QoE Degradation Attack in Dynamic Adaptive Streaming over ICN

Original Citation:

Availability:
This version is available at: 11577/3291752 since: 2019-03-14T11:21:48Z

Publisher:
Institute of Electrical and Electronics Engineers Inc.

Published version:
DOI: 10.1109/WoWMoM.2018.8449767

Terms of use:
Open Access
This article is made available under terms and conditions applicable to Open Access Guidelines, as described at http://www.unipd.it/download/file/fid/55401 (Italian only)

(Article begins on next page)
QoE Degradation Attack in Dynamic Adaptive Streaming over ICN

Mauro Conti†, Ralph Droms‡, Muhammad Hassan§, and Sebastiano Valle¶

Department of Mathematics, University of Padua, Padua, Italy †§¶
Google, Cambridge, MA USA §

Email: †conti@math.unipd.it, ‡rdroms@google.com, §hassan@math.unipd.it, ¶sebastiano.valle@studenti.unipd.it

Abstract—The dramatic growth in video demand has motivated content providers (e.g., Netflix and YouTube) to evolve towards Dynamic Adaptive Streaming over HTTP (DASH). However, the rapid growth of multimedia content presents content providers the problem of rapidly expanding the available bandwidth. Therefore, considering the effectiveness of the next generation networking paradigm, i.e., Information Centric Networking (ICN), recently DASH has been introduced to ICN. The inherent features of ICN including efficient content distribution and reduced bandwidth requirements greatly increases the network bandwidth utilization, which smoothly handles the multimedia content delivery. However, we identified that these features of ICN also causes new vulnerabilities in the network.

In this paper, we propose a novel attack that exploits the two fundamental ICN characteristics: in-network caching and interest aggregation. We show that an adversary with limited resources is able to disrupt the adaptive behaviour of the DASH streaming control system to degrade the perceived Quality of Experience (QoE) of a benign user. In particular, the proposed attack that shows the vulnerability in ICN-based multimedia streaming, and forces DASH clients to oscillate with the high frequency and amplitude between various resolutions while streaming. We perform extensive simulations on AMuSt-ndnSIM to evaluate the attack performance, that is an increase in the annoyance factor in spatial dimension and the QoE degradation for the honest user. In addition, we propose possible countermeasures to alleviate these attacks.

I. INTRODUCTION

Currently, video content represents a major component of traffic over the Internet. In fact, video streaming services such as Netflix and YouTube together amount to as much as 50% of Internet traffic [1]. New approaches to video content delivery that cache the video content in the network have been proposed to reduce demand on network bandwidth. Information Centric Networking (ICN) is a networking paradigm that provides support for caching at network layer through unmanaged storage of network content in routers [2], [3]. The routers thus become both forwarding and caching elements. In particular, in ICN the host-centric communication (IP) approach is replaced with a content-centric approach.

One way in which content providers such as Netflix and YouTube have optimized bandwidth utilization is through the use of HTTP-based Dynamic Adaptive Streaming (DASH) [4]. DASH provides a dynamic approach to time-shift control on media requests in response to fluctuating bandwidth conditions experienced by individual users. The goal is to download the highest resolution video through the available network bandwidth, providing the best possible video quality. Since the adaptation decision is based on real-time bandwidth measurement, the DASH client is responsible for optimizing the video quality; however, the client has very limited knowledge about the network conditions [5]. Considering the effectiveness of ICN architecture in reducing bandwidth utilization in delivering shared video content, the research community has investigated the implementation of DASH on ICN [6]–[9]. Several results, including [10]–[14] demonstrate the advantages of the dynamic video delivery of DASH with the receiver-driven content delivery with in-network caching of ICN.

A. Contribution

In this paper, we identify that by adversely exploiting two fundamental ICN features, in-network caching and interest aggregation, an adversary can attack the adaptive behaviour of DASH streaming control system to degrade the user perceived quality of experience (QoE) [15]. Assuming the adversary can access the content through ICN routers and is aware of the multimedia content desired by the honest client, we focus on designing an algorithm that identifies the vulnerability in ICN against dynamic adaptive streaming, which an adversary could use to exploit the in-network caching provided by routers. To this end, the major contributions of this work are as follows:

• We design and implement a new attack model that makes the DASH client compute false bandwidth estimations for bitrate adaptations during a multimedia session. The attack forces the client under attack to download video content of highly variable bitrates, representing very high and low resolutions, while streaming a video file. In particular, the adversary exploits the adaptation mechanism of DASH and fundamental features of ICN routers to launch the attack. We show that this altered behaviour of DASH degrades the user’s QoE by increasing frequency and amplitude of oscillations in the displayed video content.

• We implement the proposed model and validate the results via simulations on AMuSt-ndnSIM [16] taking into account the QoE evaluation metrics [17]. Furthermore, we propose possible countermeasures to alleviate the attack.
The rest of the paper is organized as follows. We begin by discussing dynamic adaptive streaming background and the related work done on the ICN in Section II. Section III describes our system and adversary models. The proposed attack model for QoE degradation is presented in Section IV and evaluated in Section V. Our proposed mitigation approaches are presented in Section VI and finally we conclude in Section VII.

II. BACKGROUND AND RELATED WORK

DAS has become the most widely used adaptive bitrate streaming technique for on-demand, real-time multimedia streaming, and has been defined by ISO/IEC as MPEG-DASH (Dynamic Adaptive Streaming over HTTP). Apple HTTP Live Streaming, Microsoft Smooth Streaming, and Adobe HTTP Dynamic Streaming are related streaming technologies. DASH specifies the description of multimedia content and the way that content is segmented for delivery. In DASH, the media content is encoded in different versions, which vary in bitrates, resolutions, codecs, and so on. These versions are further decomposed into segments of specific durations. The DASH client requests individual segments using HTTP GET requests [10]. The DASH client adapts to varying network conditions by requesting successive segments with different characteristics. There are two primary adaptation strategies: (i) Rate Based (RB) and (ii) Buffer Based (BB). RB strategies use estimates of available network to select segments to be downloaded, as in Probe and Adapt (PANDA) [18]. BB strategies use available buffer space to select segments, as in Buffer Occupancy based Lyapunov Algorithm (BOLA) [19], [20]. Elements of RB and BB strategies can be combined into Rate and Buffer based (R&B) strategies [21].

The research community has been exploring the implementation of DAS strategies for delivery of video content over ICN1 Numerous authors in [7], [8], [10]–[13] have evaluated the data-oriented delivery and in network caching offered by ICN as a support for DAS. For instance, the authors in [14] shows an integration of DASH and ICN by enabling a proxy service between HTTP and ICN. Moreover, authors in [10], [13] show the implementation of DASH client as a native ICN interface by transforming the HTTP request and reply messages to corresponding interest and content messages. Figure 1 presents a proposed architecture for DASH over ICN [10], where DASH-related components are marked in light blue and ICN-related components in dark grey color.

A Media Presentation Description (MPD) defines the relationship between a segment’s associated characteristics (e.g., bitrate, resolution, codec, timeline) and its name. As used in DASH over ICN, each MPD lists the ICN names (i.e., Uniform Resource Identifier) of the media segments instead of URLs [6]. The versioning of segments in ICN indicates different representations of DASH-based multimedia content. Hence, a client requests the content appropriate to the estimation of the available bandwidth and network conditions [23]. In particular, the ICN interest messages are issued to retrieve video segments and in return the video segments may be provided by the original content source or returned from in-network caches.

The role of the DASH streaming control mechanism is to adapt the client requests based on the available bitrate and network bandwidth to provide a smooth streaming session with high Quality of Experience (QoE). The work in [10], [11] shows that DASH over ICN is able to compete with the existing HTTP streaming system in terms of average download bitrate. Moreover, it is able to provide smooth streaming with reduced bandwidth requirements for the origin server. The authors in [13] also demonstrates the usefulness of in-network caching in the presence of multiple clients fetching the same content, and resulting in increased video quality over time. Furthermore, the implementation of ICN-based dynamic adaptive streaming exhibits advantage while using Scalable Video Coding (SVC) [12], showing that layered approach increases the efficacy of adaptation process and guarantees a smooth playback without stalling.

In this paper, we observe the behaviour of DASH streaming control system while interacting with ICN’s implicit characteristics. The effective support of in-network caching for popular content and native multicast capability in ICN greatly reduces the traffic burden for the content providers. However, these features also create new opportunities for the malicious users to degrade the QoE of a DASH client during its streaming process.

III. SYSTEM AND ADVERSARY MODEL

In this paper, we consider the scenario of dynamic adaptive streaming in ICN as illustrated in Figure 2. The content provider server called producer (P) contains the multimedia data (S) in a DASH-compatible format. S is divided into a collection of n segments of equal length, where each segment is encoded and stored at several media encoded resolutions (b) representing different viewing bitrates. A client, (C), requests the segment(s) from P in one of the available bitrates of S.

In our scenario, the adversary (Adv) is aware of the media content being requested by C in advance. Each interest and the corresponding content from C and Adv traverses one or

---

1Among numerous ICN styles, Named Data Networking (NDN) [22] and Content-Centric Networking (CCN) [2] are projects which have gained considerable attention in research groups of both academia and industry.

Fig. 1: Dynamic adaptive multimedia streaming over ICN
more routers before being satisfied by $P$ or one of the ICN routers, $R$. The goal of $Adv$ is to degrade the QoE perceived by $C$ while streaming the media content. In our scenario, every intermediate router operates according to ICN defaults [22]. Moreover, the forwarding strategy adopted is bestRoute [7], which routes packets according to the lowest path cost.

A. System Model

In our scenario all the entities, $C$, $Adv$, $P$ and $R$ implement the ICN stack. For multimedia streaming in ICN, $C$ and $P$ use the aforementioned DASH over ICN model [10], [13]. $Adv$ is aware of DASH but will make requests for video content to degrade the QoE of the content streamed by $C$, rather than trying to optimize its own content delivery. We use two different coding formats for DASH-compliant multimedia content: Advanced Video Coding (AVC) [24] and Scalable Video Coding (SVC) [25]. In AVC, each segment of different bitrates is represented by a unique segment name. For example, the first segment of the 100 kbps representation is referenced as /itec1/dash/bunny/bunny2s000kbit/bunny2s1.m4s. For SVC, the multimedia content is encoded in different independent layers of quality called the base layer (BL) and the enhancement layers (EL), where each layer subsequently enhances the video quality. The segments are referenced by the MPD, in which they are listed by their URIs [6]. $C$ requests the segments according to the DASH streaming strategy, which implements the adaptive performance. We investigate the behaviour of $C$ in the presence of an $Adv$ while using all type of adaptation strategies that are referenced as standard in DASH streaming control system, i.e., Rate-Based (RB), Buffer-Based (BB), and Rate-Buffer-based (R&B) [16]. Below we describe the functionality of these DAS adaptation logic.

1) Rate-Based adaptation logic: The RB adaptation algorithms [18], [26] use the previous segment’s measured bandwidth as an estimate of the bandwidth available for the next segment, and then select the appropriate segment based on that bandwidth estimate. $C$ estimates the future available bandwidth for the next segment using an exponential moving average, which is given in Equation 1,

$$
\lambda_{k+1} = (1 - \beta) * \lambda_k + \beta * \lambda_t
$$

where $\lambda_{k+1}$ denotes the new estimate for bandwidth and $\lambda_k$ denotes the previous estimate. $\lambda$ denote currently measured bandwidth and it is calculated by taking the ratio of the current segment size to the download time of that segment. $\beta$ is a constant which controls the effect of more recent measurements on the bandwidth estimate. $C$ uses the bandwidth estimate to select the highest possible encoded bitrate $b_{k+1}$ as a requested bitrate, i.e., $b_{k+1} < \lambda_{k+1}$ for the next segment.

2) Buffer based adaptation logic: The BB adaptation logic function is independent of bandwidth estimation. BB selects the video quality according to the current buffer occupancy $B(t)$. Categorically, the buffer is divided into multiple levels and $C$ requests the $b_{k+1}$ according to its actual buffer level. We use Bandwidth independent Efficient Buffering (BiEB) [20] as a standard in our model with a maximum buffer limit of 33 seconds.

3) Rate and Buffer-Based Adaptation logic: The R&B adaptation algorithm [21] uses the RB strategy of bandwidth estimation for the next segment $\lambda_{k+1}$, while stabilizing the buffer level $B(t)$ around a target value, $B_{max}$. This strategy keeps the adaptation process as smooth as possible by avoiding reaction to short-term bandwidth spikes and stalling. In particular, the algorithm functions by the use of two threshold values ($B_{min}$ and $B_{max}$) along with $\lambda_k$ and $\lambda_k + 1$. The increase/decrease of the video quality is governed in two ways: the decrease, when $B(t) > B_{max}$ then algorithm keeps the current $b$. When the buffer lever is $B_{min} \leq B(t) \leq B_{max}$, it quickly shifts to a lower bitrate encoding. Furthermore, the lowest bitrate encoding is requested when $B(t) < B_{min}$. Conversely, when available bandwidth increase, if the buffer level is $B_{min} \leq B(t) \leq B_{max}$ or greater than $B_{max}$ the quality is increased with respect to estimated bandwidth.

B. Adversary model

In our analysis, we assume $Adv$ is connected to the same first-hop router as $C$, and is able to fetch content from $S$. By using geo-locating techniques [27], we could relax the first assumption and require only that $Adv$ just needs to be connected to a nearby router. Using these existing techniques $Adv$ can identify the router closest to the consumer.

We assume that $Adv$ has prior knowledge of $S$ that $C$ is going to request in near future. Several existing techniques supports this assumption apart from the preliminary knowledge required to execute the attack subjective to $C$. $Adv$ can exploit timing attacks as a side channel to breach privacy and infer if that content has been previously requested by $C$ [28]. Moreover, $Adv$ could also probe for the MPD with a timing attack, hence discovering whether $C$ has previously requested it. These techniques allows $Adv$ to predict the video in which $C$ was or is currently interested in.

Furthermore, we assume that $C$ may be share the same wireless link, so the traffic traces are exposed and may be
easily eavesdropped [29]. The Adv could infer the online activities of a user by analyzing the traffic and then be able to predict the content and its source [30]. We assume that multimedia content is publicly available on the Internet, and Adv can access S by means of generic Internet Service Providers (ISPs).

IV. A Bitrate Oscillation Attack for DAS over ICN

In this section, we present details of our proposed attack, which degrades the user perceived QoE while streaming multimedia content in DASH over ICN. In order to degrade the QoE for C during the streaming session, Adv creates oscillations in the adaptive behaviour of the DASH streaming control system. The Adv implements the attack by forcing the DASH control to dynamically switch between high and low representations frequently (e.g., b+ and b−), as shown in Figure 2.

In summary, suppose C wants to stream a video file, S. To trigger oscillations, Adv requests selected segments of S in advance of requests from C. The segments returned from P through the network to Adv are stored on each intermediate router that earlier forwarded the corresponding interest [2]. When C subsequently requests the segments of S in sequence, some of those segments are returned from P and some from the intermediate routers. The difference in the times required to deliver the segments to C will cause oscillation in the quality of the segments requested by C, degrading the QoE of the delivered video stream.

We assume that Adv has knowledge of the video stream S that will be requested by C. However, Adv is not aware of the specific bitrates of the segments of S that C will request. Initially, Adv receives the MPD containing the list of the all available segments of S in their various encoded representations (bitrates) [13]. As Adv implements its attack, it uses the list of segments in MPD to request a non-sequential subset of S in such a way which leads to higher oscillations for C. For each selected segment of S, Adv requests all available representations of that segment to ensure that any version of that segment requested by C will be cached in an intermediate router.

Recall that, as the interest traverses through each router which is on the path from Adv to P, the routers create a state in the form of Pending Interest Table (PIT) entry to satisfy the requirements of interest aggregation [22]. After receiving the interest, P injects the requested content back into the network, which follows the same route from which the interest is received.

In our model, illustrated in Figure, Adv requests the segments of S in ascending order, skipping a number of segments between each requested segment. As shown in Figure 3, when C subsequently requests a segment of S, either Adv has previously requested that segment or the segment was one of the segments skipped by Adv and, therefore, not requested. Consider a specific segment, $S_n$, requested by C. If Adv has previously requested $S_n$ and it has been returned from P, a copy of $S_n$ will be available in the cache of $R_2$ and will be returned to C, in the round-trip time between C and $R_2$. If Adv has previously requested $S_n$ but the segment has not yet arrived at $R_2$, the interest from C will be aggregated at $R_2$ and $S_n$ will be returned to C in less than the full round-trip time between C and P, depending on where $S_n$ is on the path from P to Adv. If $S_n$ has not previously been requested by Adv, the request will be forwarded to P and $S_n$ will be delivered in the round trip time between C and P.

When subject to this attack, C interprets the relatively short delivery time for segments pre-fetched by Adv as indicating high available bandwidth in the network, while C interprets the longer delivery times for other segments as indicating low available bandwidth. By pre-fetching segments of S with the consecutive gaps, Adv causes the DASH adaptation strategy at C to frequently switch between different bitrates, such as very low and high.

To better illustrate the attack and its corresponding behavior at C, let’s consider the topology in Figure 3, where two DASH-enabled clients Adv and C are connected to $R_2$ via $AP_1$ and $AP_2$. The $R_2$ has a cache $Cs_2$ that can hold the multimedia content S originating from P. Furthermore, S is available in single-layer coding formats, such as AVC. P provides multiple representations of each segment, here each representation features to a set of representations, $Sn(b_i...j)$. The representations are listed in the MPD, using the ICN naming scheme for the segments [10]. Following the adaptive streaming control, C requests a representation that best matches with its current estimate of available bandwidth and buffer utilization. Adv and C are streaming the content S from

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv</td>
<td>adversary</td>
</tr>
<tr>
<td>C</td>
<td>client</td>
</tr>
<tr>
<td>P</td>
<td>server providing video file at S</td>
</tr>
<tr>
<td>S</td>
<td>video file at P</td>
</tr>
<tr>
<td>$S_n$</td>
<td>$n^{th}$ segment of S</td>
</tr>
<tr>
<td>$R_i$</td>
<td>routers</td>
</tr>
<tr>
<td>$AP_i$</td>
<td>access point</td>
</tr>
<tr>
<td>N</td>
<td>total number of video segments</td>
</tr>
<tr>
<td>$Cs_{rt}$</td>
<td>cache on $R_i$</td>
</tr>
<tr>
<td>$b_{(i,j)}$</td>
<td>set of available bit rates of S</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>consecutive gap of variable length</td>
</tr>
<tr>
<td>$b_f$</td>
<td>bitrate of S received from $Cs$</td>
</tr>
<tr>
<td>$b_j$</td>
<td>maximum bitrate</td>
</tr>
<tr>
<td>$b_i$</td>
<td>minimum bitrate</td>
</tr>
</tbody>
</table>

Table I:

Summary of notations used

**Fig. 3:** Topology considered
P. Adv launche the attack at time $t_i$, and C starts streaming at time $t_j$, where $t_j > t_i$. Now we present the design and configuration of our attack Algorithm 1 at Adv using which it requests content in an ascending order with a predefined consecutive gap ($\alpha$). We then investigate the behavior of C in the presence of our algorithm running at Adv.

After retrieving the MPD at time $t_i$, Adv issues a series of interests, $S(n+\alpha)$, where $\alpha$ is the consecutive gap of variable or fixed length to issue discontinuous requests toward P with all the available bitrates ($b_{(i,j)}$) of S. These interests traverse through the path, $Adv \rightarrow AP_1 \rightarrow R_2 \rightarrow R_1 \rightarrow R_0$. P returns the requested segments, which follow the reverse path and are cached at intermediate R. At time $t_j$, C retrieves the MPD for the desired content. By parsing MPD, C sends interest messages to fetch $S_n$ with an appropriate bitrate, say $b_k$, which is estimated on current network conditions [4]. During the streaming session, when C request an interest $S(n+\alpha)$, the path this specific interest will traverse is $C \rightarrow AP_2 \rightarrow R_2$ since $R_2$ replies the content from $C_{S_2}$. Let $B_j$ be the bitrate representation of the segment received from $C_{S_2}$, $b_{f+1}$ be the next, and $b_i$, $b_j$ be the available lowest and maximum bitrates. For the interest $S(n+\alpha)$, C finds the download rate higher than the previous segment. Due to this, the DASH adaptation logic at C believes that the available bandwidth is suitable for requesting maximum representation for next segment, and switches to a higher bitrate representation, such as $b_j$. Therefore, C requests $S(n+\alpha+1)$ with bitrate $b_j$. Because Adv has not previously requested that segment and it has not been cached at $R_2$, the interest is forwarded to P, through the path $C \rightarrow AP_1 \rightarrow R_2 \rightarrow R_1 \rightarrow R_0$. Consequently, upon reception of $S(n+\alpha+1)$ from P, C again computes the available bandwidth for the next segment, and estimates a lower throughput due to increased round trip time (RTT) of $S(n+\alpha+1)$. In turn, C switches to a lower representation for the subsequent segment. Subsequently, the process will be repeated, hence causing the requests from C to rapidly result in a number of cache hits and misses.

Algorithm 1 Attack for Adversary (Adv)

1: procedure SEQUENCE_OF_INTEREST ($S_{i,j,\alpha}$)
2: MPD $\leftarrow$ Send requests to P $\triangleright$ MPD = \{S(n)_{b_{i,j}}\}
3: for $n = 1$ to $\leq N$ do $n + \alpha$
4: Content(S(n)$_{b_{i}}$) $\leftarrow$ Interest(S(n)$_{b_{i}}$)
5: for $f \in b_{(i,j)}$ do
6: Content(S(n)$_{b_{j}}$) $\leftarrow$ Interest(S(n)$_{b_{j}}$)
7: $\triangleright$ S(n)$_{b_{i}}$, ..., S(n)$_{b_{j}}$ caches on C$_{Sk}$
8: end for
9: end for
10: close;

Algorithms 2 and 3 illustrate the process of C to stream $S(n)_{b_{i,j}}$ adaptively for segment and bitrate selection, respectively after the attack. As a result of these algorithms, C will experience undesirable bitrate oscillations, manifesting as continuous switches between high and low representations, leading to degradation in QoE. In addition, the playback buffer depletes in case of repeated oscillations and forces C to take radical measures to refill it at the expense of smooth streaming.

V. EVALUATION METHOD AND MEASUREMENT SETUP

To evaluate the attack and investigate the performance, we used AMuSt-ndnSIM, Adaptive Multimedia Streaming Framework for ndnSIM [16]. AMuSt-ndnSIM simulates the transport of video content over NDN [22], using ndnSIM and libdash [31]. Libdash is an official reference software for DASH standard. Note that NDN is a specific instantiation of ICN which is well-suited for this evaluation.

To set up the tests, we implement the network depicted in Figure 3 with a single origin server and video clients (C and Adv) connected by multiple NDN routers. As test content, we use an AVC-encoded multimedia video [24] and the BigBuckBunny movie from the DASH/SVC Dataset [25]. The parameters and their values used in test setup are detailed in Table II. The forwarding strategy selected is minimal hop
count (BestRoute) and we chose LRU Least Recently Used as a caching policy for router caches.

A. Evaluation Metrics

The authors in [15], [17], [32] describe the impairment factors that affect QoE for DASH video. One of those factors is oscillation in the displayed video resolution, which is considered to degrade QoE. To evaluate the bitrate oscillation attack, we used two metrics that affect QoE:

- Number of switches, which indicates the frequency of video quality switches [15];
- Average switch magnitude, which indicates the average amplitude of the video quality switches [15], [17].

B. Attack Effectiveness

The simulation results demonstrate that an adversary is able to manipulate the behaviour of a DASH client, causing it to miscalculate the available bandwidth and, therefore, choose to request and display segments of different resolution even though the underlying network conditions are unchanged. The DASH client frequently switches between low and high quality resolutions, as shown in the simulation results below. These frequent quality fluctuations in video streaming degrade the QoE, since the quality perceived is severely affected by the increase in the amplitude and frequency of the bitrate switching [15].

Figures 4 and 5 report the bitrate requested by the DASH client for both the cases (with and without an adversary) using RB and R&B adaptation logic. Our results in figures 7 and 8 shows that frequency of bitrate switching in RB and R&B adaptation logic increases markedly in the presence of an adversary. Moreover, the attack significantly increases the average switch magnitude of bitrate fluctuations for these adaptation logics while streaming AVC content, as shown in Figure 14. The results shown in Figure 6 shows the bitrate requests in BB adaptation logic (AVC) and Figure 9 shows that the DASH client experiences an increase in the number of switches. Figure 14 shows that the client experiences an increase in average switch magnitude of bitrate fluctuations in the presence of the attacker for all three examples of the adaptation logic.

For the RB adaptation logic in the SVC dataset, the attack results in higher frequency of switches in the download bitrate, as shown in Figures 10 and 12). However, the magnitude of the switches is relatively small when compared to what we found for AVC dataset as shown in Figure 14. This is because the number of available layers of representations (i.e., three EL and one BL) is low as compared to the twenty representations available in AVC. Regardless, the adversary is able to cause a higher number of bitrate switches in SVC dataset with respect to normal conditions, leading to a QoE degradation. From the simulations, we also observe that buffer-based adaptation logic in SVC dataset is unaffected by the attack. Figures 11 and 13 show that there is no increase in bitrate fluctuations and average switch magnitudes. We identify that buffer capacity affects positively and resists short-term bandwidth fluctuations. However, use of the buffer based adaptation logic with resource constraint devices [33] still remains an open question for researchers due to buffer size management in relation to the playback time, since in large networks multimedia delivery imposes dramatic burden on in-network caching.

VI. COUNTERMEASURES AGAINST THE QOE DEGRADATION ATTACK

In this section, we discuss several strategies to defend against the QoE degradation attack. One trivial strategy to counter the attack is to disable the router-based in-network caching provided by ICN. However, this would defeat the performance advantage in-network caching associated with in-network caching, which is one of the key features of ICN.

Another simple approach to defeat the bitrate oscillation attack would be to exploit the method of content-specific delay described by Acs, Conti, et al. [28]. The bitrate oscillations could be reduced if a router hides the cache hits by introducing artificial delays before responding with cached content.

To better illustrate the proposed approach, for each multimedia segment $S_n$, let $R$ store the original interest-in/ content-out delay ($\gamma_{S_n}$). In fact, $\gamma_{S_n}$ is the time it took $R$ to obtain $S_n$ from either the origin producer or some other router’s cache. If an interest later arrives on $R$ for Segment $S_n$, which is in $R$’s cache, $R$ delays responding by time $\gamma_{S_n}$. This procedure is shown in Algorithm 4.

**Algorithm 4 Privacy-Aware-Forwarding**

1: procedure PRIVACY_AWARE_FORWARD($S_n, C_s, \gamma_{S_n}$)
2:     if $S_n \notin C_s$ then
3:         forward$_{S_n, downstream} \leftarrow$ Local forwarding strategy
4:     else
5:         Forward$_{S_n, downstream} \leftarrow$ Add_delay$_{\gamma_{S_n}}$
6:     end if;
7: close;

With this strategy, even if the adversary is able to cause segments of video content to be cached along the path of a DASH client, that cached content will arrive with an effective delivery time that does not result in an invalid throughput estimation by the client. However, this strategy adversely affects the overall end-to-end latency, and partially defeats
some of the advantages of ICN. Also, considering the interest aggregation support of ICN routers, this strategy may not mitigate all attacks.

Another strategy to mitigate the attack and to have smooth
bitrate adaptation is a receiver-driven adaptive interest rate control mechanism [34]. The approach requires no additional support from the network, as is required by the previous artificial-delay strategy. The client controls the interest sending rate to reduce unnecessary bitrate oscillations. In the proposed method, the client calculates the RTT for each interest packet based on the time the corresponding content is received. In addition, over a series of short time intervals \( T \), the average RTT \( RTT_{avg} \) is calculated dynamically. The client adjusts the rate at which it sends interests by comparing the current RTT value of interest packet with \( RTT_{avg} \). In particular, if the current RTT value is smaller with respect to \( RTT_{avg} \), the client decreases the interest sending rate, as shown in equations 2 and 3 respectively.

\[
RTT' + \text{jitter} \ll RTT_{avg} \tag{2}
\]

\[
Ips_{now} \leftarrow Ips_{prev} - \kappa/\sqrt{Ips_{prev}} \tag{3}
\]

Here, \( Ips \) denotes the number of interest packet sent by the client per second, \( RTT' \) is the latest RTT, and \( \kappa (0 < \kappa < 1) \) is the multiplicative decrease factor to reduce \( Ips \) in case of attack, and to maintain smoothness in streaming. Similarly for consecutive \( RTT_{avg} \) decrease, interest sending rate increases, as illustrated in Equation 4. Here \( \omega (\omega \geq 1) \) is the additive increase factor to adjust adaptability to increase bandwidth estimation. Therefore, in case of a rapid decrease in the RTT of a interest due to cache hit (i.e., pre-loaded by the adversary), the adaptation logic unit behaves normally and selects constant bitrate on the basis of average RTT for the specific time period. We believe RTT base bandwidth estimations are helpful to provide smooth bitrate adaptation, and to reduce bitrate oscillations triggered by our proposed attack. However, further research is required to investigate the behavior of this method in case of multipath sources and network congestion in parallel.

VII. Conclusion

In this paper, we have provided an example of how exploitation of in-network state can lead to privacy risks and performance attacks in ICN. We designed an algorithm that allows an adversary to degrade the performance of a client streaming multimedia content in a generic ICN architecture. We validated our proposed attack via simulations on AMuSt-nnSIM. The results of our simulations showed significantly increased frequency of bitrate switching and increased amplitude of oscillations, both of which degrade the QoE at the content consumer. We then discussed possible countermeasures, showing that artificial delay introduced by routers when requested content is found in the local cache can decrease the effects of the bitrate oscillation attack. We also described a mitigation strategy that uses bandwidth estimation over fixed time intervals. We showed that this strategy, which has the advantage of being implemented entirely in the client, can be effective in significantly decreasing the effect of the bitrate oscillation attack.

REFERENCES


