Fair-RTT-DAS: A Robust and Efficient Dynamic Adaptive Streaming over ICN

Mauro Conti[†], Ralph Droms[‡], Muhammad Hassan[¶], and Chhagan Lal[§]
Department of Mathematics, University of Padua, Padua, Italy [†¶]
Google, Cambridge, MA USA [‡]
University of Padua, Italy, and Manipal University Jaipur, Rajasthan, India[§]
Email: [†]conti@math.unipd.it, [‡] rdroms@google.com, [¶]hassan@math.unipd.it, [§]chhagan@math.unipd.it, [¶§]Corresponding Author

***Abstract*—To sustain the adequate bandwidth demands over rapidly growing multimedia traffic and considering the effectiveness of Information-Centric Networking (ICN), recently, HTTP based Dynamic Adaptive Streaming (DASH) has been introduced over ICN, which significantly increases the network bandwidth utilisation. However, we identified that the inherent features of ICN also causes new vulnerabilities in the network. In this paper, we first propose a novel attack called as Bitrate Oscillation Attack (BOA), which exploits fundamental ICN characteristics: in-network caching and interest aggregation, to disrupt DASH functionality. In particular, the proposed attack forces the bitrate and resolution of video received by the attacked client to oscillate with high frequency and high amplitude during the streaming process. To detect and mitigate BOA, we design and implement a reactive countermeasure called *Fair-RTT-DAS*. Our solution ensures efficient bandwidth utilization and improves the user perceived Quality of Experience (QoE) in the presence of varying content source locations. For this purpose, Fair-RTT-DAS consider DASH's two significant features: round-trip-time (RTT) and throughput fairness. In the presence of BOA in a network, our simulation results show an increase in the annoyance factor in user's spatial dimension, i.e., increase in oscillation frequency and amplitude. The results also show that our countermeasure significantly alleviates these adverse effects and makes dynamic adaptive streaming friendly to ICN's implicit features.**

***Index Terms*—Information centric networking; Quality of Experience; Video Streaming; Oscillation attack; DASH; RTT fairness.**

I. Introduction

The mobile video preponderance is certain in future Internet traffic trends such as according to Cisco VNI estimation, video data will consume more than 80% of the IP traffic, and the wireless mobile devices will generate two third of the Internet traffic [1]. In fact, the video streaming services such as Netflix and YouTube together amount nearly 50% of Internet traffic. Similarly, other services such as Hulu, Amazon and HBO GO are gaining popularity as well. Due to this rapid increase in multimedia traffic over Internet, the network operators are facing challenges in meeting the bandwidth requirements of the end users. As a result, new networking paradigms have been proposed to cache the traffic within the network and ease the bandwidth blockage. Information Centric Networking (ICN) is an emerging networking paradigm with an inherent support for caching at the network layer. In ICN, the routers perform forwarding as well as caching of data packets. In particular, ICN replaces the host-centric communication approach with a content-centric approach [2], [3].

One approach that content providers are using to optimize the bandwidth usage for efficient multimedia delivery is through Dynamic Adaptive Streaming over HTTP (DASH) [4]. DASH provides a dynamic approach to time-shift control on media requests in response to fluctuating bandwidth conditions experienced by individual users. In particular, it strives to adopt the most appropriate resolution of each video in varying network conditions to attain the best possible quality, hence it is becoming a standard for multimedia streaming applications. In DASH, the adaptation decision is based on real-time bandwidth measurement and it moves the accountability of quality assurance towards the consumer, who might have very limited knowledge about the network conditions [5]. Considering the effectiveness of ICN architecture and to overcome the multimedia streaming limitations, recently DASH has been implemented in ICN. The research community emphasize the explicit ICN characteristics as a support to DASH [6]–[9]. Also, numerous outcomes [10]–[14] evidently highlights the efficacy of this combination, which amalgamate the performance of DASH with the receiver-driven chunk-based content delivery of ICN.

A. Motivation and Contribution

ICN in-network caching feature is beneficial for the content provider in terms of lower transmission delay and reduced bandwidth. However, we show that it makes Dynamic Adaptive Streaming (DAS) to be more challenging in ICN by exposing it to new security risks. In case of ICN, we show that it is inadequate to deliberate fair throughput estimation for DAS. This is because DAS relates to the style of TCP/IP and it was initially designed for host-centric networking model.

Driven by the importance of addressing security issues at the initial stages of a potential new Internet architecture (i.e., ICN), we identify that an attacker can adversely exploit two fundamental ICN features, namely *in-network caching* and *interest aggregation*. In particular, the adversary is able to harm the adaptive behaviour of DASH streaming control system, which leads to the degradation of user perceived QoE. To successfully perform the attack, we assume that the adversary has access to the content through ICN routers, and

she is also aware of the multimedia content desired by the benign client. Using our proposed algorithm, an adversary could exploit the implicit characteristics of ICN routers to launch a Bitrate Oscillation Attack (BOA) for DASH over ICN.

To the best of our knowledge, we are the first to propose this type of attack and its countermeasure. We believe that our proposed attack, which is supported by a comprehensive investigation is essential before ICN can be considered adequate for DAS, and it is deployed for real-world multimedia applications. The initial results depicting the impact of our proposed attack on QoE perceived by the end-users is reported in [15]. In this paper, we further extend the attack scenarios and analysis, and we propose a receiver driven approach called *Fair-RTT-DAS* to countermeasure the attack. Fair-RTT-DAS uses a scalable adaptive rate control technique to enhance user perceived QoE in the presence of an adversary and ICN vital features. Fair-RTT-DAS maintains fairness[1] in the face of uneven round trip time (RTT) values caused by ICN dynamic in-network caching and implicit multicast support features. We show that utilizing Fair-RTT-DAS, the DASH client which is experiencing higher throughput variations due to varied content source locations or due to BOA, it will be able to switch with better resolutions leading to improved QoE. To this end, the major contributions of this work are as follows.

- We design and implement a new attack called *Bitrate Oscillation Attack* (BOA). The attack forces the honest client(s) to stream continuously in immensely different available bitrates (i.e., 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 leads to increase in the annoyance factor for the user's spatial dimension.
- We propose and implement an effective countermeasure called *Fair-RTT-DAS*. It is a client-driven approach for DAS over ICN to resist the BOA in presence of both, the adversary and the inherent content source variations of ICN.
- We fully implement the proposed attack and the countermeasure. Our simulation results show the adverse impact of the attack and the effectiveness of our countermeasure. The evaluation and validation are done using the results obtained on AMuSt-ndnSIM [16] simulator. The results in terms of QoE evaluation metrics [17] clearly shows the feasibility and impact of our work.

B. Organization

The rest of the paper is organized as follows. We present an overview of ICN in Section II. The brief background information about dynamic adaptive streaming and related work done on the DASH over ICN is discussed in Section III. Section IV describes our system and adversary models. Our proposed attack and its mitigation algorithms along with the functional details have been presented in sections V and VI. The result evaluation and analysis that shows the effectiveness and feasibility of our work are presented in Section VII. Finally, we conclude in Section VIII.

[1]Unlike TCP/IP where fairness implies equal resource allocation to multiple flows, we use the term *fairness* to depict optimization of throughput in a single flow by smoothing the RTT of received data packets.

II. ICN Overview

The distinct features of ICN architecture such as receiver-driven mechanism, in-network caching, inherent support for mobility, and multi-cast routing makes it a perfect candidate to fit in the design space of receiver-driven multimedia streaming systems. Several recently proposed works exposed the importance of ICN as a valuable alternative to TCP/IP in order to advance the competence of the current multimedia streaming systems (please refer to Section III-A for more details). In this section, we briefly review the ICN features in the light of DAS necessities and significant advantages of ICN adaptation in DAS multimedia delivery. It is out of the scope of our work to provide a comprehensive survey on ICN, hence we refer the interested readers to [18]. Among numerous ICN styles, Name Data Networking (NDN) [19] and Content-Centric Networking (CCN) [2] projects have gained considerable attention in research groups of both academia and industry. Besides of being prominent, the architectures of NDN and CCN are very similar, moreover, their variances do not affect the consideration and explanation of the paper. Therefore, although our experimental evaluation is based on the NDN Forwarding Daemon, to avoid any ambiguities, we refer to the reference architectures comprehensively as ICN in the rest of our paper.

In contrast to IP where content is explicitly exchanged among nodes, ICN clients directly request content pieces from the network without addressing content provider's location. Specifically, when an ICN client wants to consume a specific *content*, it sends out a unique *interest* associated to that content. Each interest is identified by a Uniform Resource Identifier (URI) with a routable name scheme, e.g., `/example.com/video4u/examples.mp3` [19]. Unlike IP, where security is provided by the upper layers, ICN comes with security in mind. In ICN, each piece of content is shared along with a signature key that can be used to verify the integrity of the received content. Specifically, security in ICN follows a data-centric model. Each content is signed by the content provider, allowing interest senders to verify its integrity and data-origin authentication [20]. Open source implementation of NDN and CCNx gives more responsibilities to the routers as it introduces router-side content caching and interest aggregation [2]. Upon receipt of an interest for a content, ICN router first check whether a requested content is already present in the *cache* (i.e., Content Store). If the content is not found in the cache, router look in a *Pending Interest Table* (PIT) for a pending interest issued for the same content. The router forwards the interest towards its destination, in case, a PIT miss occurs. However, if there is a match in the PIT, further interests issued for the same name are not forwarded,

instead collapsed in the PIT. Later, when the requested content arrives at the router, all the pending interests for it are satisfied just by sending the content back to all the hosts who issued those interests. In this way, ICN provides explicit support of multicast data routing, which indeed means huge benefit for receiver-driven multimedia delivery. The router's *Forwarding Information Base* (FIB) is responsible for forwarding interests towards the content provider via one or more network interfaces (*faces*) based on the routes to the origin node(s). The requested data packet is then forwarded towards the sender by simply traversing, in reverse, the path of the preceding interest [19].

III. Related Work

In recent years, DAS has become the most used adaptive bitrate streaming technique that supports on-demand and real-time multimedia streaming. Therefore, most of the Internet video streaming providers such as Netflix, Amazon and Sky rely on DAS [7]. Along with solutions like Apple HTTP Live Streaming, Microsoft Smooth Streaming, and Adobes HTTP Dynamic Streaming which were highly appreciated, MPEG-DASH (Dynamic Adaptive Streaming over HTTP) is ratified by ISO/IEC and it became the utmost used standard for DAS. DASH specifies the description of multimedia content availability and the process of how it shall be segmented.

Figure 1 shows the idea of adaptive multimedia streaming over HTTP. The figure indicates that media content is encoded in different versions, and it provides variability in bitrates, resolutions, codecs, and so on. These versions are further sliced into segments of specific lengths, and the client adopts a pull-based approach to request each segment individually using HTTP GET requests [10]. In particular, the multimedia content on the HTTP server entails two distinct elements, namely: segments and Media Presentation Description (MPD). The relationship between a segment's associated characteristics (e.g., bitrate, resolution, codec, timeline) and the location are provided by the so-called XML based MPD, where HTTP URL represents an individual segment. Initially, the MPD file is retrieved by DASH client. Thus, using the information in MPD file, the DASH client requests the most appropriate bitrate by considering the user's current context, i.e., bandwidth fluctuations, preferences, etc. [24]. As a consequence, the streaming system is pull-based, and the entire streaming logic is located on the client, which makes it scalable, and possible to adapt the media stream to the client's capabilities.

This dynamic adaptation of bitrate is based on different DAS strategies, which are mainly classified in two immense families: (i) Rate Based (RB), and (ii) Buffer Based (BB), referring that bitrate adaption processes either by estimating the throughput level or the buffer level. There are several representatives of these strategies. For instance, Probe and Adapt (PANDA) [21] and Buffer Occupancy based Lyapunov Algorithm (BOLA) [22], [23] are very popular, and mostly characterized as the benchmark for RB and BB in the literature. Despite this generalized categorization of DAS strategies,

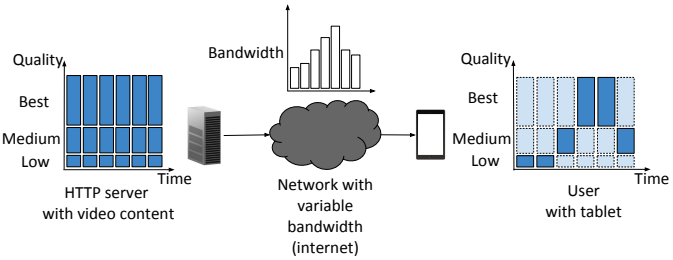

Fig. 1: Dynamic adaptive multimedia streaming principle

both metrics, i.e., throughput and buffer level, are often collaboratively used together in order to attain an improved adaptation process, named as Rate and Buffer based (R&B).

A. Dynamic Adaptive Streaming over ICN

With the rise of ICN [2], [19] as a new communication paradigm for the future Internet and the popularity of adaptive multimedia streaming over HTTP [24], DASH over ICN has gained significant attention from the research community. Numerous works [7], [8], [10]–[13] have considered in-network functionalities 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. Authors in [10], [13] fully exploit the potential of ICN and shows the implementation of DASH client as a native ICN interface. In particular, the framework transforms the HTTP request and reply messages to corresponding interest and content messages. Figure 2 presents the proposed architecture of DASH over ICN [2], where DASH-related components are marked in light blue and ICN-related components in dark grey colour.

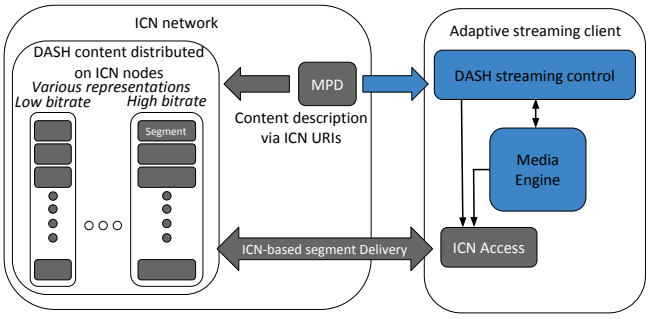

Fig. 2: Dynamic adaptive multimedia streaming over ICN

In general, similar to HTTP-based multimedia streaming, the MPD defines the relationship between a segment's associated characteristics 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 ICN naming scheme in DASH over ICN supports versioning and segmentation, which is necessary for multimedia streaming in ICN [10]. The versioning of segments in ICN indicate

different representations of DASH-based multimedia content. Figure 3 illustrates the example of CCN/NDN versioning, which is chosen to map different representations of DASH segments [10]. For instance, the *representation* 1 and 2 are directed to the versions of CCN URI denoted by $\_v1$ and $\_v2$, respectively. Similarly, DASH segmentation structure for the content is also supported by the ICN naming scheme. Hence, a DASH client can request the content appropriate to the estimation of the available bandwidth and network conditions [24]. ICN interest messages are issued to retrieve video segments, and in return, the video segments are provided either by the original content source or returned from in-network caches of routers.

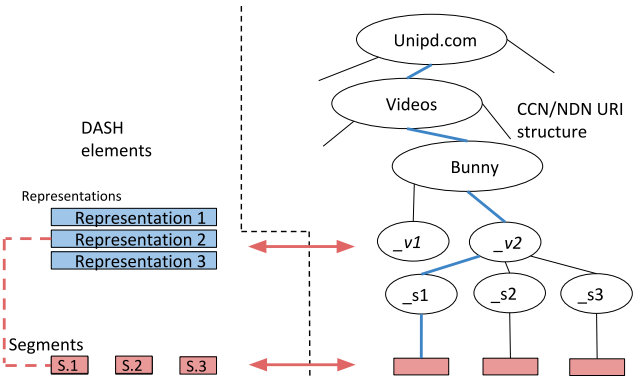

Fig. 3: DASH's representation mapping to the versions of ICN's naming scheme

The role of the DASH streaming control mechanism is to adapt the client requests based on the available bitrate and network bandwidth. Therefore, it provides 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. Also, it is able to provide smooth streaming with reduced bandwidth requirements for the origin server. Furthermore, the authors in [13] demonstrate 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.

We observe the behaviour of DASH streaming control system while interacting with ICN 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.

IV. System and Adversary Models

In this paper, we consider the scenario of multimedia streaming in ICN as illustrated in Figure 4. The content provider server called *producer* (P) stores the multimedia data (S) in a DASH-compatible format. S contains a collection of n number of equal-length segments, where each segment is available to be streamed in several media encoded bitrates (b), i.e., resolutions. A client (C) request 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(s) being requested by C in advance. Each interest from C and *Adv* traverses one or more routers before being satisfied by P or one of the ICN router, R. The goal of *Adv* is to degrade the QoE perceived by C during the streaming process. In our scenario, every router operates according to the default settings of ICN [19]. Moreover, the forwarding strategy adopted is *bestRoute* [7], which routes packets with respect to lowest path cost.

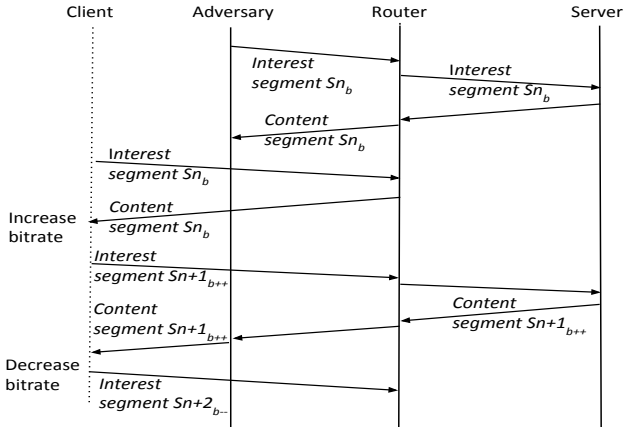

Fig. 4: Interest sequence to perform bitrate oscillation attack

A. System Model

In our scenario, all the entities, C, Adv, P, and R implement the ICN stack. For adaptive 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 types of coding formats for DASH-compliant multimedia content: Advanced Video Coding (AVC) [25] and Scalable Video Coding (SVC) [26]. In AVC, each segment of different bitrates is represented by a unique segment name. For instance, the first segment of the 100 kb/s representation is referenced as `/dash/bunny/_2s_100kbit/bunny_2s1.m4s`. In SVC, video content is encoded in different independent layers of quality called as the base layer (BL) and the enhancement layers (EL), where each layer subsequently enhances the video quality. The segments are referenced by the MPD file, in which these segments are listed by their URIs [6]. C requests

the segments according to the DAS streaming control system, 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 briefly describe the functionality of these DAS adaptation logic techniques.

1) Rate-Based adaptation logic: The *RB* adaptation algorithms [21], [27] stand on the idea of using the previous segment's measured bandwidth as a measure of bandwidth estimation for next segment. This is because C measures the available bandwidth on each instance while downloading the segment. By using an exponential moving average, C can estimate the available bandwidth for the next segment using the Equation 1.

$$\lambda_{k+1} = (1 - \beta) * \lambda_k + \beta * \lambda, \tag{1}$$

where λ_{k+1} denote the new estimate for bandwidth and λ_k denote the previous estimate. λ denote current bandwidth, which is calculated by taking the ratio of current segment size to its download time. β is a constant which reduces the impact of fresh measures on the estimate. This information supports C to select the highest affordable media encoded bitrate b_{k+1} as a requesting bitrate, i.e., $b_{k+1} < \lambda_{k+1}$.

2) Buffer based adaptation logic: The *BB* adaptation logic function is independent to bandwidth estimation instead it selects the video quality according to the current buffer occupancy $B(t)$. 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) [23] 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 [28] proves to be a stronger coexistence between *RB* and *BB* decision techniques. In this algorithm along with the bandwidth estimation for next segment λ_{k+1}, C also goals to stabilize the buffer level $B(t)$ around a target value (B_{max}). This keeps the adaptation process as smooth as possible by avoiding to react on short-term bandwidth spikes and stalling. In particular, the algorithm functions use two threshold values (B_{min} and B_{max}) along with λ_k and $\lambda_k + 1$. The increase/decrease of the video quality is governed in two ways. Decrease when $B(t) > B_{max}$ then algorithm keeps the current b. When the buffer lever is $B_{min} \leqslant B(t) \leqslant B_{max}$, it quickly shift to a lower quality. Furthermore, the lowest quality is requested when $B(t) < B_{min}$. Conversely, if the buffer level is $B_{min} \leqslant B(t) \leqslant B_{max}$ or greater than B_{max} the quality is increased with respect to estimated bandwidth.

B. Adversary model

In our analysis, we assume Adv connects to the same first-hop router to which C is connected, or to some on-path router(s) between P and C. By using geo-locating techniques [29], we could even relax the first assumption and require only that Adv just connects to the closest router. Using these existing techniques [29], Adv can identify the router closest to the consumer. In Section VII, we show the impact on C when the Adv launches the attack by connecting to different locations in the network. However, the maximum adverse impact to the victim results when Adv is connected to the same first-hop router, which later we take trademark in the rest of paper.

We assume that Adv has prior knowledge of the multimedia content (S) that C will be requesting in near future. Several existing techniques support 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 [30]. Moreover, Adv could also probe the MPD by exploiting the timing attacks to discover whether C has previously requested it or not. These techniques allow Adv to predict the video that C is going to request. Furthermore, it is quite possible that C may share the same wireless link, so the traffic traces are exposed and may be easily eavesdropped [31]. The Adv could infer the online activities of a user by analyzing the traffic and then it can be able to predict the content and its source [32].

V. 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 4.

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 [19]. After receiving the interest, P injects the requested content back into the network, which follows the same route from which the interest is received.

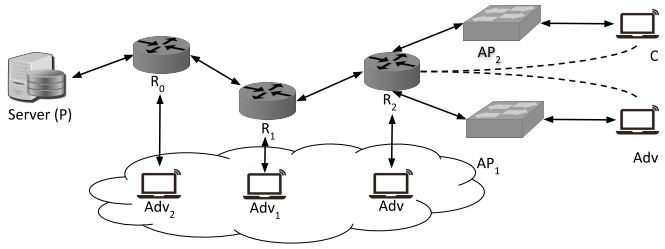

Fig. 5: Topology considered

To perform the attack, Adv issues legitimate interests for S in a specific order. Each interest requested by Adv is for a new segment, and it requests all the available bitrates (b) of that segment. The interest aggregation on ICN routers (specifically, in NDN/CCN) always aggregate similar interests and forward a single request on their behalf. In our attack scenario, since each interest requests are for a new S_n, therefore, each S_n is always counted as a new entry in the PIT, and thus it is forwarded to the upstream routers. Adv exploits interest aggregation to store all the legitimate interests on all the on-path router's PIT until the content is received back for those respective interests. In this way, by issuing different legitimate interests in the PIT of all on-path routers for each S_n (in all available bitrates, $b_{(i,j)}$), Adv can store segments of relevant interest on intermediate routers cache.

In our model, Adv requests the segments of S in ascending order, skipping a number of segments between each requested segment. As shown in Figure 5, 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 complete 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. In contrast, 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.

TABLE I: Summary of notations used

Notation	Meaning
Adv	adversary
C	client
P	server providing video
R_l	routers
AP	access point
S	video file at P
N	total number of segments in video
S_n	n^{th} segment of S
CS_l	cache at R_l
$b_{(i,j)}$	set of available bit rates of S
α	consecutive gap of variable length
b_f	bitrate of S received from CS
b_j	maximum bitrate
b_i	minimum bitrate
MPD	media presentation description (XML file)
$x(t)$	interest packet sending rate
$w(t)$	client interest sending window
c	link capacity
$d(t)$	inter-frame interval
T	sliding window time interval
s	MTU/fragment size
x	pre-defined constant value

To better illustrate the attack and its corresponding behavior at C, let's consider the topology in Figure 5 where two DAS-enabled clients Adv and C are connected to R_2 via AP_1 and AP_2. The router R_2 has a cache, say CS_2, that can hold the multimedia content S originating from P. S is available in single-layer coding formats such as Advanced Video Coding (AVC) and P provides multiple representations of each segment, here each representation features to a set of resolutions ($Sn_{(b_i..j)}$). These representations are listed in the MPD of S and are organized with respect to the ICN naming scheme for the segments [10]. Following the adaptive streaming control, C request a representation that best matches with its current network conditions, and it can adapt to fluctuations in the network bandwidth by switching to lower or higher representations. The Adv and C are streaming the content S from P. The Adv launches BOA at time t_i, and the C starts streaming at time t_j, where $t_j > t_i$. We now present the design and configuration of our BOA Algorithm 1 running at Adv. It requests content in an ascending order with a predefined consecutive gap (α). 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 in ascending order, i.e., $S(n+\alpha)$, where α is the consecutive gap of variable or fixed length to issue the discontinuous requests toward P with all the available bitrates ($b_{(i,j)}$) of S. Each request issued by the Adv is for a new segment with all available bitrates ($b_{(i,j)}$) till it requests a total of N number of segments. For each interest, say $S(n+\alpha)_{b_{(i,j)}}$, routers first check if the corresponding segment is available in their cache. Than check if the interest is listed in the PIT after that router forward it to FIB [2]. This is because the interest is not previously requested, and there is no segment relevant to it in CS. Therefore, each router lists the entry in PIT for these

Algorithm 1 Algorithm for Adversary (Adv)

procedure SEQUENCE_OF_INTEREST (S, i, j, α)
  $MPD \leftarrow$ *Send_ requests_ to_ P* ▷ $MPD = \{S(n)_{b_{i,j}}\}$
  for $n = 1, n \leqslant N, n + \alpha$ **do**
    $Content(S(n)_b) \leftarrow Interest(S(n)_b)$
    for $i \leqslant f \leqslant j$ **do**
      $Content(S(n)_{b_f}) \leftarrow Interest(S(n)_{b_f})$
        ▷ $S(n)_{b_i}, \ldots S(n)_{b_j}$ *caches on* CS_k
    end for
  end for
close;

interests and forwards them to P. These interests traverse the path $Adv \rightarrow Ap_1 \rightarrow R_2 \rightarrow R_1 \rightarrow R_0$. Following the characteristics of ICN [2], the segments in response to the interests follow the reverse path and these are cached at intermediate routers. At time t_j, C retrieves the MPD for the desired content. Based on the MPD, C sends interest messages to fetch S_n with an appropriate bitrate (say b_k) which is estimated based on its current networking conditions [4]. During the streaming session, when C request an interest $S(n+\alpha)$, the path that this specific interest will traverse is $C \rightarrow Ap_2 \rightarrow R_2$. It is because R_2 will return the content from its cache (CS_2). Let B_f be the bitrate representation of the segment received from CS_2, and b_{f+1} be the next, and b_i, b_j be the available lowest and maximum bitrate. For the interest $S(n + \alpha)$, C finds the download rate higher than the previous segment. Due to this, the DAS adaptation logic at C believes that the available bandwidth is suitable for receiving the maximum representation for next segment, and it switches to a highest representation, i.e., b_j. Now C requests $S(n+\alpha+1)$ with the maximum bitrate b_j, however, the interest explicitly traverses toward P through the path $C \rightarrow Ap_1 \rightarrow R_2 \rightarrow R_1 \rightarrow R_0$. This is because the requested segment has not yet been cached at R_2. Consequently, upon reception of $S(n + \alpha + 1)$ segment 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 $S(n + \alpha + 1)$. Therefore, C will switch to a lower representation for the subsequent segment. This process will be repeated again and again due to Adv, hence causing the requests from C to result in many cache hits and misses rapidly.

Algorithms 2 and 3 depicts the procedure at C to stream $S(n)_{b_{i,j}}$ adaptively after an attack. Algorithm 2 illustrate the procedure, where C requests N segments in a sequential order with dynamic bitrate adaptation. The bitrate selection for each requested segment in defined in Algorithm 3. For each segment request, adaptation control system of DASH estimates the download rate to select the best suited bitrate. Since the download rate is affected by the RTT of segments received from CS, therefore, for the segments $S(n + \alpha + 1)$, the highest bitrate is requested, i.e., b_j. However, again due to low download rate for the subsequent segment, the lower

Algorithm 2 C Segment selection process

procedure SELECT_SEGMENT_PROC($S(n)_{b_k}$,i,j,α,f)
  MPD $\leftarrow$ *Send_ requests_ to_ P* ▷ $MPD = \{S(n)_{b_{i,j}}\}$
  $S(r)_{b_k} \leftarrow$ *Select_ segment_proc()*
  $Content(S(r)_{b_k}) \leftarrow Interest(S(r)_{b_k})$
  $S(n)_{b_k} \leftarrow S(r)_{b_k}$
  $ind \leftarrow r$
  while $ind \neq N$ **do**
    $k \leftarrow$*Bitrate_adaptation_proc*$(S(n)_{b_k}, i, j, \alpha, f)$
    $Content(S(m)_{b_k}) \leftarrow Request(S(m)_{b_k})$ ▷ $m \in (r, N]$
    $S(n)_{b_k} \leftarrow S(m)_{b_k}$
    $ind \leftarrow m$
  end while
close;

Algorithm 3 C bitrate selection process

procedure BITRATE_ADAPTATION_PROC($S(n)_{b_k}$,i,j,α,f)
  if $n == r + q \times \alpha$ **then** ▷ $q \in \{1, \ldots, (N-r) div\ \alpha\}$
    $download\_rate \leftarrow Adaptation\_control\_sys()$
    if $k \leqslant download\_rate$ **then**
      $temp\_k \leftarrow w$ ▷ $w \in (i, j]$
    else
      $temp\_k \leftarrow w$ ▷ $w \in [i, j)$
    end if
  else
    $download\_rate \leftarrow Adaptation\_control\_sys()$
    if $k \geqslant download\_rate$ **then**
      $temp\_k \leftarrow w$ ▷ $w \in i$
    else
      $temp\_k \leftarrow w$ ▷ $w \in (i, j]$
    end if
  end if
  Return $temp\_k$
close;

bitrate is requested.

In order to simplify the functioning of BOA, i.e., exploitation of interest aggregation and in-network caching, we consider a specimen of multimedia data. Let's assume that C wants to fetch a video file (V_{file}) that consists of ten segments, say $S_{(1,.,10)}$. At producer, each segment is available in multiple video bitrate ($b_{i,j}$). To launch the attack, Adv issues a sequence of interests at time t_i in an ascending order, with a consecutive gap of one. The adversary requests each segment individually in all available bitrate, particularly, $S_{1(b_{i,j})}, S_3(b_{i,j}), .., S_9(b_{i,j})$. When the edge ICN router receives interests for $S_1(b_{i,j})$, it first checks the content store (CS) for the respective segment. Since the segments are not previously cached, therefore, PIT marks the entries for the interest ($S_1(b_{i,j})$) and forwards them to FIB in order to route them to the producer (P). On the retrieval of relevant

segments, all the on-path routers follow in-network caching [**?**], and thus stores $S_1(b_{i,j})$ in CS. In the similar way, the segments $S_3(b_{i,j}), S_5(b_{i,j})$,.., till $S_9(b_{i,j})$ are also requested by Adv and are stored in the CS of the on-path routers.

After the aforementioned process, assume that at time t_j ($t_j > t_i$), C starts streaming the same video file, i.e., V_{file}. Initially, it retrieves the MPD to get the characteristics associated with all the segments. For S_1, C requests the bitrate, say b_k by considering its own context, i.e., throughput estimation and DASH adaptation logic preferences [24]. Since all the bitrates for S_1, including b_k, are previously requested by Adv, therefore, the request for $S_1(b_k)$ is replied by the first on-path router's cache. It results in high throughput estimation for the next segment (i.e., S_2) due to reduced RTT of S_1. Due to the reduced RTT, C requests S_2 with highest bitrate, i.e., $S_2(b_j)$. However, $S_2(b_j)$ was not requested by Adv, thus it will not be available in the cache of the router. As a result, $S_2(b_j)$ is replied by the P with higher RTT. Based on the new RTT, the C again reduce the resolution and request bitrate b_k for S_3. For the full multimedia streaming, the above process of C, switching the bitrate for alternative segments between b_k and b_j repeats, thus the BOA is caused successfully.

Due to the functionality of the procedures mentioned above, C will experience undesirable bitrate oscillations, manifesting as continuous switches between high and low representations, leading to degradation in user-perceived QoE. Moreover, 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.

VI. Fair-RTT-DAS: Fair-RTT-based Dynamic Adaptive Streaming over ICN

In this section, we present the details of our proposed countermeasure called *Fair-RTT-DAS* that mitigates the BOA in DAS over ICN. We show that unlike the traditional Internet architecture, it is not sufficient to estimate the bitrate for next segment in ICN by just considering the RTT values of the adjacent receiving segments. This is because the producer location keeps on changing in ICN due to the content source variation caused by in-network caching and interest aggregation. It leads to the radical difference in RTTs of consecutive segments retrieved in an on-going streaming session. In general, the segment(s) retrieved from intermediate routers will have small RTT as compared to the ones received from the producer (P). If the change in the consecutive segment's location in a session is too frequent, DAS will falsely estimate a higher or lower throughput for the subsequent segments. We discussed in section V that the false throughput estimation stands as a vulnerability in DAS over ICN. We claim that in DAS over ICN, it is not sufficient just to discuss the throughput fairness which narrates the style of TCP/IP related research [33]–[35]. In our approach, we emphasize to maintain fairness in throughput estimation for the segments (within a single video file) with varied source locations.

To guarantee trustworthiness and evenness in bitrate adaption, we design a consumer-driven model. Our proposal preserves the fairness in the segment's RTT within a streaming session. Fair-RTT-DAS countermeasures the inference attacker with the aim to attain the following requirements.

- significantly reduce the precision of BOA in order to effectively alleviate the adverse impacts of the attack.
- efficient bandwidth utilization to download appropriate bitrate segments in dynamic network conditions.
- ensure scalability in terms of the deployment of our countermeasure with lower additional overheads.

To accomplish the above objectives, Fair-RTT-DAS dynamically estimates the available bandwidth, identify the false estimations, and circumvent for the default bandwidth estimation process that is used in conventional bitrate adaptation method. Fair-RTT-DAS framework ensures fair-RTT based mechanism on top of DAS streaming control system. The significant element of Fair-RTT-DAS is its unique rate adjustment function that leads to adequate throughput estimations in presence of content source variations.

Fair-RTT-DAS entails a set of algorithms integrated to DAS client with the goal to identify the attack and moderate their marks. In addition, the intermediate routers are not required to report the accumulative statistics of inward and outward interests to DAS client. Thus, Fair-RTT-DAS has the advantage due to its ease of deployment and scalability in view of growth in network traffic. Fair-RTT-DAS consists of two major phases: (i) *detection*, and (ii) *reaction*. In the detection phase, the DAS client identifies the attack, while the reaction phase eccentrically controls the sending rate of the interest packets.

A. Basic idea of Fair-RTT-DAS

The elementary idea of Fair-RTT-DAS is to enforce the DAS client to use a collaborative approach for bitrate adaptation, which includes our definition of RTT-fairness in conjunction with conventional bandwidth estimation approach. Since conventional bitrate adaptation depends mainly on bandwidth estimations, thus it is not adequately ingenious to identify the variations in the source of the content. Hence, to keep the bitrate oscillations to the minimum and low switching amplitude [17], [36], our strategy exploits fair-RTT-based bitrate adaptations to fetch the best available bitrate representation in the presence of an adversary.

The foremost concern is to ensure that the DAS client identifies the segments of the media content with the variation in source location, during the streaming process. Secondly, bitrate adaptation process should efficiently request the available representations to avoid the false bandwidth calculations to mitigate the BOA. By careful analysis of the variations in RTT between the consecutive interests and their corresponding content of segments, Fair-RTT-DAS identifies the symptom of adversarial presence. Hence, to avoid the inferior behavior of DAS, Fair-RTT-DAS dynamically modifies the sending rate of interest messages for the segments that might cause the QoE degradation in the session.

B. Detection Phase

To maintain RTT-fairness, our model identifies the source for chunks (i.e., fragments[2]) by considering packet level communication statistics. Although ICN routers cache segments and frames independently, but DASH performs bandwidth estimation on the segment level listed in MPD, instead of the fragment level. Below we describe in detail the functionality of Fair-RTT-DAS for intra-segment streaming (i.e., at fragment level) and source variation detection.

1) Intra-segment Communication: Fair-RTT-DAS aims to identify the variation in content source by taking advantage of the transport and link layer statistics [37] [33] [38]. The segments in DASH stands as a sequence of bytes identified by a globally unique identifier, and these may vary with each other in size [39]. The network devices are restricted to forward packets up to the maximum transmission unit (MTU), thus segments are fragmented into smaller chunks (or frames) before transmission. The common design approach of fragmentation - as this paper assumes - is that each resulting fragment of a segment represents a uniquely identifiable piece of transmission and addressed unit in its own right [37], [38], [40]. For instance, any segment larger than 1449 bytes [3] is fragmented and identified independently. When DASH client requests a segment longer than the maximum packet size of the network, the response contains the meta-data of that requested segment. The meta-data includes fragment identifiers (i.e., interests) that makes up the requested segment, segment size, and additional security and integrity information [38] [41], as it is shown in Figure 6. DASH client then issues pipeline of interests to request fragments simultaneously in order to completely utilize the link capacity. This is accomplished dynamically by calculating the instantaneous *interest sending rate* $x(t)$ that client handles [34],

$$x(t) = \frac{w(t)}{p + \frac{q(t)}{c}}. \tag{2}$$

Here, $w(t)$ defines the size of the receiver window, which is the maximum number of interest a client is allowed to send while waiting for the corresponding segments. $q(t)$ denotes the instantaneous queue occupancy at the source, c is the link capacity in packets/sec, and p is the round trip propagation delay. The equation shows that the interest rate linearly grows proportional to the inverse of round trip time RTT.

Based on the above pattern, gauging the segment request is an umbrella concept, which consists of a group of consecutive interests for fragments belonging to the same segment. Typically, there is no correlation between fragment requests related to two consecutive segments for bitrate selection directly. The DAS bitrate adaptation model selects appropriate representation for the whole next segment based on the previous segment download rate. This approach completely ignores the fragment-level effects on bitrate selection. We design a model to integrate the intra-segment communication with conventional bitrate adaptation model. For this purpose, Fair-RTT-DAS keeps several statistics on intra and inter-segment communication level. In particular, it accounts the RTT of the initial fragment (RTT') relevant to a previous segment(s). It also records the *interest sending rate* $x(t)$ used to request the fragments in the previous segment(s).

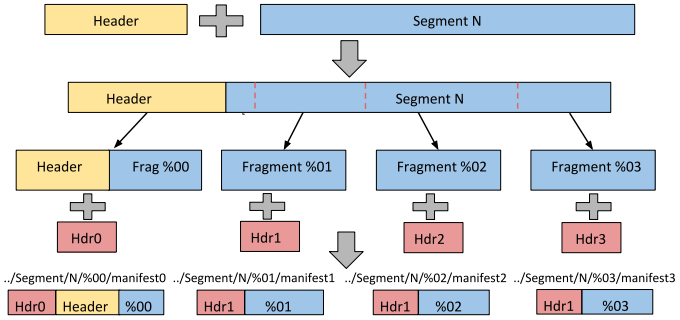

Fig. 6: Segment fragmentation

2) RTT Measurements: To identify the source variations throughout a session, Fair-RTT-DAS measures the time taken to send the first interest for each segment and receives the relevant content. As described earlier, this content may contain information relevant to fragmentation and link-layer communication. The resulting rate of interest packet sending and content delivery highly depends on the notion of the distance between the client and the content source. Here we explicitly define it through *effective round trip time* (RTT'), in analogy with the RTT of connection-based transmission protocols like TCP [34].

Our mechanism implements the bootstrapping mode on DASH client, which operates by calculating the moving average of RTT' for initial fragments received from a number of previous segments in a constant time period, T. Since the proposed attack relates to the variation in the source location, Fair-RTT-DAS compares the RTT' variations with the time T, which in case of a successful attack is noticeably radical as compared to the network congestion. In addition, Fair-RTT-DAS uses average measured RTT' in each consecutive T, instead of the weighted average of RTT'. It is because the weighted average is effective only in case of smooth rate adaption and removal of measuring errors. Moreover, it is more functional to eliminate the variation in long-term source variations. Therefore, the average measured RTT' for each constant time period enables the detection of attack [42]. In addition, to distinguish these reasons, i.e., the RTT' variation caused by adversary or network congestion, the statistical time unit (i.e., T) must be set to a small value.

3) Source Variation Detection: In our method, DASH client remains in bootstrapping mode until it executes the measurement of average RTT', namely $RTT'(T)_{avg}$ in the first T

[2]Please note that a video session is divided into multiple segments and a segment greater than MTU could further consist of multiple fragments.

[3]Amustndnsim currently packetizes media into segments that are less than the typical 1449 bytes.

time period. After this bootstrapping mode, Fair-RTT-DAS compares the RTT' of each first fragment of the current segment with the $RTT'(T)_{avg}$ of previous segments. In case the RTT' received of the current segment is greater than $RTT'(T)_{avg}$, the detection phase leads the system to move as per DASH streaming control system. However, if the value is radically smaller w.r.t $RTT'(T)_{avg}$, this leads to identify the attack and source variation, as shown in Equation 3. In this way, the detection phase is able to identify source variations among consecutive segments which can trigger oscillations.

$$RTT' + jitter \lll RTT'(T)_{avg} \tag{3}$$

To better illustrate the bootstrap phase, we designed a model in Algorithm 4, which stores the RTT' of the manifest file of each segment called $manifest(S(r)_{b_k})$ that were received in during last T time period. In particular, the first fragment to request is the file containing meta-data (i.e., manifest file). While streaming the session, if the time value ($Time\_queue$) is less than T, Fair-RTT-DAS selects the b_k as per default bandwidth estimation. Therefore, DAS streaming control system selects the appropriate bitrate as shown in Algorithm 3. Furthermore, the interest packet sending rate $x(t)$ is maintained by Equation 2. Throughout the streaming, the T (i.e., the time in which average RTT' (RTT_queue) is measured) slides along the session and it always stores the $RTT'(T)_{avg}\_manifest(S(r)_{b_k})$ for previous T. In particular, Fair-RTT-DAS uses moving average $RTT'(T)_{avg}$ in each consecutive T, which is always measured dynamically at each instance for previous T period. Therefore, the moving average measured ($RTT'(T)_{avg}$) for each constant time period enables the detection of attack or source variation in an ongoing streaming session. The proposed novel fragment request procedure is described in Algorithm 5, which operates during bootstrap-phase and after it.

4) Parameter setting: Our investigation shows that maximum oscillations during the attack only happens when an attacker generates interests with consecutive gaps. In case, if attacker issues two contiguous interests with a consecutive gap, the efficiency of the attack reduces by 50%. In fact, adversary helps the victim to improve its QoE, if it requests more than two continuous segments. Since in this case, the client will be receiving most of the continuous segments from the edge or intermediate routers, resulting in reduced bitrate oscillations and high bandwidth utilization. Therefore in our experimental analysis, adversary requests a unilateral sequence of segments to achieve the maximum efficiency in the attack. The figures 7 and 8 show the relation of switching frequency and number of continuous segments requested by attacker. From the results, it can be seen that switching frequency declines when Adv requests more number of consecutive segments.

The value of the time period (T) depends highly on user-end application and service requirements for a streaming session. The higher value of T may improve the smooth bitrate selection, but it can be a factor against efficient bandwidth utilization. The value of T also relates to the number of

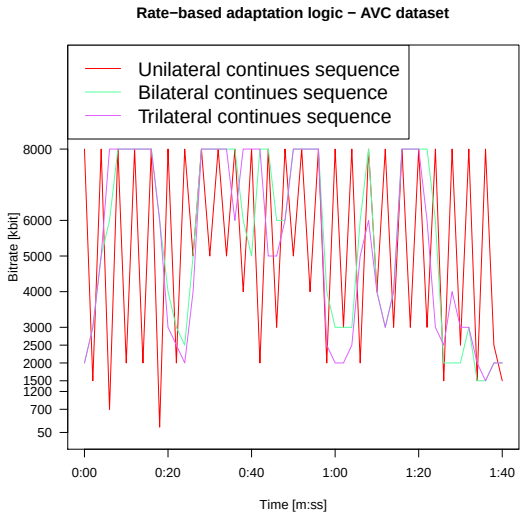

Fig. 7: Comparison of attack sequences (RB-AVC)

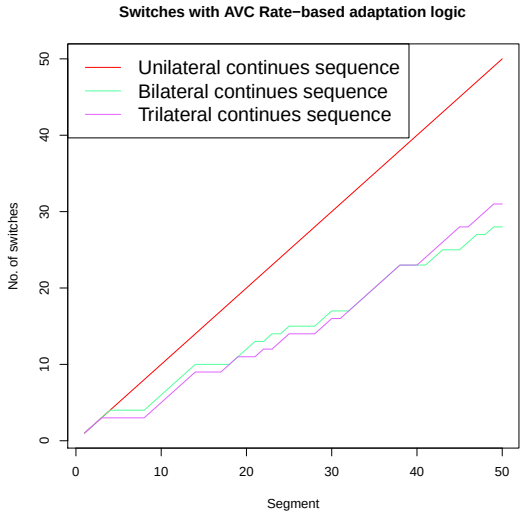

Fig. 8: Switches comparison of attack sequences (RB-AVC)

segments issued continuously by an attacker. It is because the detection phase uses the value of T to detect the RTT' variations. In addition, the number of segments falling in the T period helps reaction phase to identify the positive cache hit which may help to increase the user's QoE. For instance, if many continuous segments are available on the router's cache due to content popularity, the reaction phase will efficiently utilize the bandwidth and switch to higher bitrates (later detailed in SectionVI-C).

The simulation results in Figures 7 and 8 reports that maximum impact of attack is seen at times when adversary generates unilateral continuous segment with consecutive gap. Therefore, if the attacker is issuing the request to generate maximum attack impact, than the minimum number of segments to fully detect the attack should be more than one. In

Algorithm 4 Fair-RTT-DAS algorithm (Bootstrap phase)

procedure SELECT_SEGMENT_PROC($S(n)_{b_k}$,i,j,α,f,T)
  $MPD \leftarrow$ *Send_ requests_ to_ P* ▷ $MPD = \{S(n)_{b_{i,j}}\}$
  $S(r)_{b_k} \leftarrow$ *Select_ segment_proc()*
  $Content(S(r)_{b_k}) \leftarrow Interest(S(r)_{b_k})$
  $Time\_queue := empty$
  $RTT\_queue := empty$
  $Flag : false$
  while $(Time\_queue.sum() \leq T)\&(r \neq N)$ **do**
    $k \leftarrow$*Bitrate_adaptation_proc*$(S(r)_{b_k}, i, j, \alpha, f)$
    $t\_end(S(r)_{b_k}, t\_start(S(r)_{b_k}, RTT'\_manifest(S(r)_{b_k}), S(m)_{b_k} \leftarrow Request\_fragment(Flag, Time\_queue, RTT\_queue)$
    $Time\_queue\_add(t\_end(S(r)_{b_k}$-$t\_start(S(r)_{b_k})$
    $RTT\_queue\_add(RTT'\_manifest(S(r)_{b_k})$
    $S(r)_{b_k} \leftarrow S(m)_{b_k}$ ▷ $m \in (r, N]$
  end while
  $Flag : true$
  while $r \neq N$ **do**
    $k \leftarrow$*Bitrate_adaptation_proc*$(S(r)_{b_k}, i, j, \alpha, f)$
    $t\_end(S(r)_{b_k}, t\_start(S(r)_{b_k}, RTT'\_manifest(S(r)_{b_k}), S(m)_{b_k} \leftarrow Request\_fragment(Flag, Time\_queue, RTT\_queue)$
    $Time\_queue\_add(t\_end(S(r)_{b_k}$-$t\_start(S(r)_{b_k})$
    $RTT\_queue\_add(RTT'\_manifest(S(r)_{b_k})$
    if $Time\_queue.sum() \geq T$ **then**
      $Time\_queue.PoP()$
      $RTT\_queue.PoP()$
    else
      $Flag : false$
    end if
  end while
close;

our case, the most appropriate value of T is at least equal to the playback duration of two segments. In addition, the relationship of T with the bandwidth utilization is shown in Figure 9, and the results report that for increased value of T, the client exhibits smooth rate adaptation and fewer switches. Although, it remarkably reduces the bandwidth utilization and streams with lower bitrate values.

C. Reaction Phase

Once an ongoing attack is suspected, it triggers the Fair-RTT-DAS to react. To have a smooth rate adaptation, we name this process as *adaptive phase*, and it aims to adopt best possible bitrate in the presence of an adversary. For each segment after the detection, the countermeasure adaptively increases the interval time between the fragment requests belonging to the segments. In general, client sends interest fragments according to the sending interval $d(t)$ (please refer to Equation 4). In Equation 4, k (refer to Equation 5) is the correction of data packet size; s (in bytes) is the maximum chunk size (i.e, MTU), and u is the pre-defined constant value [42]. The size of the data packet (s) is highly dependent on the individual application and network. Also, the bandwidth

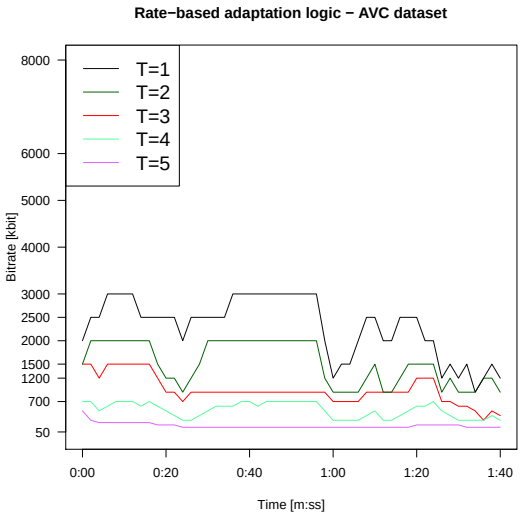

Fig. 9: Bandwidth and T relation (RB-AVC)

consumption of their data packets are different when each application send s interests with the same $x(t)$. Therefore, we use k to remove the difference in data packet size among competing data packet flows. For instance, $d(t)$ will increase with the growth of s as compared with other flows. In our model, we take s uniform in the whole scenario. Hence, to mitigate the attack, the adaptive phase decreases $x(t)$ for each identified segment that increases the interval between the interests sent for fetching the fragments $d(t)$.

$$d(t) = \frac{1}{k * x(t)}; \tag{4}$$

$$k = \frac{u}{s}. \tag{5}$$

The functionality of the bootstrap phase is to process according to the DAS steaming control system and to select the $x(t)$ according to default streaming system. In case of a successful attack while bootstrapping is ongoing, it may seem to have false bandwidth estimation. However, it is required since to obtain $RTT'(T)_{avg}$. After the bootstrap phase, if the algorithm detects the attack, the DAS client adopts to the adaptive phase. The key contribution of the adaptive phase is to reduce interest sending rate ($x(t)$) after attack identification for the future segment. Hence, the bitrate estimation done by DASH streaming control system is directly affected by controlling $x(t)$ to have better QoE. It functions and controls the fragment sending rate in order to maintain evenness in downloading rate of identified segments comparing to previous ones.

In case of an attack, when a fragment is replied back from the *router* instead of the *producer*, RTT' decreases drastically. The adaptive phase will decrease the interest sending rate by increasing the $d(t)$ for the remaining fragments within that specific segment. It leads to the reduced download rate, although, the segment is replied by the router's cache. Here the parameters to control the $x(t)$ are made directly proportional to QoE perceived by the user. Therefore, the adaptive phase maintains the inter-packet interval, which will be the same as it is experienced by the previous segment. To accomplish this, the DAS client maintains $d(t)$ for current segment using the previous segments $x(t)$ values.

Fair-RTT-DAS performs equally better in case of content source variation. For instance, if most of the continuous segments are stored on routers, it will help to increase the QoE for default dynamic adaptive streaming. It is due the advantage of two fundamental aspects which Fair-RTT-DAS takes into account. First, moving average RTT calculation of initial fragment for each segment in T period, ($RTT'(T)_{avg}$). Secondly, smooth bitrate adaptation which also considers the positive cache hit for efficient bandwidth utilization. Therefore, considering the value of T as a playback duration of two segments and its sliding nature, the $RTT'(T)_{avg}$ always reduces when subsequent segments are being retrieved from the router's cache instead of the producer. Later, this new reduced value of $RTT'(T)_{avg}$ will be used to detect the source variation, which will help to increase the bitrate. For instance, if more subsequent segments results in lower RTT', the adaptive phase selects higher interest sending rate, by maintaining the packet interval of previous segment $d(t)$, which is retrieved from the router's cache. It is worth mentioning here that we have selected the value of T equal to the playback duration of two segments. Hence, if the subsequent segments are coming from routers cache, e.g., with reduced RTT', the adaptive phase will select previous $d(t)$ value, which is more reduced. In this way, the interest sending rate ($x(t)$) is increased for the current segment, and it results in increased bitrate adaptation in case more subsequent segments are available in caches, and it helps to increase the bandwidth availability for DASH client.

The detailed description of the adaptive and detection phase is given in Algorithm 5. To detect, it compares the RTT' of manifest of each segment ($RTT'\_manifest(S(r)_{b_k})$) with the average $RTT'(T)_{avg}\_manifest(S(r)_{b_k})$ which is estimated over time T. In case of an attack, the DASH client reduces the $x(t)$ to the value of previous segment. This leads to increase in the inter-fragment interval ($d(t)$) within a segment. Conversely, in order to make client compatible with network congestion and to increase bandwidth, the fragments are sent as per default DASH streaming control system, i.e., instantaneous *interest sending rate* $x(t)$ that client handles [34] (refer to Equation 2).

VII. Evaluation and Result Analysis

In this section, we investigate the performance of adaptive multimedia streaming over ICN in presence of adversary. We implement and evaluate the effectiveness of our proposed countermeasure namely Fair-RTT-DAS for multimedia streaming over DASH in ICN. To this end, we perform extensive simulations using AMuSt-ndnSIM, which is an Adaptive Multimedia Streaming Framework for ndnSIM [16]. AMuSt-ndnSIM framework provides support to create a bridge between multimedia traffic and NDN [19], categorically based on ndnSIM [43] [44] and libdash [45]. Note that NDN is a specific instantiation of ICN which is well-suited for this evaluation. AMuSt framework offers a set of applications for producing and consuming adaptive video traffic but exchanging HTTP with NDN. The functionality of DASH is provided by the libdash library, which is an open source library with an interface to DASH standard and an official reference software for DASH standard [45].

A. Test Setup

To set up the tests, we implement the network depicted in Figure 5 with a single origin server and a number of multimedia clients (including honest and malicious hosts) connected with multiple NDN routers. To configure the producer (P) with real-time video traffic, we use an AVC-encoded multimedia video [25] and the *BigBuckBunny* movie from the DASH/SVC Dataset [26]. Other network parameters and their values used in our test setup are given in Table II. The forwarding strategy used at NDN routers is minimum hop count *(BestRoute)*, and we chose *Least Recently Used* (LRU) as a caching policy for router caches.

Algorithm 5 C Fragment request algorithm (Detection and Reaction phase)

procedure REQUEST_FRAGMENT_PROC($S(r)_{b_k}$,Flag, Time_queue,RTT_queue)
  $Content(S(r)_{b_k}) \leftarrow Request\_manifest(S(r)_{b_k})$
  if Flag $=$ true **then**
    if $RTT'\_manifest(S(r)_{b_k} \lll RTT\_queue.sum \div RTT\_queue.length()$ **then**
      $x(t)\_(S(r)_{b_k}) \leftarrow x(t)\_(S(r)_{b_k-1})$
      $Content\_fragment_i\_(S(r)_{b_k}) \leftarrow Content\_fragment_i\_(S(m)_{b_k})$ ▷ $i \in (1,n], m \in (r,N]$
    else
      $DASH\_control\_sys() \leftarrow x(t)\_(S(r)_{b_k})$
      $Content\_fragment_i\_(S(r)_{b_k}) \leftarrow Content\_fragment_i\_(S(m)_{b_k})$ ▷ $i \in (1,n], m \in (r,N]$
    end if
  else
    $DASH\_control\_sys() \leftarrow x(t)\_(S(r)_{b_k})$
    $Content\_fragment_i\_(S(r)_{b_k}) \leftarrow Content\_fragment_i\_(S(m)_{b_k})$ ▷ $i \in (1,n], m \in (r,N]$
  end if
close;

TABLE II: Parameters for simulations

Parameters	Value
No. of video segments (N)	250
Video period(s)	100
No. of edge routers	1
Available bitrates (AVC)	20
Layers of quality (SVC)	4
Duration per segment(s)	2
Bandwidth between the nodes (Mbps)	10
Delay between C to edge router (µs)	200
Point to point delay (µs)	10
Max buffer size (s)	30
Consecutive gap α	2
fragment size s (byte)	1449
Constant value u	1449
Value of T (s)	3
Drop Tail Queue (max. packets)	20
Cache policy	LRU
Start up delay (s)	0.1
Max. buffered time (s)	30

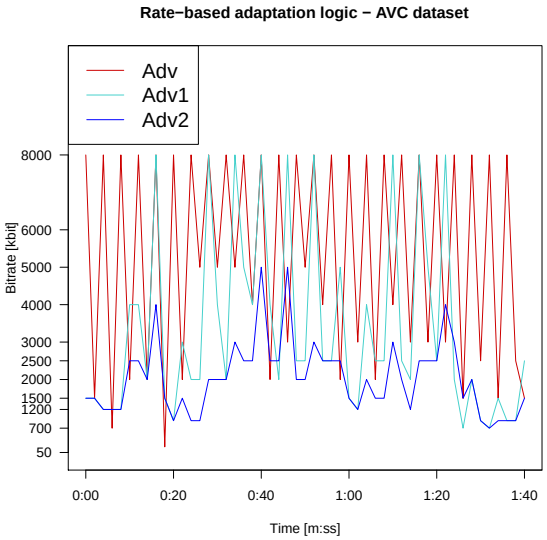

Fig. 10: Dynamic adaptive streaming to different adversarial locations using RB (AVC)

B. Evaluation Metrics

The QoE in video streaming relies on the intermingling of high video quality (e.g., high bitrate) and high streaming performance (e.g., continuous playback without re-buffering). The authors in [46] and [17] explain the impairment factors that affect the user experience for dash video, and it illustrates that frequent switching in video representations in a session diminishes streaming quality. Thus, the spatial quality of video can be determined by the level of variations occurred during a streaming session. To evaluate the attack and Fair-RTT-DAS, we use the following metrics.

- Number of switches: It indicates the frequency of video quality switches [36];
- Average switch magnitude: It indicates the average amplitude of the video quality switches [17], [36].

C. Attack Impact

Figures 10 and 11 show that Adv is able to degrades the QoE of client while being connected to any on-path router, however, the maximum degradation occurs when it is connected to the same first-hop router. In simulation results, we report the case when Adv is connected to the same first-hop router to which the client is connected. Figures 12 and 13 report the bitrate requested by the DAS client for both the cases (with and without the attack) using RB and R&B adaptation logic. Our results in Figures 15 and 16 show that frequency of bitrate switching in RB and R&B adaptation logic increases remarkably in the presence of an adversary. Moreover, the attack massively increases the average switch

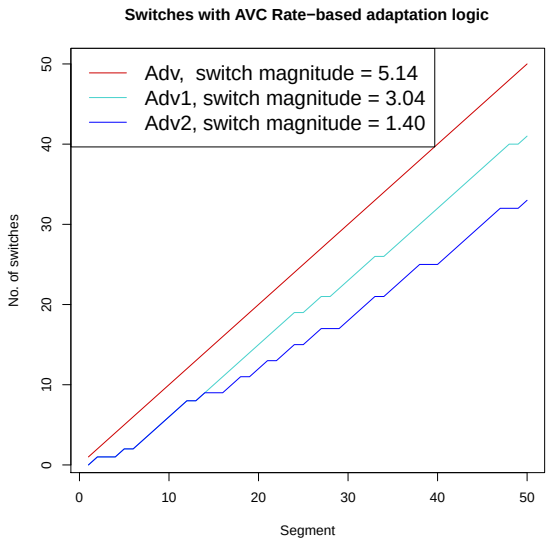

Fig. 11: # of switches to different adversarial locations using RB (AVC)

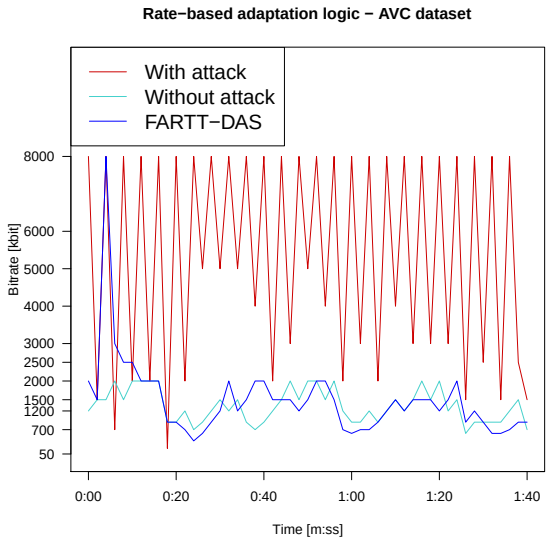

Fig. 12: Dynamic adaptive streaming using RB (AVC)

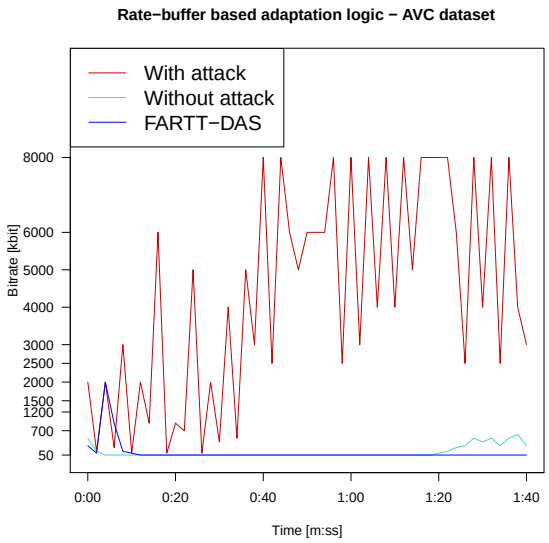

Fig. 13: Dynamic adaptive streaming using R&B (AVC)

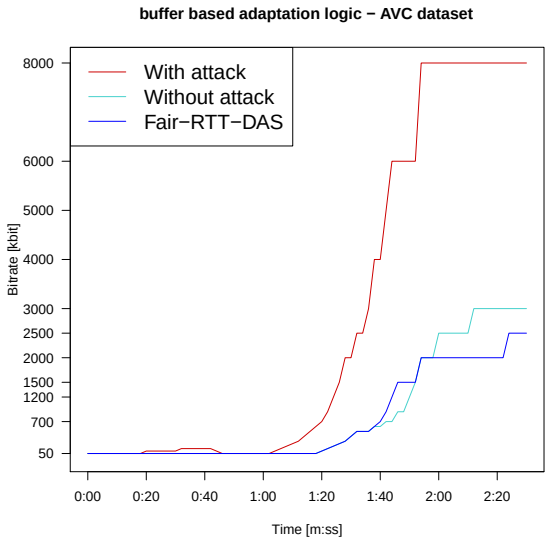

Fig. 14: Dynamic adaptive streaming using BB (AVC)

magnitude of bitrate fluctuations for these adaptation logic while streaming AVC content (please refer to Figure 22).

The results in Figure 14 shows the bitrate requests in BB adaptation logic (AVC). Figure 17 depicts that the DASH client experiences an increase in the number of switches, and it can be seen at first glance that a victim is experiencing higher video bitrate. However, considering the user's QoE evaluation metrics (detailed in Section VII-B), this not satisfactory. From Figure 17, we can see that the number of switches in Buffer-based (AVC) are much higher in presence of an adversary for default DASH over ICN. But, with Fair-RTT-DAS, it is merely equal to scenario without attack. Similarly, we can see the average switch magnitude difference in Figure 22, which shows that without our proposed approach the user is experiencing approximately 40% more average switch magnitude (i.e., 0.3 with attack, and 0.18 with Fair-RTT-DAS). From these two metrics, we can observe a remarkable QoE degradation for users in the presence of attack, and Fair-RTT-DAS is able to mitigate the attack as well as it performs efficiently as compared to normal scenario (i.e., without attack).

Categorically, for all adaptation logic in AVC, Figure 22 shows that the client experiences an increase in average switch magnitude of bitrate fluctuations in the presence of the attacker. This increased switch magnitude is experienced by client because the adversary forces the client to switch multiple

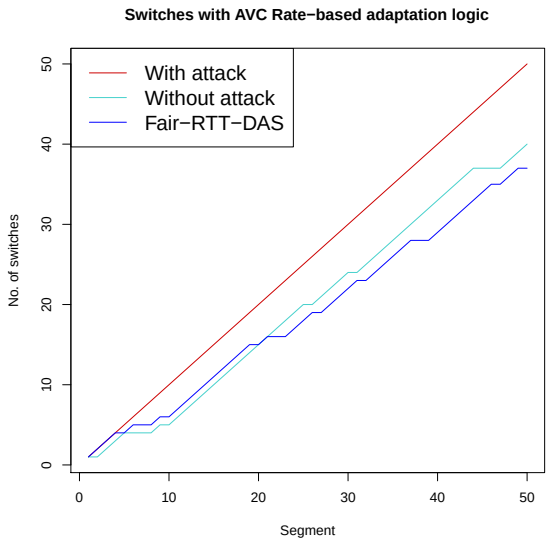

Fig. 15: # of switches RB (AVC)

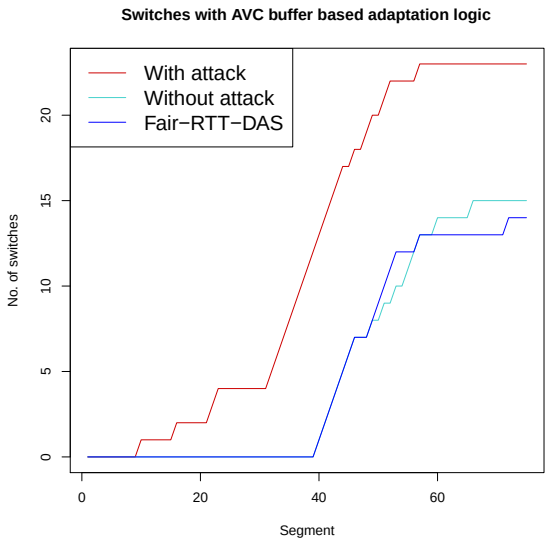

Fig. 17: # of switches BB (AVC)

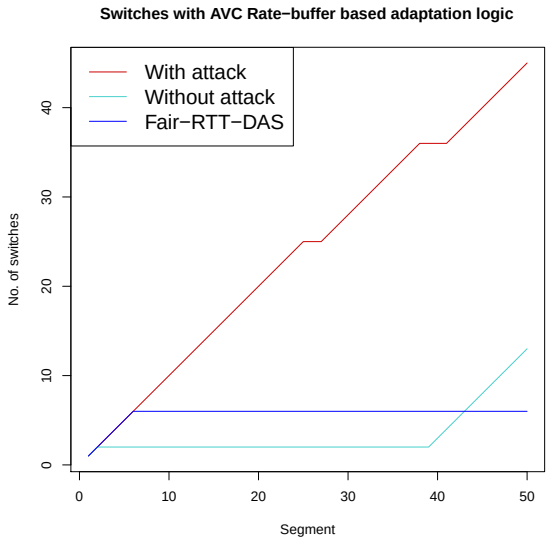

Fig. 16: # of switches R&B (AVC)

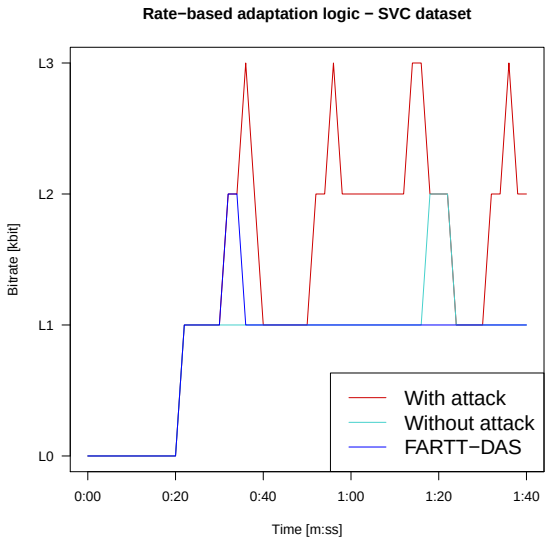

Fig. 18: Dynamic adaptive streaming using RB (SVC)

times and between extremely low and high resolutions.

For the RB adaptation logic in the SVC dataset, the attack results in higher frequency of switches in the download bitrate (refer to figures 18 and 20). However, the magnitude of the switches is relatively small when compared to what we found for our AVC dataset as it is shown in Figure 22. It 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 reasonable QoE degradation. From the simulations, we also observe that buffer-based adaptation logic in SVC dataset is unaffected by the attack. Figures 19 and 21 show that there is no increase in bitrate fluctuations and average switch magnitudes. We identify that buffer capacity affects positively and resits to short-term bandwidth fluctuations. However, use of the buffer based adaptation logic 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.

In our simulation setup, we consider the case of a single DASH client in the presence of one or more Adv. It is because we focus on the attack, which is subjective to the vulnerability identified in DASH bitrate adaptation logic. The case in which more than one victim may attach to the same

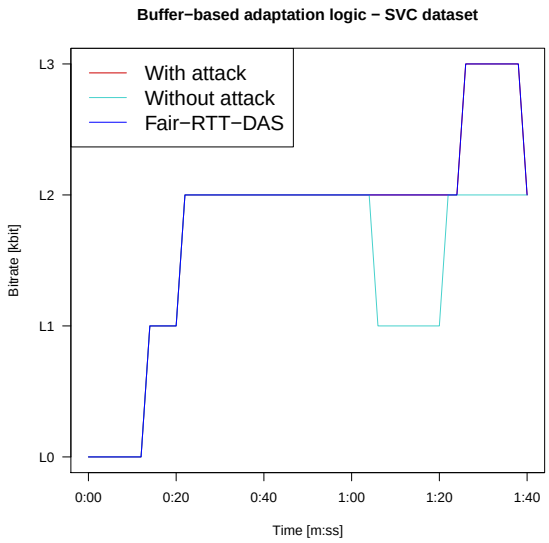

Fig. 19: Dynamic adaptive streaming using BB (SVC)

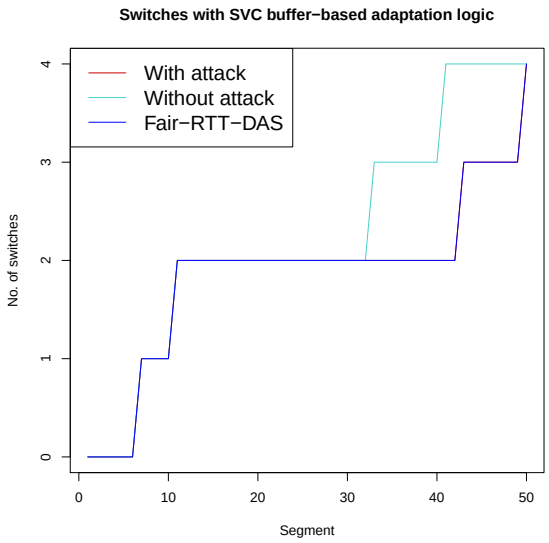

Fig. 21: # of switches BB (SVC)

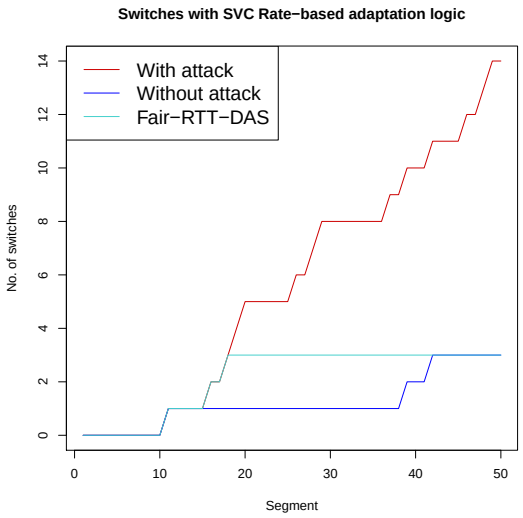

Fig. 20: # of switches RB (SVC)

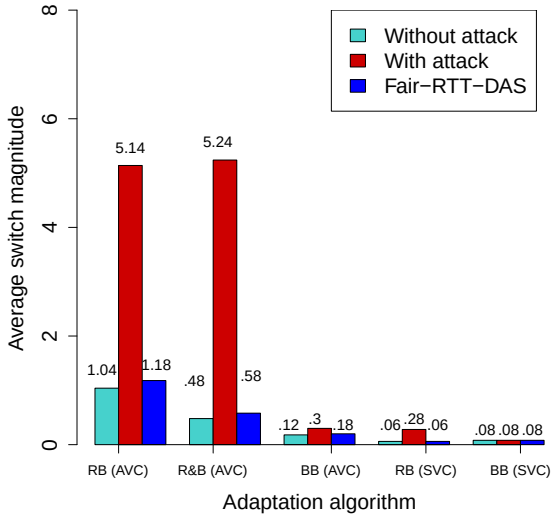

Fig. 22: Average switch magnitude

edge router to which Adv is connected also faces the similar QoE degradation. However, it entails two cases: (i) the bitrates requested by all the victims solely depends on their bitrate adaptation strategies and hardware constraints. Besides, all users might face different bandwidth fluctuations considering different access networks, e.g., WiFi, LAN, 3G, 4G etc. Therefore, it is not guaranteed that all the users will request similar video bitrates for the same multimedia content. In this case, all the victims will face the related impact of attack since all clients will counter the specific interest sequence generated by Adv, and (ii) if there are numerous victims attached to the same first-hop router and requests the same multimedia content, it may result in content popularity for that content since all tend to request the segments in a sequential manner with different bitrates. Due to an increased number of clients requesting the same segments also increases the chances of same bitrates requested by the clients. Therefore, considering the efficacy of in-network caching, most of the continuous segments with multiple bitrates gets stored on the routers. It helps the bandwidth constraints and improves user's QoE since most of the consecutive segments with various bitrates can be fetched from routers instead of the origin server.

D. Fair-RTT-DAS Effectiveness

The simulation results for our proposed countermeasure reports that DASH client is able to sustain the perceived QoE in the presence of an adversary. This is because Fair-

RTT-DAS maintains RTT smoothing within the packets of the same video session. Our approach identifies the source variations which are hard to identify for a DASH client. After identifying the attack, Fair-RTT-DAS makes the DASH client to select the most appropriate bitrate with respect to the best possible QoE perceived. Fair-RTT-DAS also satisfies the fundamental characteristics of adaptive streaming, since DASH client adaptively adjusts the bitrate representations while effectively utilizing the bandwidth in fluctuating network conditions.

Figures 12 and 13 report the performance of Fair-RTT-DAS with and without adversarial model using the RB and R&B adaptation logic. It highlights the phenomena in which the clients follow the victim's pattern initially, which indeed represents the bootstrap phase. However, later it rapidly implements smooth bitrate adaptation. The results in Figures 15 and 16 illustrates that frequency of bitrate switching in above mentioned adaptation logic declines remarkably in the presence of an adversary. In addition, we notice a slight improvement in bitrate fluctuations comparing to traditional DASH due to bootstrapping phase. However, for some extent Fair-RTT-DAS compromises on bandwidth utilization in order to deliver smooth bitrate adaptation. The Fair-RTT-DAS correspondingly upholds the average switch magnitude of the bitrate fluctuations to default values for the adaptation logic (please refer to Figure 22). Results in Figure 14 show the bitrate requests for Fair-RTT-DAS in BB adaptation logic (AVC) and Figure 17 confirms the reduction in the switching frequency. Figure 22 confirms that the DASH client experiences merely equal value to default for average switch magnitude of the bitrate fluctuations in the presence of an attacker for all three examples of adaptation logic.

Fair-RTT-DAS results in reduced bitrate oscillations (refer to Figures 18 and 20) for the RB adaptation in SVC dataset. Moreover, it also maintains the equivalent magnitude for a number of switches when compared to the conventional DASH streaming system as it is shown in Figure 22. From Figures 19 and 21, we can identify that BB adaptation logic in SVC dataset takes advantage of the attack. The client may seems to have better resolution due to pre-fetching and it takes advantage of buffer size and SVC. However, the Fair-RTT-DAS is also unaffected in this scenario and participate to enhance the victim's perceived QoE.

VIII. Conclusion

The key features of ICN which includes in-network caching and native tendency to support multicast has been shown to have unforeseen privacy consequences [30]. In this paper, first we show that how an adversary can exploit the implicit features of ICN (i.e., in-network caching and interest aggregation) and the adaptive streaming mechanism of DASH, to degrade the performance of DASH over ICN. Then, we propose a countermeasure to detect and mitigate such an attack. We validated our proposed approaches (the attack and its countermeasure) via simulations that are done on AMuSt-ndnSIM. Through result evaluation and analysis, we conclude that high frequency of bitrate switching increases the annoyance factor in spatial dimension and the high amplitude of oscillations decrease the satisfactory visual quality (i.e., degrade the user perceived QoE) at the end-user. Our countermeasure uses the concept of maintaining RTT and throughput fairness in ICN's dynamic network condition to alleviate the adverse effects of adversary. Moreover, it shows that it can further enhance the user perceived QoE in presence of varied content source locations and ICN's implicit characteristics.

Acknowledgement

Mauro Conti is supported by a Marie Curie Fellowship funded by the European Commission (agreement PCIG11-GA-2012-321980). This work is also partially supported by the EU-India REACH project (agreement ICI+/2014/342-896) and EU TagItSmart! Project (agreement H2020-ICT30-2015-688061).

References

[1] "Cisco visual networking index: Forecast and methodology, 20152020," .

[2] V. Jacobson *et al.*, "Networking named content," in *ACM International Conference on Emerging Networking Experiments and Technologies*, 2009, pp. 1–12.

[3] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, July 2012.

[4] W. Li, S. Oteafy, and H. Hassanein, "Rate-selective caching for adaptive streaming over information-centric networks," *IEEE Transactions on Computers*, vol. PP, no. 99, pp. 1–1, 2017.

[5] S. Zhao, Z. Li, D. Medhi, P. Lai, and S. Liu, "Study of user qoe improvement for dynamic adaptive streaming over http (mpeg-dash)," in *2017 International Conference on Computing, Networking and Communications (ICNC)*, Jan 2017, pp. 566–570.

[6] S. Lederer, C. Mueller, B. Rainer, C. Timmerer, and H. Hellwagner, "An experimental analysis of dynamic adaptive streaming over http in content centric networks," in *2013 IEEE International Conference on Multimedia and Expo (ICME)*, July 2013, pp. 1–6.

[7] B. Rainer, D. Posch, and H. Hellwagner, "Investigating the performance of pull-based dynamic adaptive streaming in ndn," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2130–2140, Aug 2016.

[8] J. Samain, G. Carofiglio, L. Muscariello, M. Papalini, M. Sardara, M. Tortelli, and D. Rossi, "Dynamic adaptive video streaming: Towards a systematic comparison of icn and tcp/ip," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2166–2181, Oct 2017.

[9] M. F. Majeed, S. H. Ahmed, S. Muhammad, H. Song, and D. B. Rawat, "Multimedia streaming in information-centric networking: A survey and future perspectives," *Computer Networks*, vol. 125, no. Supplement C, pp. 103 – 121, 2017, softwarization and Caching in NGN. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128617302414

[10] S. Lederer, C. Mueller, C. Timmerer, and H. Hellwagner, "Adaptive multimedia streaming in information-centric networks," *IEEE Network*, vol. 28, no. 6, pp. 91–96, Nov 2014.

[11] S. Lederer, C. Mueller, B. Rainer, C. Timmerer, and H. Hellwagner, "Adaptive streaming over content centric networks in mobile networks using multiple links," in *2013 IEEE International Conference on Communications Workshops (ICC)*, June 2013, pp. 677–681.

[12] S. Petrangeli, N. Bouten, M. Claeys, and F. D. Turck, "Towards svc-based adaptive streaming in information centric networks," in *2015 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, June 2015, pp. 1–6.

[13] Y. Liu, J. Geurts, J. C. Point, S. Lederer, B. Rainer, C. Mller, C. Timmerer, and H. Hellwagner, "Dynamic adaptive streaming over ccn: A caching and overhead analysis," in *2013 IEEE International Conference on Communications (ICC)*, June 2013, pp. 3629–3633.

[14] A. Detti, M. Pomposini, N. Blefari-Melazzi, S. Salsano, and A. Bragagnini, "Offloading cellular networks with information-centric networking: The case of video streaming," in *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2012, pp. 1–3.

[15] M. Conti, R. Droms, M. Hassan, and S. Valle, "Qoe degradation attack in dynamic adaptive streaming over icn," in *IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2018, (In press).

[16] C. Kreuzberger, D. Posch, and H. Hellwagner, "Amust framework - adaptive multimedia streaming simulation framework for ns-3 and ndnsim," 2016.

[17] Y. Liu, S. Dey, D. Gillies, F. Ulupinar, and M. Luby, "User experience modeling for dash video," in *2013 20th International Packet Video Workshop*, Dec 2013, pp. 1–8.

[18] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 1024–1049, Second 2014.

[19] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM CCR*, vol. 44, no. 3, pp. 66–73, 2014.

[20] A. Compagno, M. Conti, and M. Hassan, *An ICN-Based Authentication Protocol for a Simplified LTE Architecture*. Cham: Springer International Publishing, 2018.

[21] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for http video streaming at scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, April 2014.

[22] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "Bola: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, April 2016, pp. 1–9.

[23] C. Sieber, T. Hofeld, T. Zinner, P. Tran-Gia, and C. Timmerer, "Implementation and user-centric comparison of a novel adaptation logic for dash with svc," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013, pp. 1318–1323.

[24] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proceedings of the 3rd Multimedia Systems Conference*, ser. MMSys '12. New York, NY, USA: ACM, 2012, pp. 89–94. [Online]. Available: http://doi.acm.org/10.1145/2155555.2155570

[25] S. Lederer, C. Mueller, and C. Timmerer, "Dynamic adaptive streaming over HTTP dataset," in *Proceedings of the Third Annual ACM SIGMM Conference on Multimedia Systems (MMSys12)*, M. Claypool, C. Griwodz, and K. Mayer-Patel, Eds. New York, NY, USA: ACM, feb 2012, pp. 89–94.

[26] C. Kreuzberger, D. Posch, and H. Hellwagner, "A scalable video coding dataset and toolchain for dynamic adaptive streaming over http," in *Proceedings of the 6th ACM Multimedia Systems Conference*, T. O. Wei, Ed. New York, NY, USA: ACM, mar 2015, pp. 213–218. [Online]. Available: http://concert.itec.aau.at/SVCDataset/

[27] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 326–340, Feb 2014.

[28] S. Akhshabi, S. Narayanaswamy, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptive video players over http," *Image Commun.*, vol. 27, no. 4, pp. 271–287, Apr. 2012.

[29] A. Compagno, M. Conti, P. Gasti, L. V. Mancini, and G. Tsudik, *Violating Consumer Anonymity: Geo-Locating Nodes in Named Data Networking*. Cham: Springer International Publishing, 2015, pp. 243–262.

[30] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, and C. Wood, "Privacy-aware caching in information-centric networking," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, 2017.

[31] F. Zhang, W. He, X. Liu, and P. G. Bridges, "Inferring users' online activities through traffic analysis," in *Proceedings of the Fourth ACM Conference on Wireless Network Security*, ser. WiSec '11. New York, NY, USA: ACM, 2011, pp. 59–70. [Online]. Available: http://doi.acm.org/10.1145/1998412.1998425

[32] M. Liberatore and B. N. Levine, "Inferring the source of encrypted http connections," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 255–263. [Online]. Available: http://doi.acm.org/10.1145/1180405.1180437

[33] S. Salsano, A. Detti, M. Cancellieri, M. Pomposini, and N. Blefari-Melazzi, "Transport-layer issues in information centric networks," in *Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking*, ser. ICN '12. New York, NY, USA: ACM, 2012, pp. 19–24.

[34] G. Carofiglio, M. Gallo, and L. Muscariello, "Icp: Design and evaluation of an interest control protocol for content-centric networking," in *2012 Proceedings IEEE INFOCOM Workshops*, March 2012, pp. 304–309.

[35] S. Arianfar and b. Jörg Ott and Lars Eggert and Pekka Nikander and Walter Wong, year=2010, "A transport protocol for content-centric networks."

[36] P. Ni, R. Eg, A. Eichhorn, C. Griwodz, and P. Halvorsen, "Flicker effects in adaptive video streaming to handheld devices," in *Proceedings of the 19th ACM International Conference on Multimedia*, ser. MM '11. New York, NY, USA: ACM, 2011, pp. 463–472. [Online]. Available: http://doi.acm.org/10.1145/2072298.2072359

[37] "Ndnlp: A link protocol for ndn," 2012. [Online]. Available: https://named-data.net/wp-content/uploads/TRLinkProtocol.pdf

[38] S. Arianfar, P. Nikander, and J. Ott, "On content-centric router design and implications," in *Proceedings of the Re-Architecting the Internet Workshop*, ser. ReARCH '10. New York, NY, USA: ACM, 2010, pp. 5:1–5:6.

[39] S. Lederer, C. Mueller, C. Timmerer, C. Concolato, J. Le Feuvre, and K. Fliegel, "Distributed dash dataset," in *Proceedings of the 4th ACM Multimedia Systems Conference*, ser. MMSys '13. New York, NY, USA: ACM, 2013, pp. 131–135. [Online]. Available: http://doi.acm.org/10.1145/2483977.2483994

[40] "Packet fragmentation in ndn: Why ndn uses hop-by-hop fragmentation (ndn memo)," 2015, https://named-data.net/wp-content/uploads/2015/05/ndn-0032-1-ndn-memo-fragmentation.pdf.

[41] C. Ghali, A. Narayanan, D. Oran, G. Tsudik, and C. A. Wood, "Secure fragmentation for content-centric networks," in *2015 IEEE 14th International Symposium on Network Computing and Applications*, Sept 2015, pp. 47–56.

[42] E. M. T. Yoneda, R. Ohnishi and J. Burke, "Consumerdriven adaptive rate control for real-time video streaming in named data networking," in *Internet Conference 2015*. IC2015, 2015, pp. 23–32.

[43] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, October 2012. [Online]. Available: http://named-data.net/techreports.html

[44] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2: An updated NDN simulator for NS-3," NDN, Technical Report NDN-0028, Revision 2, November 2016.

[45] C. Mueller, S. Lederer, J. Poecher, and C. Timmerer, "Demo paper: Libdash - an open source software library for the mpeg-dash standard," in *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, July 2013.

[46] Y. Liu and J. Y. B. Lee, "A unified framework for automatic quality-of-experience optimization in mobile video streaming," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, April 2016, pp. 1–9.