On-Demand Video Streaming based on Dynamic Adaptive Encrypted Content Chunks

Daniel Posch, Hermann Hellwagner
Institute of Information Technology (ITEC)
Alpen-Adria-Universität (AAU)
Klagenfurt, Austria
e-mail:{firstname.lastname@itec.aau.at}

Peter Schartner
Institute of Applied Informatics, System Security Group
Alpen-Adria-Universität (AAU)
Klagenfurt, Austria
e-mail: p.schartner@syssec.at

***Abstract*—This paper proposes a framework for on-demand video streaming that enables secure and efficient delivery of data towards the end user. Our proposal requires the combined usage of three different technologies. The first one is a recent proposal by Jacobsen et al. [1][2] called Content-Centric Networking (also known as Named Data Networking). It is a network architecture that introduces named data as the most valuable element in the network and divides it into so called content chunks, which are self-identifying and self-authenticating data units. The second concept we utilize derives from the approach of Dynamic Adaptive Streaming over HTTP [3], which allows clients to dynamically choose the quality of the received video stream according to their available resources. Finally, we adapt the concept of Broadcast Encryption [4] to form a tool to control the access to provided content streams. The combination of these technologies enables us to design a framework that allows streaming providers to transport data to customers as dynamic adaptive encrypted content chunks, which is an efficient, flexible and scalable way of multimedia data transport.**

Keywords*—*Content-Centric Networking, Dynamic Adaptive Streaming over HTTP, Broadcast Encryption, Video on Demand.

I. Introduction

Multimedia services for real-time entertainment have become the dominant traffic source in today's Internet. Sandvine [5] claims that 58.6%/49.9% (fixed/mobile access) in America, 47.3%/50.0% in Asia and 34.8%/34.2% in Europe of the traffic in peak times was caused by real-time entertainment in 2012. For instance, in America the biggest single traffic source is Netflix (28.8%), which provides a video on demand service. Netflix heavily depends on Content Delivery Networks to deliver multimedia content to its costumers, since today's Internet lacks content-aware (re-)transmission capabilities. Furthermore, the real-time transport of multimedia is challenging and puts hard demands on a network infrastructure. For this reason different protocols have been developed, including the established Real-Time Transport Protocol (RTP) and newer proposals such as Dynamic Adaptive Streaming over HTTP (DASH) [3] that uses the Hypertext Transfer Protocol for multimedia delivery. On the first glance this seems non-intuitive, since HTTP has been developed for best effort and not for real-time multimedia transport. Nevertheless, there are various good reasons for this choice as argued in [3]. For now, let us briefly discuss what could be the cause for the development of such concepts. The originally designed Internet architecture has not been intended for its current usage. Today its host-based communication paradigm is unhandy, especially when it comes to multimedia delivery. This is definitely one trigger for the recently emerging effort around the topic Future Internet (FI) and its community [6]. A widely discussed proposal for a FI architecture is Content-Centric Networking (CCN) [1][2]. The novel idea of this concept is to establish data as the central element in the network. Hosts communicate via the requests of small content chunks by name rather than requesting data explicitly from another host's address. Since each content chunk is uniquely identified by a name and protected against manipulation by a digital signature, data can be cached inherently in the network. This enables a much more efficient transport of content to multiple receivers, which is the common use case for most multimedia applications.

978-1-4799-1270-4/13/$31.00 © 2013 IEEE

This paper describes a framework for Video on Demand (VoD) services that builds upon CCN with the objective to enhance the transport efficiency of multimedia data. Furthermore, the idea is to combine CCN with the concepts of DASH [3] and Broadcast Encryption (BE) [4]. We expect from this fusion to gain an efficient and scalable way of content delivery to fixed and mobile clients. The support of heterogeneous end devices can be accomplished by using a concept from DASH, which is to offer movies in different qualities. This enables clients with varying resources (bandwidth, buffer status) and hardware capabilities (screen resolution, processing power) to choose the best fitting quality level according to their current situation. BE will be used as a tool for implementing a Digital Rights Management (DRM) functionality, to offer providers the possibility to define the set of possible consumers dynamically. So, it is possible to base a fee-based VoD service on the presented framework, similar as it is currently implemented by big vendors such as Netflix.

The remainder of the paper is organized as follows. Section 2 briefly describes why each of the three technologies (CCN, DASH, and BE) is integrated into the framework and points out the major gain from each of them. Section 3 describes our implementation approach and summarizes the evaluation results relevant to transport efficiency. In Section 4 we briefly discuss open issues of the prototype and present problem-solving approaches. The paper is finally concluded in Section 5 that gives also hints for possible future work.

II. Underlying Concepts and Technologies

This section points out the most valuable properties of CCN, DASH and BE, since their fusion forms the foundation of the proposed framework. We argue why their combination is beneficial for the implementation of a VoD service. For further details on the utilized technologies we refer the reader to the indicated resources.

A. Content-Centric Networking

A major bottleneck of today's Internet architecture is that a packet is strictly transmitted to a single receiver, although it might contain data that is relevant for multiple participants. This unfavourable behaviour is caused by the Internet Protocol (IP), which does not allow to aggregate streams easily even if multiple request of the same content share common source-destination paths. Especially VoD providers are confronted with this disadvantage, since consumer requests of videos are usually Pareto distributed [7]. So, an efficient multimedia delivery framework requires a network architecture that offers an inherent caching feature. Content-Centric Networking [1][2] can exactly fulfil this requirement. This approach splits content into small named chunks. There are two basic types of packets, which are called the Interest and the Content Object (CO). To fetch some data, a host creates an Interest that includes the name of the requested content. The network routes the Interest towards any potential data source(s), which will return the data included in a Content Object if the content exists. The general structure of the packet types is illustrated in Figure 1. As can be seen from the figure, the CO is authenticated with a digital signature. This enables any network participant to cache a data object and pass a legitimate copy to any requesting source. The receiver can be certain that the copy is unaltered and therefore equivalent to the original if the CO's signature is valid. The resulting advantage of the signature is that content can be cached inherently in the network, which enables efficient multimedia data transport to multiple receivers. So, one major reason for resting our framework on the CCN technology is the ability of efficient content delivery. Another point that makes this technology interesting is its support for end user mobility. The framework presented in this paper should allow mobile clients to consume a stream without any interruptions even if they are moving between different networks. The current design of the Internet is heavily location dependent and does not allow this easily. This can be perfectly seen from IP's design principle to coalesce location with identity. An IP address represents an identity in a network. If a mobile client changes its geographical position, it will eventually enter a new network, obtain a new IP address and therefore "change" its identity. For instance, this identity change would result in the loss of all transport connections if the client streams data over the TCP/IP protocol suite. This definitely results in an unsatisfactory and untenable usage experience for mobile users. Note that this example is related to real-world solutions. For instance, mobile clients that stream movies via DASH could encounter this issue. In CCN this problem does not exist, because CCN talks only about data and does not care from which location it can be actually obtained.

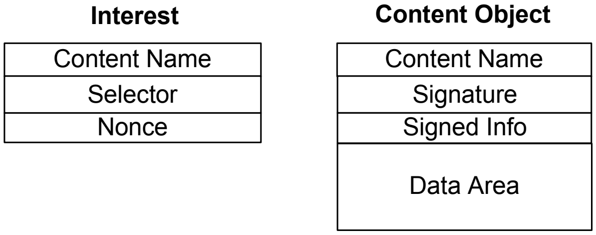

Fig. 1: The basic CCN packet types [1].

B. Dynamic Adaptive Streaming over HTTP

Mobility introduces challenges which cannot successfully be dealt with inside CCN. There are two issues that have to be definitely considered when creating a VoD framework. The first problem is that mobile clients may encounter bandwidth alterations while moving between different networks. The second issue is that mobile clients usually have less resources available than stationary clients. Mobile end-devices have to be energy efficient, their CPU/GPU is usually slow and their display resolution is quite small. The different requirements that heterogeneous clients demand from a VoD service require the service to be adaptive. To integrate this functionality in the proposed framework we investigated DASH [3], which was exactly designed to solve the previously mentioned issues. DASH is a pull-based protocol, which positions the entire logic on the client side. The server is responsible for offering a video in different quality levels. The video is split into segments that can be fetched and displayed independently. The information about the segments is published as a meta description. This piece of information is the so called Media Presentation Description (MPD). It defines which resources and hardware capabilities are needed to stream and playback the video segments of a specific quality level (also called representation). The client uses an integrated adaptation logic that considers certain resource parameters to pull the correct video segments from the best fitting representation to ensure a smooth playback. Several proprietary commercial deployments of this approach including solutions from Adobe [8], Apple [9] and Microsoft [10] show that this technique is perfect to deal with the issue of changing bandwidth conditions and various resource requirements from heterogeneous end-devices. So, the objective of the proposed framework is to fuse the functionality of DASH with the CCN concept.

C. Broadcast Encryption

Until now the concept for the framework supports efficient transport of multimedia data through CCN, and the DASH-like adaptive behaviour enables the usage of heterogeneous mobile end devices. One essential feature is still missing. Consuming the VoD service should be fee-based. From the perspective of the service provider only legitimate clients that pay a fee for the service should be able to access the data stream. This requirement is quite challenging, since one has to consider the following facts. In CCN data is publicly authenticated and inherently cached in the network. Once data is inside the network it can be fetched from any participant without security concerns. A simple encryption of the content may prohibit the access from non-legitimate clients. However, if the data was encrypted separately for each client, the benefits that could be achieved from CCN caching would vanish, since each data packet for each client would be different. Encrypting the content for all clients with a single key is also no option, because there might be traitors in the system. Traitors may publish the key and enable non-legitimate clients to consume the data. Once the key would have been compromised it would require a lot of resources from the VoD provider to recover from this. First, a new key would have to be distributed among all clients, secondly the content would have to be re-encrypted and re-transmitted over the network. Additionally the traitor could not be identified and therefore prevented from publishing the key again. The solution to relax this issue is Broadcast

Encryption [4], which is actually a key management scheme. It enables a sender to transmit messages confidentially to a continuously changing set of receivers in an asynchronous way. In general, Broadcast Encryption works as follows. The sender creates a set of secrets. Each legitimate client gets a small subset from this set of secrets. The sender randomly determines the current session key, which will be used for the ongoing communication to the clients. The session key is ”encrypted” with the set of secrets, so that only a legitimate client with a correct subset of secrets can derive the session key. There are various BE schemes available to implement this functionality [11][12][13]. The usage of BE allows that traitors (participants that publish their secrets or keys) can be traced and excluded from the ongoing communication. To exclude a participant the sender just has to generate a new session key, encrypt it in such a way that the traitor’s secrets cannot derive the session key and distribute it over the network. The disadvantage of content re-encryption still exists, but we see the following three possibilities to relax the problem for the proposed framework:

1) Updates of the session key should be delayed until a set of compromised secrets has been gathered. Temporarily this may enable some of the non-legitimate clients to consume the data stream. However, if the delay is not too long, this should not be a major problem in a home entertainment use case.
2) Updates of the session key should be triggered in non-peak times. Additionally, if the content is dynamically re-encrypted at the same moment it is requested the first time, untapped content will not be re-encrypted and no resources will be wasted.
3) An implementation of the framework could use some cloud-based services to dynamically adapt its resources. Regardless if there is a lack of computation time or bandwidth, the cloud service could be used to overcome the shortage of resources.

So, the presented framework rests on CCN, DASH and BE to create dynamic adaptive encrypted content chunks for multimedia delivery in VoD scenarios. Figure 2 sketches the basic idea of the framework and summarizes the benefits that can be achieved from the utilization of each technology.

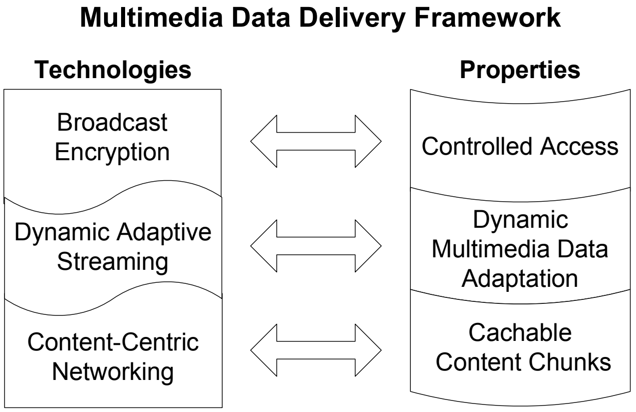
Multimedia Data Delivery Framework
Technologies
Properties
Broadcast Encryption
Controlled Access
Dynamic Adaptive Streaming
Dynamic Multimedia Data Adaptation
Content-Centric Networking
Cachable Content Chunks

Fig. 2: The basic concept of the proposed framework and its advantages.

III. Implementation and Evaluation

In this section we present our prototype that is based on the proposed framework. The prototype has been evaluated and the results will be illustrated. The aim of the evaluation was to determine if a real-world deployment of the framework is realistic and if our expectations regarding efficiency are met.

A. Prototype Software

To examine how well the concept of the framework works, we have implemented a server and a client that enable VoD streaming based on the three previously mentioned technologies. For this purposes we have used CCNx [14], which is a prototype implementation of CCN offered as open-source software by the Palo Alto Research Center. The functionality of CCNx is still limited, but it enables data transfer via content chunks as an overlay over a transport protocol. For our implementation, we have chosen to use CCNx as an overlay over UDP rather than over TCP in order to not encounter TCP-based side effects (retransmission, timeouts, etc.) and to introduce only a minimal overhead.

The client software is based on the VLC media player [15] provided by the open-source community VideoLan and a plugin that has been developed by Liu et al. [16], which provides DASH-based streaming functionality over CCN instead of over HTTP. Only minor changes had to be done on the plugin to serve our purposes. The adaptation logic, which is responsible for the streaming decisions, is kept extremely simple. The client maintains a buffer of 30 seconds. If the buffer drains below 30%, the adaptation logic simply chooses the representation with the lowest possible quality in order to prevent re-buffering events. A re-buffering event occurs if the client’s video buffer is drained and therefore the video playback pauses until enough data is available again. This event must be prevented at all costs, since it is the worst case scenario for the quality of experience (QoE) for consumers. However, if the buffer is filled sufficiently, the next segment will be chosen according to the average download speed of all previously downloaded segments.

The server component has been developed from scratch using the C++ framework Qt. The server is responsible for offering the DASH-encoded video segments and the MPDs to the clients. We have integrated a very basic Broadcast Encryption scheme into the client and the server. The server uses the scheme to transmit the current session key to all clients. The key can be used by the client to decrypt the encrypted video segments provided by the server.

B. Evaluation Set-up

For the evaluation we have used the following test set-up as illustrated in Figure 3. Two clients are connected via a common CCN router to the streaming server. The router is serving as a network inherent cache, which should allow both clients to benefit from increased bandwidth if both are requesting the same content. The bandwidth between the server and the router is limited to 6 Mbit/s and an RTT of 30 ms is introduced to simulate a bottleneck. The connection between the clients and the router is fast with properties of a typical local area network. We have evaluated the maximum goodput (throughput of application data) that can be achieved from the server to the clients. It is roughly 5 Mbit/s. The reason for this is the overhead introduced by the CCNx protocol and the underlying protocols, which is according to our measurement about 15% (depending on the transferred data).

For the evaluation of our prototype, we prepared a DASH-encoded dataset with the tool DASH-Encoder provided by Lederer et al. [17]. We encoded a 720p video of 200 seconds length into twelve representations ranging from 200 kbit/s up to 5000 kbit/s. The segment duration has been set to two seconds, which results in exactly 100 segments per representation. For all evaluations presented in the next subsection the server has been configured to update every 100 seconds the session key and the clients always stream the same video.

C. Evaluation Results

We have done several measurements and we present three of them, because they summarize the advantages and open problems of the framework best. For the first evaluation, both clients were started exactly at the same time. The streaming results are depicted in Figures 4 and 5 and they are promising. The figures show that both clients can stream the 4100 kbit/s representation of the video, although they share a common bottleneck of 6 Mbit/s. The advantage of our framework is perfectly highlighted. At the beginning the clients start to stream a low quality representation to fill up their buffers quickly. The adaptive features quickly react once the buffers are filled and segments with increased quality are chosen. The multimedia transmission is efficient, since both clients can be served from the cache of the intermediate CCN router.

Since we cannot expect clients to start streaming at the same time, the open question that remains is how the prototype implementation reacts when clients start to stream at different points in time. The following two measurements illustrate our observation regarding this question best. Figures 6 and 7 depict the streaming result when the second client is started exactly one second after the first one. According to the previous result, we expect both clients to eventually stream the segments of the 4100 kbit/s representation. However, as can be seen from the figures, the first client plateaus at the 3600 kbit/s representation, while the second one is rapidly switching between the 3600 kbit/s and the 4100 kbit/s representations. The reason for this behaviour is the following. Since the second client requests each segment a little bit later than the first one, it can benefit especially at the beginning from already fully or partially cached video segments. Therefore, its measured goodput is higher than the measured goodput of the first client. So, the second client's adaptation logic decides as soon as its buffer level is higher than 30% to download a segment from the quite high representation with 3100 kbit/s. Actually the bandwidth is not available, which leads to draining its sparsely filled buffer below the critical level forcing the client to stream the lowest representation until the level is reached again. Once it is reached, the second client once more adapts and increases the representation level. A moment later the second client decides first to stream the 4100 kbit/s representation, because its average measured goodput is over 4100 kbit/s. Since the other client is still streaming the 3600 kbit/s representation, the 6 Mbit/s of available bandwidth is too low for both. This results for both clients in measuring a small goodput. Therefore, the second client immediately drops back to the 3600 kbit/s representation, while the first client's buffer is absorbing the temporary bandwidth reduction and enables the client to keep streaming the 3600 kbit/s representation. During the remaining streaming process the second client continually tries to switch to a higher representation, because it actually measures a large enough goodput. This results in a kind of oscillating behaviour, switching between quality levels frequently, which is very bad for a user's QoE.

The third measurement illustrates another issue. This time the second client has been started about 15 seconds after the first one. As can bee seen from Figures 8 and 9 the resulting streaming performance is really poor. The clients choose different representations for streaming, resulting for both of them into draining a few times below the critical buffer level. Finally, each of them plateaus on a different low quality representation instead of synchronizing to gain the possibility to stream a high quality representation.

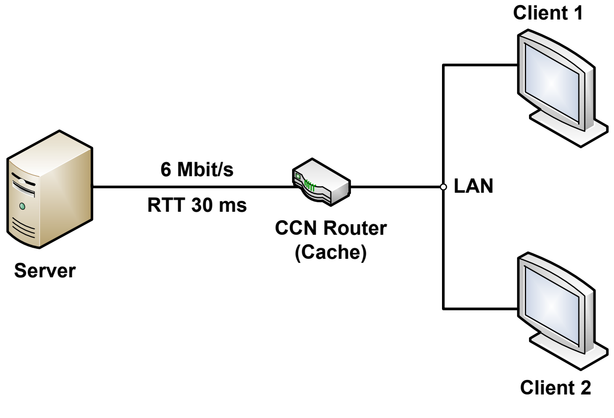

Fig. 3: Test set-up for the evaluations.

IV. Open Issues and Potential Solutions

As indicated in Section 3, the issues we encountered arise due to unfavourable streaming decisions in the client's adaptation logic. The first issue (oscillation) can be explained by the client's ignorance about the cache in the CCN router. As previously mentioned, the client's adaptation logic continuously determines the current goodput and estimates according to this series of values the correct representation quality for the downloaded video segments. However, the adaptation logic walks into the trap of cached and therefore readily available data. This leads to overestimating the actual available bandwidth. So, the solution to the first problem could be to extend the client's adaptation logic. For this an additional feature is needed that detects if the received content is cached or not, and to consider this fact when determining the best fitting representation. Actually Müller et al. encountered a similar issue with the combined usage of DASH and HTTP proxies in [18]. One solution they propose is called exponential backoff with probing. The idea is that every time the client's adaptation logic tries to switch to a higher quality it has to probe the actual available bandwidth by downloading a small part of the next video segment in the targeted quality. If this measurement returns at least the expected bandwidth, the client will switch to the higher quality. Otherwise, it can be assumed that the currently streamed representation is cached somewhere near the client and therefore switching to a higher representation would be a drawback. To prohibit extensive probing an exponential backoff for the switching attempts is

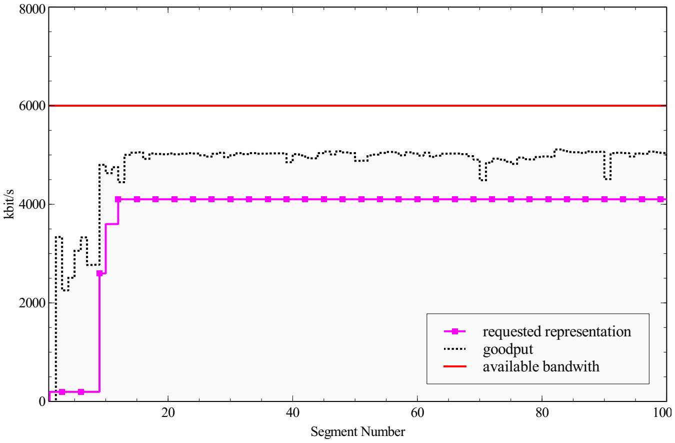

Fig. 4: Experiment 1: Result for the first client.

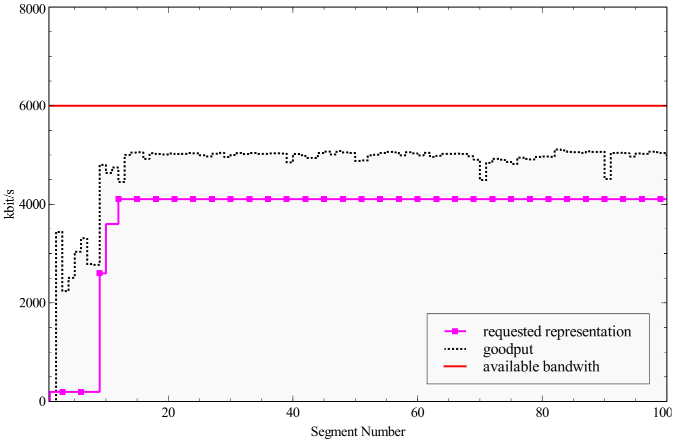

Fig. 5: Exp. 1: Result for the second client (simultaneous start).

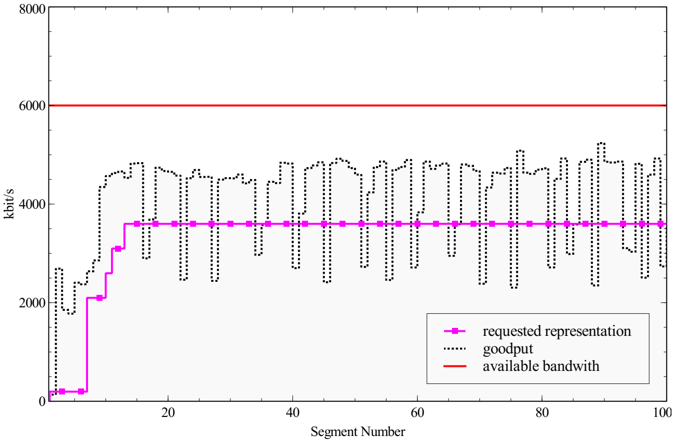

Fig. 6: Experiment 2: Result for the first client.

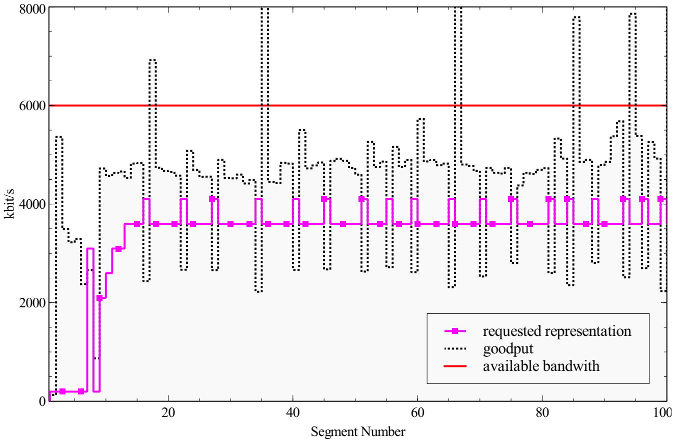

Fig. 7: Exp. 2: Result for the second client (1s delayed).

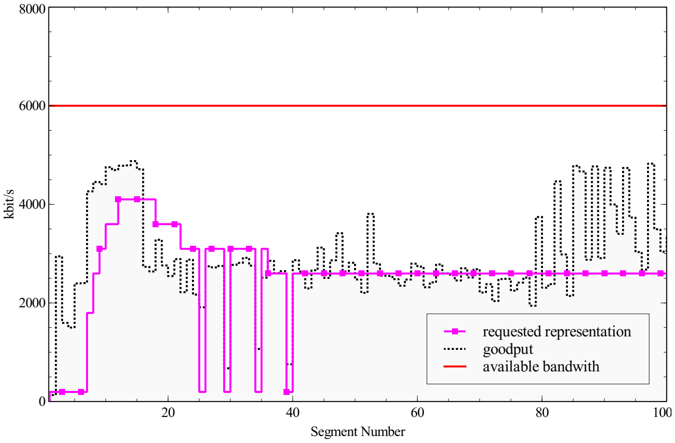

Fig. 8: Experiment 3: Result for the first client.

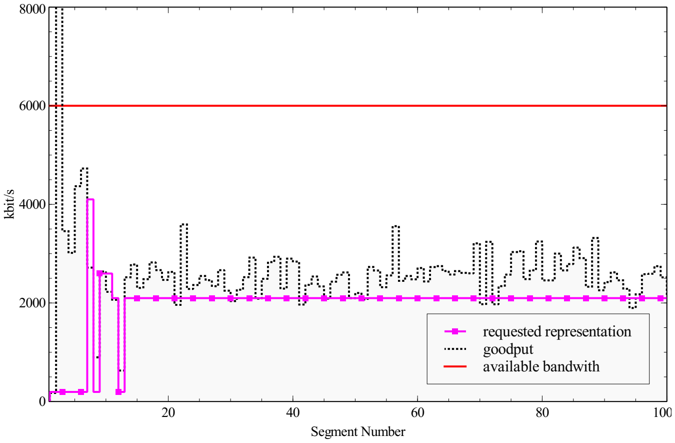

Fig. 9: Exp. 3: Result for the second client (15s delayed).

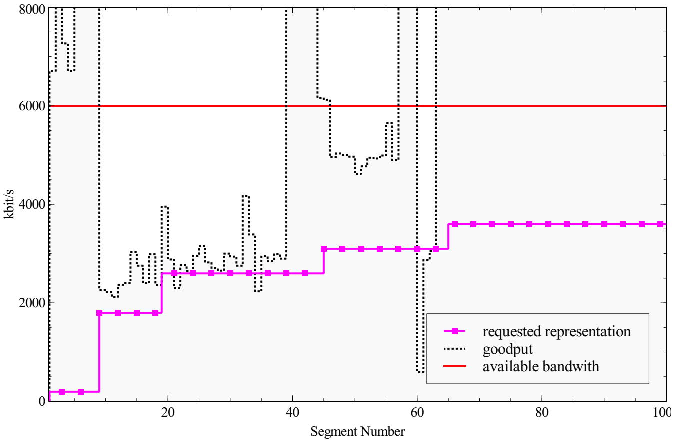

Fig. 10: Exp 4: Probing prevents switching to a higher quality. Result for the second client (10s delayed).

used to minimize resource wastage. We have integrated this concept into the client's adaptation logic and could eliminate the oscillation behaviour observed in Figure 7. The result is shown in Figure 10. Although the client measures a very high goodput rate it never attempts to switch up. Probing tells the client that the measured bandwidth is actually not available and is pretended by nearby cached content. For explanation, the sudden drop of the goodput between the segment 60 and 63 is caused by the periodic update of the session key. Therefore, cached data becomes evicted (times out) from the inherent network caches and since the first client in this test run is slightly ahead (10 seconds) with playback the second client has to fetch these segments from the origin server.

The second issue (poor streaming performance) emerges because clients are not aware of each other. This causes them to compete for the available bandwidth rather than working together to find the optimal representation regarding their quality of experience. Our framework lacks a way to synchronize the clients' streaming decisions and therefore this issue remains unsolved. However, we will work on this issue by trying different approaches of adaptation logics. For instance, we are aiming to investigate completely buffer-based solutions.

V. Conclusion and Future Work

In this paper we have proposed a framework for multimedia delivery in VoD use cases. We combined the concepts of CCN, DASH and BE in order to create dynamic adaptive encrypted chunks of data, which can be inherently cached in the network. The evaluation results show that network inherent caching can definitely increase the efficiency of multimedia delivery. However, the usage of adaptive concepts leads to the question of how to synchronize clients to exploit the advantage of cached data perfectly. Finding a solution to this issue would enhance the framework greatly. So, future work should definitely deal first with this lack of synchronization by investigating new approaches for a more sophisticated adaptation logic for the client software. Furthermore, different approaches of BE could be examined to evaluate their performance in conjunction with the proposed framework with respect to traitor tracing and secret propagation.

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '09. New York, USA: ACM, 2009, pp. 1–12, Online: *http://doi.acm.org/10.1145/1658939.1658941*.

[2] V. Jacobson, L. Zhang, D. Estrin, J. Burke, J. Thornton, D. Smetters, B. Zhang, G. Tsudik, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, "Named Data Networking (NDN) Project." PARC Technical Report NDN-0001, October 2010, Online: *http://named-data.net*.

[3] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP: Standards and Design Principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11. New York, USA: ACM, 2011, pp. 133–144, Online: *http://doi.acm.org/10.1145/1943552.1943572*.

[4] A. Fiat and M. Naor, "Broadcast Encryption," in *Advances in Cryptology CRYPTO 93*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1994, vol. 773, pp. 480–491, Online: *http://dx.doi.org/10.1007/3-540-48329-2_40*.

[5] Sandvine, "The Global Internet Phenomena Report," 2012, Online: *http://www.sandvine.com/*.

[6] EC FIArch Group, "Fundamental Limitations of Current Internet and Path to Future Internet," March 2011, Online: *http://tinyurl.com/FIArch*.

[7] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '07. New York, USA: ACM, 2007, pp. 1–14, Online: *http://doi.acm.org/10.1145/1298306.1298309*.

[8] "Adobe HTTP Dynamic Streaming," Online: *http://www.adobe.com/products/hds-dynamic-streaming.html*.

[9] R. Pantos, "HTTP Live Streaming," *Appel Inc.*, 2013, Online: *http://tools.ietf.org/html/draft-pantos-http-live-streaming-11*.

[10] A. Zambelli, "IIS Smooth Streaming Technical Overview," *Microsoft Corporation*, Online: *http://tinyurl.com/IISSmothStreaming*.

[11] C. K. Wong, M. Gouda, and S. S. Lam, "Secure Group Communications Using Key Graphs," in *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '98. New York, USA: ACM, 1998, pp. 68–79, Online: *http://doi.acm.org/10.1145/285237.285260*.

[12] D. Naor, M. Naor, and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," in *Advances in Cryptology CRYPTO 2001*, ser. Lecture Notes in Computer Science, J. Kilian, Ed. Springer Berlin / Heidelberg, 2001, vol. 2139, pp. 41–62, Online: *http://dx.doi.org/10.1007/3-540-44647-8_3*.

[13] D. Halevy and A. Shamir, "The LSD Broadcast Encryption Scheme," in *Advances in Cryptology CRYPTO 2002*, ser. Lecture Notes in Computer Science, M. Yung, Ed. Springer Berlin / Heidelberg, 2002, vol. 2442, pp. 145–161, Online: *http://dx.doi.org/10.1007/3-540-45708-9_4*.

[14] Palo Alto Research Center, "Project CCNx," 2009, Online: *http://www.ccnx.org/*.

[15] VideoLan Organization, "VLC Media Player," Online: *http://www.videolan.org/*.

[16] Y. Liu, J. Geurts, B. Rainer, S. Lederer, C. Müller, and C. Timmerer, "DASH over CCN: A CCN Use-Case for a Social Media Based Collaborative Project," in *CCNx Community Meeting (CCNxConn 2012)*. Sophia Antipolis: PARC, September 2012.

[17] 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, USA: ACM, 2012, pp. 89–94, Online: *http://doi.acm.org/10.1145/2155555.2155570*.

[18] C. Müller, S. Lederer, and C. Timmerer, "A Proxy Effect Analysis and Fair Adaptation Algorithm for Multiple Competing Dynamic Adaptive Streaming over HTTP Clients," in *Proceedings of the IEEE Conference on Visual Communications and Image Processing Conference (VCIP 2012)*, K. Aizawa, J. Kuo, and Z. Liu, Eds. San Diego, CA, USA: IEEE, Nov 2012, Online: *http://www-itec.uni-klu.ac.at/bib/files/PID2500949.pdf*.