how it performed against DAITA traffic, VPN traffic without DAITA, and its capabilities on operating with a reduced dataset.

As previously discussed, two differing methods of implementing a sliding window were made, and from these differences some properties emerge when looking at

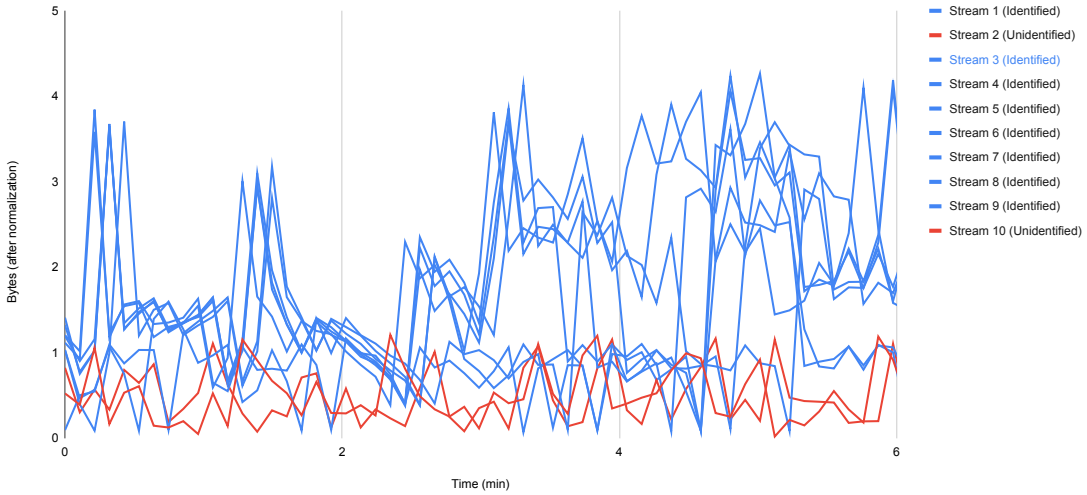



Figure 4.2: Graph showing sizes of normalized bursts of the video *eEd5x48* for each of the connections as presented in Table 4.1. Identified shown in blue and unidentified shown in red.

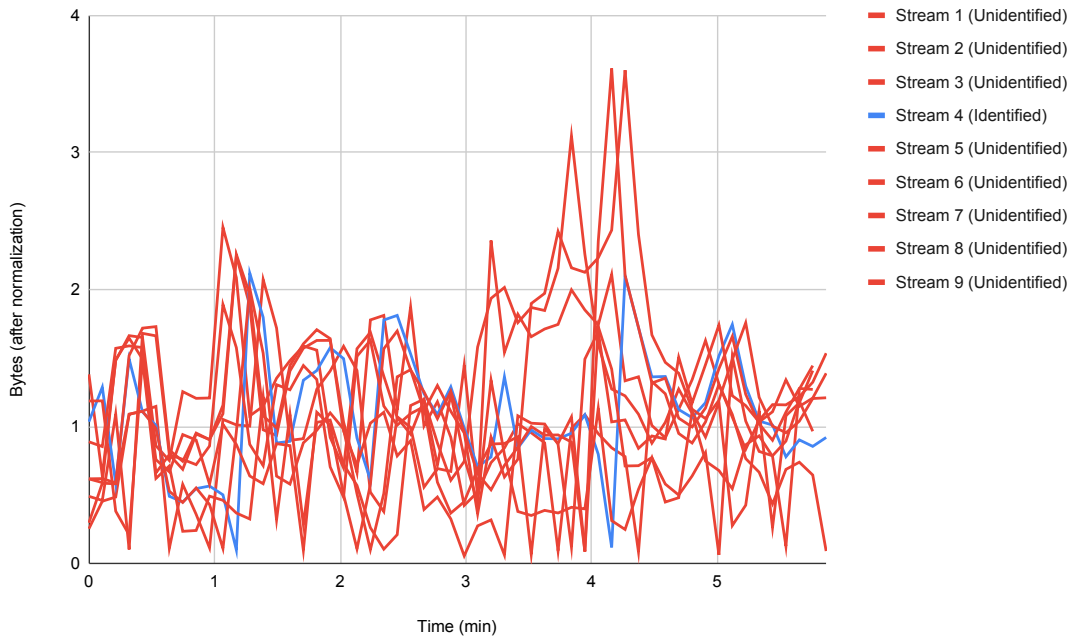



Figure 4.3: Graph showing sizes of normalized bursts of the video *Kxk9oYw* for each of the connections as presented in Table 4.1. Identified shown in blue and unidentified shown in red.

| Connection nr | Kxk9oYw | KM3PnNo | KqW3PzM | eEd5x48 | ewAbMQN |
|---|---|---|---|---|---|
| **1** | 0 | 0 | 0 | 1 | 0 |
| **2** | 0 | 0 | 0 | 0 | 0 |
| **3** | 0 | 1 | 0 | 1 | 1 |
| **4** | 1 | 1 | 1 | 1 | 1 |
| **5** | 0 | 1 | 0 | 1 | 1 |
| **6** | 0 | 1 | 1 | 1 | 1 |
| **7** | 0 | 0 | 0 | 0 | 0 |
| **8** | 0 | 1 | 0 | 1 | 0 |
| **9** | 0 | 0 | 0 | 1 | 0 |
| **10** | 0 | 0 | 0 | 0 | 0 |

Table 4.1: Table showing identifiability of videos under different connections, where 1 is an identified video and 0 is unidentified. Same DAITA connection used for every video, then reset for next iteration.

normalized traffic. As can be seen in the figures below. The method shown with 3.2 exhibits a clamping behaviour, which reduces the resolution of the data as it forces it to be within the range $(0, 1)$. Furthermore this clamping seems to reduce the apparent level of noise present in the traffic, although this is most likely an effect of the resolution loss, as the main signal is reduced as well. Additionally a smaller window size emphasises traffic locally more than larger window sizes as seen when comparing figure 4.5 and 4.6.

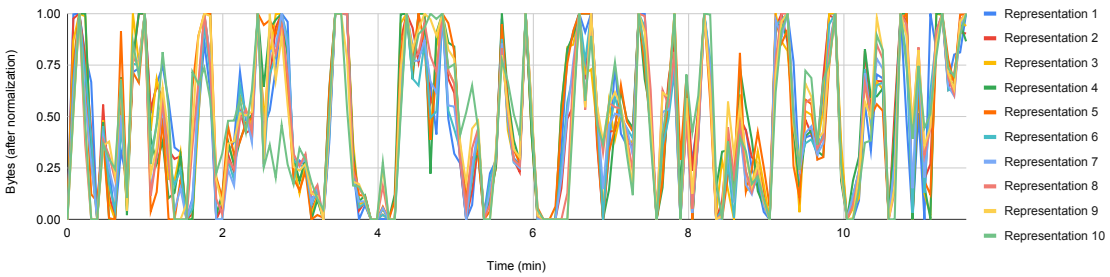



Figure 4.4: 10 representations of a video normalized with the method described in equation 3.2 and figure 3.2, clearly showing the clamping behaviour previously discussed.

### 4.2.1 Against DAITA

Both methods of normalizing using a sliding window were tested on the testing set shown in Figure 4.1, with values of $\theta$ (threshold score) of 2.2 (default), 1.5, and 1. All tests using the method shown in Figure 3.2 yielded no positives for $\theta > 1$, and $\theta = 1$ resulted in one true positive as well as three false positives. For the tests using the other normalization method, all tested values of $\theta$ resulted in all videos being unidentified for window sizes of 4, 8, and 16.

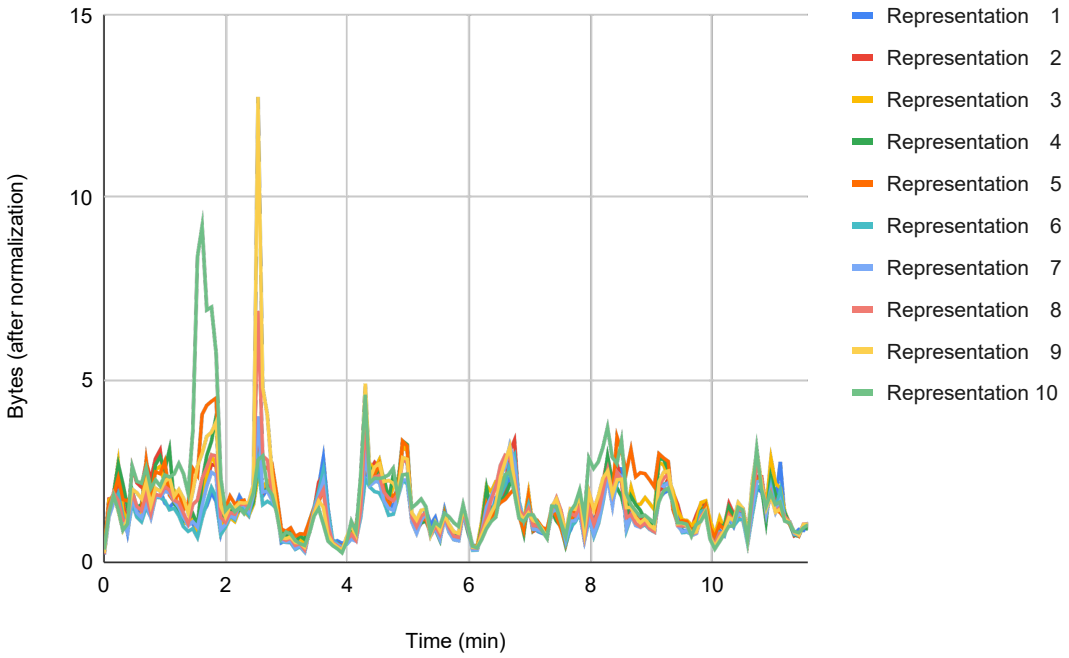



Figure 4.5: 10 representations of a video normalized with the method outlined in equation 3.1 and figure 3.1 with a window size of 8

| Window size | $\theta$ | Identified | Misidentified |
|---|---|---|---|
| 16 | 2.2 | 0 | 0 |
| 16 | 1.5 | 0 | 0 |
| 16 | 1.0 | 0 | 1 |
| 16 | 0.75 | 0 | 1 |
| 8 | 2.2 | 0 | 0 |
| 8 | 1.5 | 0 | 0 |
| 8 | 1.0 | 0 | 0 |
| 8 | 0.75 | 0 | 0 |
| 4 | 2.2 | 0 | 0 |
| 4 | 1.5 | 0 | 0 |
| 4 | 1.0 | 0 | 0 |
| 4 | 0.75 | 0 | 0 |

Table 4.2: Running the attack with different window sizes and values of $\theta$ on the streams of video **ewAbMQN** seen in figure 4.1

### 4.2.2 Database Reduction

Running the attack on the entire SVT dataset uses 9.3 GB of memory, since there are several representations of the same video, reducing the amount of representations would offer significant improvements on the hardware requirements of the attack. The attack was tested with reduced datasets; Testing with one representation per video, the attack used 1.4 GB of memory while executing, an improvement of roughly

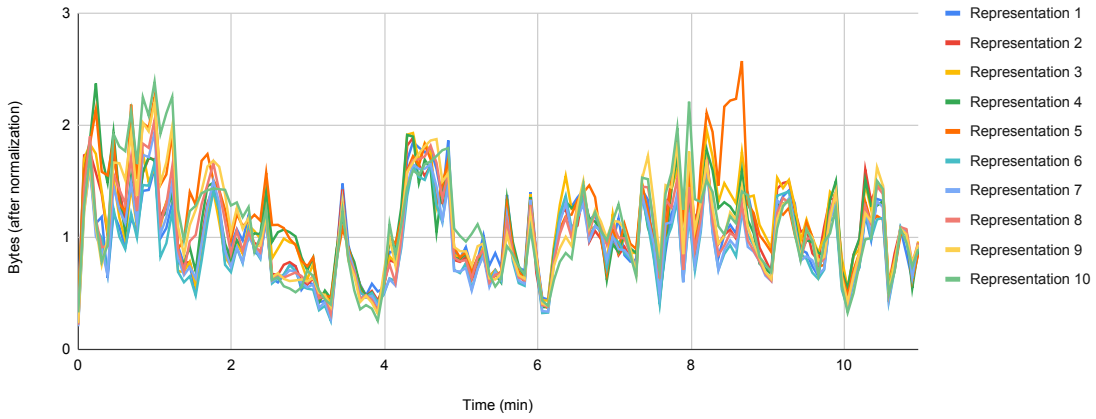



Figure 4.6: 10 representations of a video normalized with the method outlined in equation 3.1 and figure 3.1 with a window size of 16

| Dataset variant | Memory usage | Precision | Recall |
|---|---|---|---|
| Full | 9.3 GB | 1.0 | 0.985 |
| Keep highest | 1.4GB | 1.0 | 0.985 |
| Keep second highest | 1.4GB | N/A* | 0.0 |

Table 4.3: Running the attack on the testing set of 276 videos from *Endangered Privacy* using three different datasets. (* No videos successfully identified)

85%. Although the degree of reduction that is possible will vary between different datasets. It will vary based on the overall size of the dataset (non-dataset related memory usage being a smaller fraction for larger datasets), and the amount of representations a given service uses per video (SVT offers 10 representations for most videos).

The scaling between different representations of the same video is not perfectly modelled as $i_{high} = \rho i_{low}$ although the difference between qualities is not great. Considering a set of representations $I$ where $i \in I$ is each representation let $s_{ij}$ be a segment in $i$. A set of normalized segments $I'$ is defined as

$$I' = \bigcup_{i \in I} i'$$

where

$$i' = (\frac{s_{i0} - min(i)}{max(i) - min(i)}, \frac{s_{i1} - min(i)}{max(i) - min(i)} ... \frac{s_{in} - min(i)}{max(i) - min(i)})$$

For each pair of representations a differential set $\Delta i'_{ab}$ where $a$ and $b$ are different representations of the same video, is defined as

$$(|s'_{a0} - s'_{b0}|, |s'_{a1} - s'_{b1}| ... |s'_{an} - s'_{bn}|)$$

From $\Delta i'_{ab}$ we can derive a set of preserved segments and k-d points. The set of preserved segments $P_{seg}$ for a given tolerance $\epsilon$ is defined as

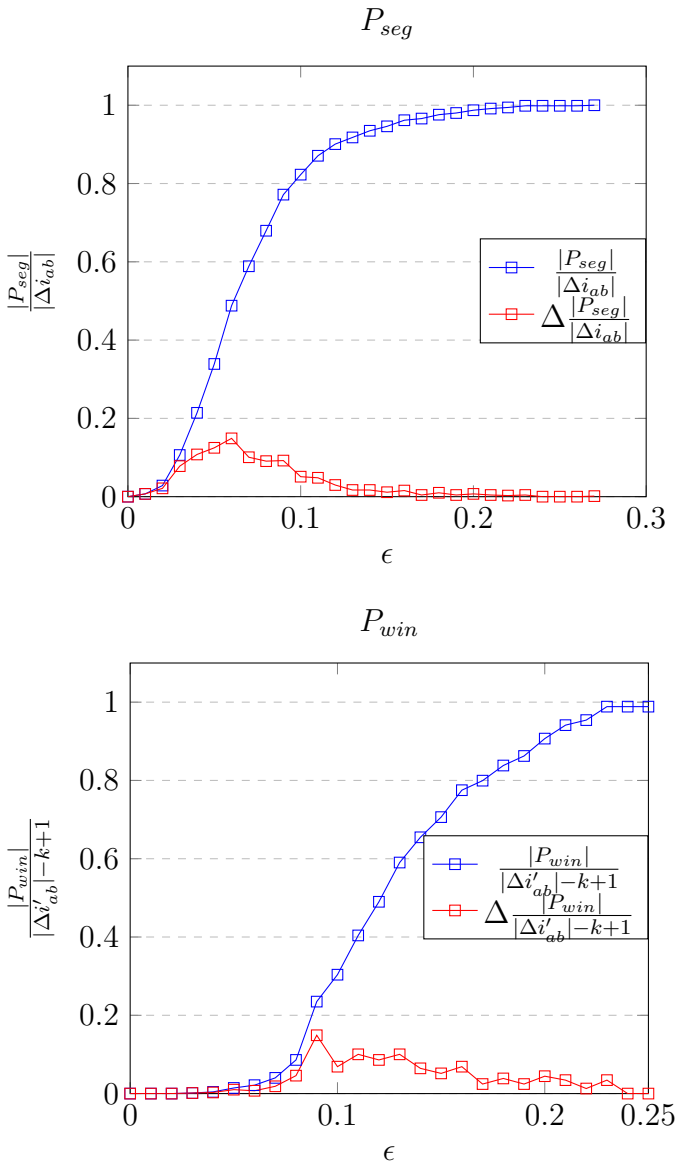
$$\{\Delta s'_j \in \Delta i'_{ab} | \Delta s'_j \leq \epsilon\}$$

Similarly, the set of preserved k-d points $P_{win}$ allowing for a tolerance of $\epsilon$ per segment is defined as

$$P_{win} = \{(\delta_j, \ldots, \delta_{j+k-1}) \mid \delta_n \leq \epsilon \; \forall n = j, \ldots, j+k-1\}$$

where $\delta_j = |s'_{a,j} - s'_{b,j}|$.

Note that this is equivalent to the difference of k normalized segments in a row being less than $\epsilon$. Given the notation for the k-d point window presented in section 2.2.1. We can then look at $P_{seg}$ and $P_{win}$ for different values of $\epsilon$. (note that for $P_{win}$ $k$ is set to 8 as it is in the attack.)




To test whether the sliding window approach could be used to reduce the database, a set of 25 videos from the testing set of Endangered Privacy was used [5]. Three reduced datasets were constructed, one using the highest quality representation for each video, one using the second highest, and one using the lowest. The first round

of tests made were done using the sliding window approach that uses the same maximum and minimum within each k-d point.

Running the attack with the dataset comprised of using the highest quality representation resulted in a precision of 1.0 and recall of 0.985. To verify these results, the same dataset and testing set were used running the attack using the normalization used in Endangered Privacy. This also resulted in all true positives, the conclusion drawn from this is that all videos in the testing set were streamed using the highest quality representation.

After we noticed that all streams were captured using the highest quality representation, the attack was run using the other two reduced datasets. This resulted in all streams being unidentified, for both sliding window methods, and for the normalization used in Endangered Privacy as control. For the normalization method using a window size not tied to $k$ (the dimensionality of the points), the window sizes tested were 4, 8, 12, and 16. The performance of both methods of introducing a window for normalization were similar for the full dataset, although the attack was not successful for a window size of 4.

Attempts to re-tune the parameters were made for both implementations, specifically the variables representing score threshold, query size, and, "anti-cluster radius": $\theta$, $\tau$, and $r$ were varied to evaluate whether the changes in the k-d transformation would result in an untuned system.

For the first tuning tests, $\theta$ was changed, and the values 2.2 (default system parameter), $1.5, 1.0$, and 0.75 were tested. For a $\theta$ of 2.2 and 1.5 no positives were found, when running with $\theta = 1.0$, one out of 25 videos were identified, and with $\theta = 0.75$ three false positives were produced in addition to the previous true positive, these results were when testing the method with per k-d point minimum and maximum. For the other method, normalizing each segment equally across k-d points, no true positives were present.

Both methods also tested for different values of $r$, as the normalization forces some k-d points to take up the value of either 0 or 1 (that being the minima and maxima). The values tested were 0.025, 0.0125, and 0.005 yet this did not yield any difference in the resulting identifications, both for $\theta = 2.2$ and $\theta = 1.0$.

For the number of queried points, $\tau$ the values tested were 50 (used by Endangered Privacy) 100, 200, and 500. For all values tested below 500 there were no changes for any of the implementations and for a $\tau$ of 500 a portion of videos were misidentified while none were identified correctly. The reasoning behind testing larger values for $\tau$ may result in certain points being normalized close together in k-d space (the clamping behaviour seen in 4.4 results in points falling within $2\binom{k}{2}$ different hyperplanes of dimension $k-2$), which could have resulted in more points being closer to any given queried point. Therefore, attempts to query with smaller than default $\tau$ were not made.

# 5

# Discussion

This chapter aims to discuss the results found in the duration of this thesis. It will delve deeper into the results presented above, as well as discuss possible reasoning behind the results. It will also discuss what further insight these results provide, and the conclusions that can be drawn from this. There will then be a discussion provided regarding areas of interest in this field, and further research that is required.

## 5.1 DAITA

This section will discuss the results we have gained both regarding running the original attack, and both methods of normalizing using a sliding window.

### 5.1.1 Endangered Privacy

It was previously thought, that DAITA provided full protection against attacks like Endangered Privacy [5]. After further testing this thesis has shown that that is not the case. In our tests with the attack only 78% of DAITA streams remain unidentified. The reason for this is because of the root cause of why video fingerprinting works in the first place; the bursty nature of MPEG-DASH. DAITA as a protocol, aims to obfuscate network traffic by adding padding to the data that is sent over a connection. While the padding that DAITA adds to the video does change it from its original representation, it does not remove the inherently bursty nature of the video transfer protocol itself. This bursty nature can be seen in figures 4.4, 4.5 and 4.6. This then begs the question, why DAITA is able to protect against Endangered Privacy at all. The explanation to this can be demonstrated through figures 4.2 and 4.3. Fingerprinting attacks such as Endangered Privacy work by mapping the burst pattern to a pre-collected video fingerprint that exists in the attack database, that can be created by fetching the manifest file. The attack then maps the bitstream of the stream to the fingerprint of the representation that is most similar. The fingerprint is made up of a pattern of amount of data sent at specific timed intervals. When DAITA obfuscation is added to the network traffic, the amounts of data are changed, so often the representation does not match any one representation well enough to be able to identify which video is being streamed.

Another interesting set of results when running the attack against DAITA is what is shown in table 4.1. These tests show that not only may some connections be

more vulnerable to Endangered privacy (see connection 4 and 6 in 4.1, but also that the vulnerability to Endangered Privacy is not only limited to which connection that the traffic is obfuscated by, but also on the video that is being sent (see video eEd5x48 having 70% hits). In order to gain deeper insight into this relation, more tests would need to be conducted confirming this behaviour, and normalizing it to compare what kinds of video bitstreams lead to more difficulty identifying videos from fingerprints.

For some videos the traffic pattern remained very stable, as can be seen in figure 4.2. All traces but two very closely follow the same pattern with some variation in behaviour but when normalizing according to the normalization put forward in equation 3.1 (while the sliding window was not used for the tests presented in 4.1, normalizing with one is still useful for analyzing the data) with a window size of 8, the local behaviour is well-preserved as all successful (8/10 connections) traces stay roughly within the same values. We notice that the traces that fail show a similar behaviour when looking at the sign of the derivative (if following burst(s) increase or decrease in size), which is explained by a smaller denominator in Equation 3.1 relative to the burst, which in turn means that the bursts are not scaled to the same degree.

Looking at the graph of a video that was more difficult to identify such as *Kxk9oYw* shown in figure 4.3, the pattern is almost impossible to deduce. There are some preserved local characteristics as can be seen at the steep peak on the left-hand side of the graph, where all but two connections exhibited the same behaviour at roughly the same level, but this quickly falls apart for all other connections, only for brief moments of 4–6 bursts following the trace that was successfully identified. In this graph, we can see peaks that are far too high on the right-hand side, this would be the result of smaller bursts receiving more padding relative to their size compared to larger bursts. The implications of this will be further discussed in the next subsection.

### 5.1.2 Sliding Window

The result of running the attack on the non-DAITA dataset shows that both implementations of the sliding window normalization are valid ways to normalize the traffic, as the performance was equal to that of Endangered Privacy [5] (excluding the implementation shown in figure 3.1 that has window sizes that are too small). The reason for this normalization being valid for non-DAITA traffic, using a complete dataset is likely due to two important characteristics.

First and foremost the normalization is consistent, meaning that for any point being normalized, any other values used for normalizing have the same reference each time it is normalized. In the attack presented in Endangered Privacy, each point was interpolated with respect to the minimum and maximum of the segments in the dataset, this minimum and maximum being used every burst [5]. In the sliding window approach to normalization we have presented, the attack still has consistency within any given window, either each burst in a k-d point sharing the same window, or a window looking ahead of any given burst. This means that the way fingerprint

segments and traffic bursts are treated the same way and share the same points for any window.

Secondly, the approach does not lead to points of differing scale compared to the underlying attack. This property is important for the scoring function, and candidate selection to work with default parameters. Given burst sizes within the minimum and maximum of the dataset, the attack presented in Endangered Privacy will result in points falling within a k-cube with side length of two, centred at origin. For the window method shown in figure 3.2, points will always fall within the positive half of the aforementioned k-cube and for the method underlined in 3.1, the data is sparser and not limited to the cube, although it is centred and with values tending to fall within $(-1, 1)$, with the most extreme values observed during testing reaching as far as 4, which is further reduced during derivation.

One thing that is apparent with the sliding window methods, for the method in figure 3.2 in particular is that this emphasizes local behaviour, disproportionally. This is most notable for the method outlined in figure 3.2, as a difference of 1 byte between the minimum and maximum would result in values between 0 and 1 regardless of the magnitude of the data. This could lead to further issues as this method of normalizing results in points along certain planes. If any noise introduced by DAITA's padding would cause a wrong packet to a minimum or maximum, this would cause the point to fall into a different plane than that of the ground truth. Although this is not guaranteed to cause any issues, as these planes are only two dimensions lower than $k$, it is still important to be aware of this topology. It could be the case that more points within a plane are closer to some points within the same plane than the same point being normalized into the wrong plane due to noise.

As previously mentioned, it is possible that smaller bursts receive more or less padding relative to their size. This kind of noise is problematic for both methods of normalizing. For the clamping method, it simply results in the region between 0 and 1 being non-linear, which gives different normalizations depending on what level of this kind of noise is present. For the other method, this presents as even greater problems as a smaller or larger difference between small and large bursts would result in a potentially large shift in distance, in figure 4.2 and 4.3 it can clearly be seen that for this method, this kind of noise could result in k-d points that are more than halfway across the tree. This would mean that hundreds of thousands of points are closer to the point than the actual ground truth, which would make any attempt to find the underlying truth fruitless.

Regarding the database reduction, similar problems arise regarding the amplification of local behaviour. The difference between two segments of different representations of the same video is small when accounting for bandwidth, but applying the sliding window normalization amplifies this noise, resulting in traffic that is too noisy to consistently identify the right video.

## 5.2 Scalability

The underlying attack this work builds on is shown to have large-scale applicability, as it uses the largest dataset to date [5]. Additionally, its accuracy makes it a very reliable tool for identifying streamed videos independent of ABR-protocol. The results provided in this thesis show that tools such as DAITA do not offer total protection against this attack, unlike previously thought. While this thesis did not manage to change the normalization in a way to allow for the dataset to be reduced, an 85% reduction of the dataset was observed in lab conditions where the only representation chosen in the dataset was the highest quality per video (from 9.3GB to 1.4GB as seen in table 4.3).

Since the ABR-leak is more difficult to mitigate than previously thought, attempted large-scale exploitation of this kind of attack may prove to be even more destructive for privacy. If with further research, this kind of database reduction is made realistic, the attack would be significantly cheaper to deploy and scale up for surveillance on larger scales such as ISP-level. While an 85% reduction cannot be assumed as it is possible that videos will need more than one representation, and not all services will provide the same number of representations. Even if this degree of reduction is never reached, deriving 2–3 representations per video could still result in a smaller dataset of up to 70% less size.

# 6

# Conclusion

This thesis began with the aim of further understanding and developing modern private networking solutions, to protect communication and online behaviour from being commodified and used by various parties, such as data brokers, or even private hackers looking for information. As video streaming is a large part of online networking, and a relevant aspect of many people's digital content consumption, it is important to further research what kinds of threats to anonymous browsing exist, and what measures that can be taken to protect from them. This thesis was done in collaboration with Mullvad VPN, that created an algorithm called DAITA that adds a layer of obfuscation on top of their VPN solution. The goal was to test and develop how well DAITA was able to obfuscate video streaming content from the video fingerprinting attack Endangered Privacy, that analysed the bit stream produced by adaptive bitrate protocols to fingerprint and identify the video being streamed. In more detail, the question posed was the following: *How can the attack Endangered Privacy be modified to take into account obfuscation implemented by Mullvad's protocol DAITA, and what is the large-scale applicability of the attack?*

The methodology used in this thesis was to firstly test Endangered Privacy against DAITA and gain a more in-depth understanding of how they perform against one another. Then the work moved on to creating a new attack based on Endangered Privacy in an attempt to improve identification rates of videos streamed with DAITA. This attack in parallel aimed to minimise the memory required to run the attack, by reducing the dataset and making large-scale attacks more feasible.

Testing Endangered Privacy against DAITA on a larger dataset than done previously showed that Endangered Privacy is indeed able to identify DAITA videos at an identification rate of approximately 22%. The results showed that while DAITA does minimise the bursty-ness of video traffic, it does not remove it entirely, which is what allows Endangered Privacy to reach a 22% rate of identification. This bursty-ness was however inconsistent between videos and instances of connection, which made developing an attack more difficult. The attack that was developed was based on a sliding window normalisation, that would offset the added obfuscation from DAITA. Two different methods of normalisation were attempted, neither of which proved to be meaningfully successful. Endangered Privacy was also further tested against DAITA with different parameters retuned, such as threshold value, query size and anti-cluster radius parameters, but identification rates did not improve. In terms of dataset reduction, it was found that the dataset could be reduced up to

85% when all target streams used the highest-quality representation. While this works in a lab simulated environment (quality of the streams can be decided) it is difficult to ensure stream quality in real life, and goes against the reason that ABR is required in the first place.

This thesis provided important discoveries that furthered the understanding between the obscuring properties of DAITA, and the Endangered Privacy attack, as well as provided the framework for a possible new attack in order to further bypass DAITA. However, there is still a wide array of aspects of this attack that must be further researched and developed. As has been seen, there is potential to decrease the size of the database, given that the approximate streaming rate is known to the attacker. Analysing video streams and creating an attack on that basis may lead to a more practical implementation of the database reduction. Furthermore, as the ABR vulnerability (the bursty traffic pattern) is to an extent preserved with DAITA, more analysis on the bursty pattern after DAITA may provide deeper insight into a potential attack vector. Working more in depth with the parameters of the attack (threshold value, query-size etc) could also be something that can be further researched and tweaked that in combination with a more developed ABR attack, could bypass DAITA with more success.

# Bibliography


[1] F. Loh, F. Wamser, F. Poignée, S. Geißler, and T. Hoßfeld, "Youtube dataset on mobile streaming for internet traffic modeling and streaming analysis," *Scientific Data*, vol. 9, no. 1, p. 293, 2022.

[2] B. Gaines. "30 video streaming statistics 2025 (market share trends)," Accessed: Dec. 4, 2025. [Online]. Available: `https://evoca.tv/video-streaming-statistics/`.

[3] C. Kerschbaumer, F. Braun, S. Friedberger, and M. Jürgens, "The state of https adoption on the web," in *Proceedings of the 2025 Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb), San Francisco, CA, USA*, vol. 27, 2025.

[4] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *2005 IEEE Symposium on Security and Privacy (S&P'05)*, IEEE, 2005, pp. 183–195.

[5] M. Björklund and R. Duvignau, "Endangered privacy: Large-scale monitoring of video streaming services," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 6581–6597.

[6] ISO/IEC, *ISO/IEC 23009-1. Information technology - Dynamic adaptive streaming over HTTP (DASH) —*. International Organization for Standardization, Aug. 2022.

[7] J. Gu, J. Wang, Z. Yu, and K. Shen, "Walls have ears: Traffic-based side-channel attack in video streaming," in *IEEE INFOCOM 2018-IEEE conference on computer communications*, IEEE, 2018, pp. 1538–1546.

[8] Mullvad VPN. "Introducing defense against ai-guided traffic analysis (daita)." Mullvad Blog, Accessed: Aug. 9, 2025. [Online]. Available: `https://mullvad.net/en/blog/introducing-defense-against-ai-guided-traffic-analysis-daita`.

[9] T. Pulls and E. Witwer, "Maybenot: A framework for traffic analysis defenses," in *Proceedings of the 22nd Workshop on Privacy in the Electronic Society*, 2023, pp. 75–89.

[10] T. Pulls. "Daita v1 and v2 defenses," Accessed: Aug. 8, 2025. [Online]. Available: `https://pulls.name/blog/2025-03-27-daita-v1-and-v2-defenses/`.

[11] MMs. "Svt toppar med 4,1 miljoner tittare medan netflix går om tv4 och skyshowtime kliver in på topplistan," Accessed: Oct. 13, 2025. [Online]. Available: `https://mms.se/svt-toppar-med-41-miljoner-tittare-medan-netflix-gar-om-tv4-och-skyshowtime-kliver-in-pa-topplistan/`.

[12] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE multimedia*, vol. 18, no. 4, pp. 62–67, 2011.

[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*, 2017, pp. 361–368.

[14] M. Björklund et al., "I see what you're watching on your streaming service: Fast identification of dash encrypted network traces," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, IEEE, 2023, pp. 1116–1122.

[15] T. Pulls. "Evaluating using the first eight daita servers," Accessed: Dec. 11, 2025. [Online]. Available: `https://pulls.name/blog/2024-06-05-eval-first-daita-servers/`.