AN EXPERIMENTAL ANALYSIS OF DYNAMIC ADAPTIVE STREAMING OVER HTTP IN CONTENT CENTRIC NETWORKS

Stefan Lederer, Christopher Mueller, Benjamin Rainer, Christian Timmerer, and Hermann Hellwagner
Alpen-Adria-Universität Klagenfurt
Universitätsstraße 65-67
9020 Klagenfurt am Wörthersee, Austria
{firstname.lastname}@itec.aau.at

ABSTRACT

This paper presents the usage of CCN, which is a candidate for the next-generation Internet, in combination with the new Dynamic Adaptive Streaming over HTTP (DASH) standard, which was recently ratified by ISO/IEC MPEG. In contrast to the Internet Protocol, which is mainly based on the host-to-host connection paradigm originated in the 1970s, Content Centric Networking (CCN) focuses on the content itself, instead of its location. Considering the dominance of multimedia traffic in todays' Internet, the streaming performance of DASH over CCN as well as the problems introduced by this combination is worth to be investigated in detail. Therefore, we evaluate the protocol overhead introduced by the usage of CCN compared to the HTTP versions 1.0 and 1.1. Furthermore, the performance of DASH over CCN under different network conditions is compared to the performance of HTTP 1.0/1.1. Our results showed that although CCN comes together with higher protocol overhead than HTTP 1.0/1.1 as well as a prototype implementation, it can definitely compete with HTTP 1.0 in media streaming. Based on the evaluation results, problems as well as improvement possibilities are identified, which are the basis for future work in this area.

Index Terms – MPEG-DASH, CCN, Dynamic Adaptive Streaming over HTTP, Content Centric Networking, Evaluation

1. INTRODUCTION

A variety of new Internet architectures have been proposed in the last decade [1] and some of them seem to overcome the current limitations of todays' Internet. One of these new Internet architectures is the Content Centric Network approach [2], which moves the focus of traditional end-to-end connections to the content, rather than on addressing its location, i.e., devices in a network. CCN could eventually replace IP in the future, but it is also possible to deploy it on top of IP. In comparison to IP, where clients set up connections between each other to exchange content, CCN is directly requesting the content without any connection setup. This means that a client, which wants to consume content, simply sends an interest for this content into the network and the network responds with the corresponding content, wherever it may be located. Additionally, CCN is meant to provide security und trust as an integral part of the network.

From a content perspective, multimedia is omnipresent in the Internet, e.g., producing 58% of the total Internet traffic in North Americas' fixed access networks [3]. Especially the adaptive delivery of multimedia content over HTTP is gaining more and more momentum, which resulted in the standardization of MPEG's Dynamic Adaptive Streaming over HTTP (DASH) [4]. As CCN is a promising candidate for the Future Internet (FI) architecture, it is worthwhile to investigate its suitability in combination with multimedia streaming standards like DASH. Considering that CCN and DASH have several elements in common like, e.g., the client-initiated pull approach as well as the content being dealt with in pieces, this can be a potentially good combination.

The purpose of this paper is to present the delivery of DASH-based multimedia content over CCN and to give a comprehensive evaluation thereof with respect the different existing HTTP features. As DASH is designed for the delivery over HTTP, we describe different options how to deliver DASH-based content over CCN. The actual evaluation thereof comprises two parts. First, we evaluate the overhead in terms of headers and protocol-related communication introduced by replacing HTTP with CCN as transport protocol for DASH-based content. Second, we compare the delivery performance of DASH-based content over CCN in comparison to HTTP. Previous evaluations [5] have shown a significant impact of different HTTP features on the effective media throughput. Therefore, we also investigate HTTP 1.0 as well as mature 1.1 features such as HTTP request pipelining and persistent TCP connections. However, one has to consider that HTTP in its different versions is already well proven in practice and the implementations as well as the underlying protocol stack are highly optimized nowadays. The experimental results of the DASH over CCN overhead and performance offer deep insights, which are the basis to identify promising improvements of the protocol as well as the resulting streaming performance, as presented in this paper.

The remainder of the paper is organized as follows. Section 2 describes related work and Section 3 provides a brief overview of CCN. Section 4 demonstrates how MPEG-DASH can be integrated into CCN. The evaluation concerning the overhead is provided in Section 5 and Section 6 comprises the actual performance evaluation. Finally, Section 7 concludes the paper.

2. RELATED WORK

Concepts around Named Data Networking exist already since 1970 [1]. Nevertheless, CCN received a lot more attention in the past few years, encouraged by the implementations like [6]. However, the number of publications related to CCN and adaptive multimedia streaming is still limited.

In [7], authors provide an implementation and evaluation of Voice over IP (VoIP) for CCN, demonstrating that CCN is very well suited for real-time multimedia data, that it comes together with an improved availability as well as reliability, and that the computational overhead introduced by encryption or signing of the data can be neglected. However, protocol overhead and streaming performance was not covered by them. The authors in [8] demonstrate the usage of Apples' HTTP Live Streaming (HLS) using CCNx [6], but without investigating the streaming performance or the adaption of the stream to, e.g., bandwidth

Acknowledgments: This work was supported in part by the EC in the context of the ALICANTE (FP7-ICT-248652), SocialSensor (FP7-ICT-287975), and QUALINET (COST IC 1003) projects and partly performed in the Lakeside Labs research cluster at AAU.

conditions, like it is done in our paper. They propose a CCN video streaming application which acts as a proxy between the HLS client (i.e., VLC) and the CCNx implementation. In our paper, we describe another possibility which is detailed in Section 4. Live streaming over CCN on a mobile client is described in [10] and also compared against HLS. However, this comparison is focusing on the use case when multiple clients are accessing the same content, where CCN clearly demonstrates its advantages over HTTP (without considering existing HTTP proxy and caching infrastructures).

3. PRINCIPLES OF CCN

Direct communication between hosts as well as the sharing of hardware resources of hosts fits the use cases of the early years of the Internet, but todays' needs and communication schemes have changed, where everything is about the content and its ubiquitous access. TCP/IP was not designed for such tasks, but various approaches have been developed to circumvent the mapping of content to specific machines like, e.g., peer-to-peer (P2P) networks or content delivery networks (CDN) in combination with intelligent Domain Name Service (DNS) resolution.

Jacobson et al. [2] present their next generation networking approach called Content Centric Networking (CCN), which focuses on the content the user requests, rather than the interconnection between hosts. In the CCN approach, there exist only two types of packets: *interest* and *data*. The former are used for requesting the content whereas the latter are used for their actual delivery. The maximum payload of a data packet is 4096 bytes and also referred to as a CCN chunk. Data packets are handled efficiently on the network nodes, e.g., to satisfy consolidated interest packets originating from multiple clients and, thus, providing implicit support for multicast and caching of data packets on CCN nodes within the delivery network.

The interest packet addresses the content by name and may contain further selection information. For example, in case a user wants to watch a movie from example.com, the interest packet contains a content name as a Uniform Resource Identifier (URI) such as *ccnx://example.com/videos/movie.mpg*, which is also the basis for routing in the network. The packet could then be sent over multiple interfaces, e.g., Ethernet, 3G, and WiFi simultaneously. The CCN nodes will forward the interest packet based on a longest prefix matching scheme which is also used in IP routing.

4. INTEGRATION OF DASH AND CCN

The basic concept of DASH [4] is to use segments of media content, which can be encoded at different resolutions, bitrates, etc. as so-called representations. These segments are served by conventional HTTP Web servers and can be addressed via HTTP GET requests from the client. As a consequence, the streaming system is pull-based and the entire streaming logic is located on the client, which makes it possible to adapt the media stream to its capabilities. In addition to this, the content can be distributed using conventional CDNs and their HTTP infrastructure, which also scales very well. In order to specify the relationship between the contents' media segments and the associated bitrate, resolution, and timeline, the Media Presentation Description (MPD) is used, which is a XML document.

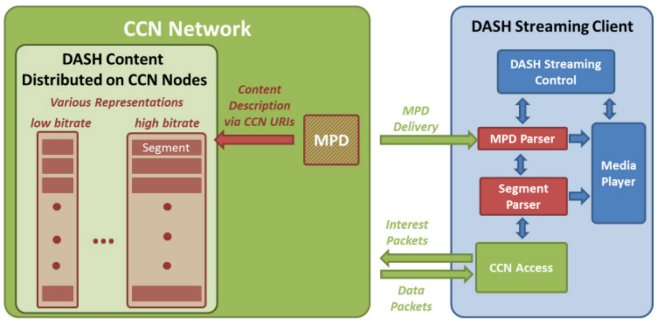

Figure 1: DASH over CCN

In contrast to CCN, the technology behind DASH is already well advanced and large-scale deployments of comparable proprietary media streaming solutions are available by major industry players (e.g., Adobe, Apple, Microsoft). DASH is intended to enable adaptive streaming, i.e., each content piece can be provided in different qualities, formats, languages, etc. to cope with the diversity of todays' networks and devices. As this is an important requirement for Future Internet proposals like CCN, the combination of those two technologies seems to be obvious. Since those two proposals are located at different protocol layers – DASH at the application and CCN at the network layer – they can be combined very efficiently to leverage the advantages of both and potentially eliminate existing disadvantages.

In principle, there are two options to integrate DASH and CCN: a proxy service acting as a broker between HTTP and CCN as proposed in [8], and the DASH client implementing a native CCN interface. The former transforms an HTTP request to a corresponding interest packet as well as a data packet to an HTTP response, including reliable transport as offered by TCP. This may be a good compromise to implement CCN in a managed network [9] and to support legacy devices. As such a proxy is already described in [8] we are focusing on a more integrated approach, aiming at fully exploiting the potential of CCN. That is, a native CCN interface within the DASH client, which adopts a CCN naming scheme (CCN URIs) to denote segments in the Media Presentation Description (MPD). Figure 1 presents the proposed architecture of DASH over CCN where DASH-related components are marked in red, CCN-related components in green, and implementation-depended components in blue. As one can see, only the network access component on the client has to be modified and the segment URIs within MPD have to be updated according to the CCN naming scheme.

Initially, the DASH client retrieves the MPD containing the CCN URIs of the content representations including the media segments. The naming scheme of the segments may reflect intrinsic features of CCN like versioning and segmentation support as presented in [7] and shown in Figure 2. Such segmentation support is already compulsory for multimedia streaming in CCN and, thus, can also be leveraged for DASH-based streaming over CCN. The CCN versioning can be adopted to signal different representations of the DASH-based content, which enables an implicit adaptation of the requested content to

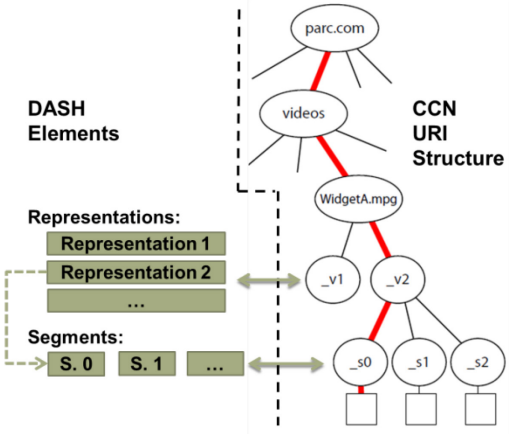

Figure 2: CCN Naming Structure [2]

the clients' bandwidth conditions. That is, the interest packet already provides the desired characteristics of a segment (such as bit rate, resolution, etc.) within the content name. Additionally, if bandwidth conditions of the corresponding interfaces or routing paths allow so, DASH media segments could be aggregated automatically by the CCN nodes, which reduces the amount of interest packets needed to request the content. However, such approaches need further research, specifically in terms of additional intelligence and processing power needed at the CCN nodes.

After requesting the MPD, the DASH client will start to request particular segments. Therefore, CCN interest packets are generated by the CCN access component and forwarded to the available interfaces. Within the CCN, these interest packets leverage the efficient interest aggregation for, e.g., popular content, as well as the implicit multicast support. Finally, the interest packets are satisfied by the corresponding data packets containing the video segment data, which are stored on the origin server or any CCN node, respectively. With an increasing popularity of the content, it will be distributed across the network resulting in lower transmission delays and reduced bandwidth requirements for origin servers and content providers respectively.

5. OVERHEAD ANALYSIS

The first evaluation comprises the protocol overhead due to packet headers and the protocol-related communication. Therefore, the theoretical lower bound is calculated followed by a practical evaluation using our evaluation network with different bitrate versions of DASH-based content.

HTTP is using TCP on the transport- and IP on the network-layer. This introduces an overhead of 20 bytes for the TCP header (+ 12 bytes for the optional header fields) [11] and another 20 bytes for the IP header [12]. If Ethernet [13] is used on the link layer, an additional 14 byte frame header is added to the TCP/IP packets. As Ethernet restricts the Maximum Transportation Unit (MTU) to 1500 bytes, the lower bound of the TCP/IP protocol overhead can be calculated as follows: Considering the resulting maximum payload of the TCP packet of 1448 bytes and the Ethernet frame size of 1514 (incl. Ethernet frame header) this results in an overhead of 4.56 % caused by headers. Additionally, one has to consider packets needed for TCP connection establishment and ACKs, as well as other Ethernet-related overhead like check sequence etc. On top of TCP, HTTP introduces further overhead for requesting and delivering DASH segments. This is depending on the length of the requested URL and the parameters used by the client and the server, which are approx. 150 bytes for the HTTP GET request and approx. 200 bytes for the HTTP response header. As this overhead is fixed regardless of the requested segment size, its influence on small segments is higher than on large DASH segments. The size of a DASH segment results from two parameters: the segment length in seconds and the quality of the chosen representation. Considering DASH segments with a length of two seconds, as used in the following practical evaluation, a segment of the 100kbps representation would have an HTTP-related overhead of 1,37 %, resulting in an total overhead caused by headers of 5,93 %. In the case of a segment of the 4500kbps representation the HTTP-related overhead is only 0,03 %, resulting in an total overhead caused by headers of 4,59 % for those kinds of segments.

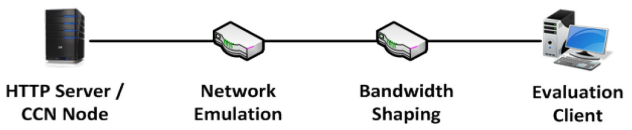

Figure 3: Evaluation Setup.

Basically, one has to consider that CCN is designed as a replacement of IP. However, as today's Internet is based on IP, CCN can be used on top of UPD/IP as well as TCP/IP in order to be compatible to today's infrastructure. Hence, for the following considerations we are using CCN on top of UPD/IP, as it is done by [2]. A UDP header of 8 bytes [14] is added to each CCN packet which is fragmented into IP packets based on the MTU (1500 bytes). This results in a UDP/IP-related lower bound of 2.47% protocol overhead. On this basis, data packets with a maximum payload of 4096 bytes and a header of approx. 550 bytes are transmitted. Each of those data packets is requested by an interest packet with sizes from approx. 150 to 250 bytes which causes a combined overhead of 18.31%. When considering the underlying UDP/IP overhead, the total overhead caused by headers is 20.78%. Additionally, one has to consider the overhead caused by retransmission of lost packets which is more expensive in CCN, due to the 4096 chunk payload.

In the following evaluations the protocol overhead produced by HTTP as well as CCN is investigated in practice to give a comparison to the calculated lower bound.

5.1 Methodology and Evaluation Setup

The protocol overhead is calculated based on the sum of bytes transferred in the network divided by the sum of media bytes received at the clients' application, as shown in Equation 1.

$$Overhead = \frac{\sum bytes\ transmitted}{\sum bytes\ of\ media\ content} - 1 \qquad (1)$$

The practical overhead evaluation is based on our evaluation setup depicted in Figure 3 using different server/client components depending on the used protocols. The network emulation and bandwidth shaping nodes are not used for these experiments and the network is just limited by the 1Gbps network connection between the nodes. However, they are necessary for the next experiments in Section 6. All nodes are based on the same hardware and are running Ubuntu Linux 12.04. The produced traffic is analyzed using Wireshark (http://www.wireshark.org/) for HTTP and using the CCNx Wireshark plugin [6] for CCN.

We conduct three different experiments delivering DASH-based content with HTTP 1.0, HTTP 1.1, and CCN:

Experiment 1 evaluates DASH with HTTP 1.0 [15]. Therefore, we use an Apache HTTP 1.0 Web server providing DASH content. On the client side we use the DASH VLC plugin [16].

Experiment 2 evaluates DASH with HTTP 1.1 [17]. We use again the DASH VLC plugin and Apache Web server, but enable persistent connections and pipelining according to HTTP 1.1.

Experiment 3 evaluates DASH with CCN [2]. We use a CCNx [6] node in the version 0.6.1 with a repository containing the same DASH content as used in Experiments 1 and 2, but with an MPD comprising CCN URIs instead of HTTP URIs. On the client side we use a modified version of the DASH VLC plugin, implementing a native interface to the ccnd daemon of CCNx.

For each experiment we use the Big Buck Bunny DASH content of [5] with a segment length of two seconds. We evaluate the overhead separately for the following representations: 100, 350, 700, 1300, 2800 and 4500 kbps.

5.2 Evaluation Results

The results are depicted in Figure 4 showing that the CCNx implementation maintains a relatively constant overhead of 23.5-23.8% for representations greater than 100kbps. For the 100 kbps representation the overhead of around 24.7 % is slightly higher

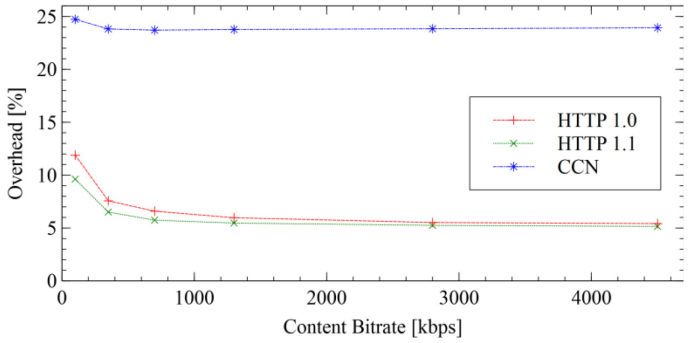

Figure 4: Protocol Overhead Analysis

which is caused by small segments comprising only a few data packets. As the last data packet of a DASH segment tends to not fully utilize the packet's payload capacity of 4096 bytes, the influence of the CCN header for those packets gets bigger in comparison to packets fully utilizing the payload capacity, which causes the higher overhead for the lower bitrate representations.

The measured overhead of the practical evaluation differs from the calculated lower bound of 20.78%. The approx. 3% additional overhead is caused by a suboptimal behavior of the CCNx ccnd daemon, i.e., the last data packet of a DASH segment is signaled by the *FinalBlockID* field of the signed information which is part of the data packet, but unfortunately only the last chunk of a segment contains this information. As the interest packets are requested in a pipelined manner, the ccnd daemon requests non-existing data packets until it receives and processes the data packet with the *FinalBlockID* field.

In contrast to CCN, HTTP 1.0 and 1.1 maintain a relatively low overhead. The overhead of HTTP 1.0 is starting from 11.89% (100 kbps representation) and decreases to 5.42% (4500 kbps rep.). When HTTP 1.1 is used, the overhead is slightly lower, starting from 9.62% for the 100 kbps representation and decreases to 5.15% for the 4500 kbps representation. In both cases, HTTP 1.0 and 1.1, one can see the influence of the fixed size HTTP header, which causes a higher overhead when small segment sizes are used, like in the case of the 100 kbps representation. The smaller overhead of HTTP 1.1 is caused by the usage of efficient features like one persistent TCP connection and the pipelining of HTTP requests. The measured overheads of 11,89 % and 9,63 % respectively are significant higher than the theoretical lower bound of 5,93 %, which comes from ACKs, not fully utilized TCP/IP packet payloads caused by the small segments and from the packets needed for connection establishment for each segment in the case of HTTP 1.0. When the segment size gets bigger, e.g., for the 4500 kbps representation, the influence of the HTTP header decreases and the overhead is very close to the theoretical lower bound of 4.59%.

6. PERFORMANCE ANALYSIS

In this section, we evaluate the performance in terms of transmitted media data of the dynamic adaptive streaming over HTTP 1.0, HTTP 1.1, and CCN. The performance is measured for each protocol by comparing the effective media bitrate received and the buffer level at the client. The evaluations are performed in the same manner for all three protocols and are based on a predefined track of bandwidth variations. As different network delays influence the performance of the different protocols, we carry out these evaluations using different round trip times (RTT) ranging from 0 to 150ms.

6.1 Evaluation Setup and Methodology

For these evaluations, the network emulation setup in Figure 3 is used. For bandwidth shaping, the Linux Traffic Control system (tc) is used in combination with a Hierarchical Token Bucket (htb) packet scheduler using Statistical Fair Queuing (sfq). The network emulation node controls the RTT using the Linux Network Emulator (netem). Each experiment uses the Big Buck Bunny DASH content from the dataset in [5], which is encoded at 14 different bitrates from 100 to 4500 kbps and has a segment length of 2 seconds. The available bandwidth, depicted as dashed line in the Figure 5 and Figure 6, is equal for all experiments and changes every 20 seconds.

We used parts of the metrics from [19] for our experiments. The average media bitrate is retrieved by the client, which could be seen as the overall performance indicator of the system. Furthermore, the buffer level represents the current fill state of the buffer of the DASH client, which gives additional information about the adaptation performance of the system.

6.2 Evaluation Results

The results for the average media bitrate and the buffer level are provided for RTT = 0 and 150 ms in the following result graphs, separated for (a) HTTP 1.0, (b) HTTP 1.1, and (c) CCN. The term "Bandwidth" indicates the available bandwidth which is predefined as introduced above. The term "Media Bitrate" indicates the requested bitrate of the segments and, thus, the playback bitrate at the client.

Figure 5 shows the results for RTT = 0 ms. In terms of the requested media bitrate, there are only a few differences between the used protocols, e.g., noticeable between seconds 20 and 40 as well as between seconds 125 and 145, where lower bitrates are requested in the case of CCN in comparison to the other protocols.

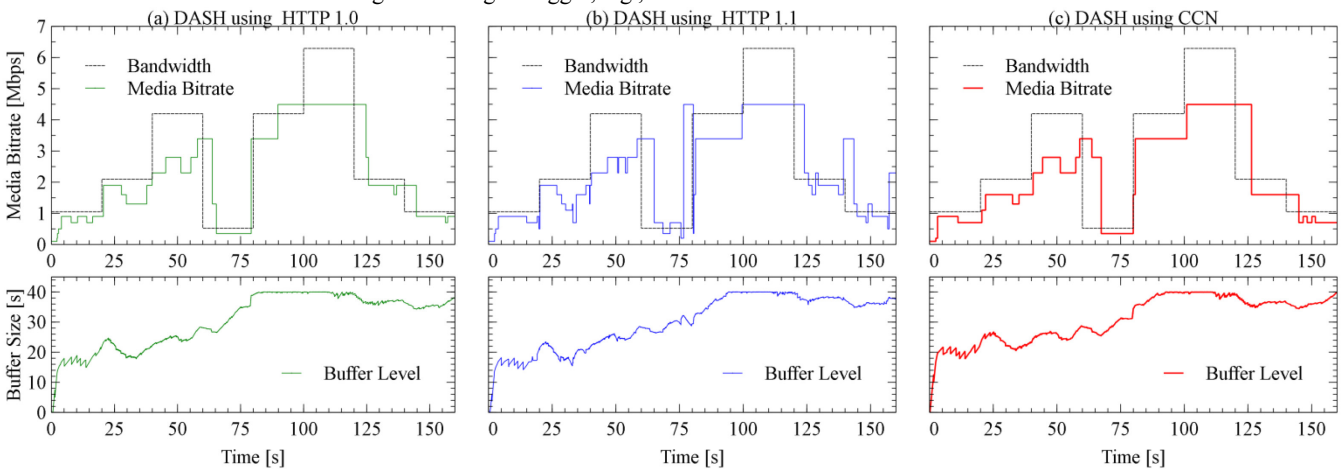

Figure 5: Performance Evaluation with RTT = 0 ms.

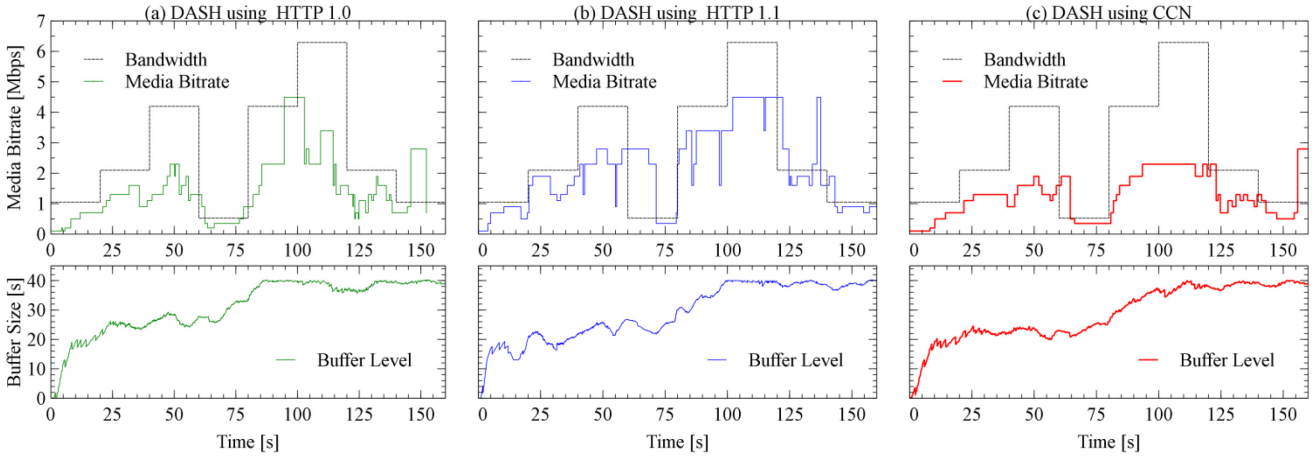

Figure 6: Performance Evaluation with RTT = 150 ms.

Additionally, all three protocols maintain a very similar buffer fill pattern, which is caused by the same behavior of the DASH adaption logic in all three experiments. Nevertheless, there are noticeable differences, as one may see between seconds 0 and 50 where CCN consistently has a higher buffer fill level than the other systems. HTTP 1.0/HTTP1.1 are relatively sensitive to variations in the size of the DASH segments, as they are caused by variations of the encoding bitrate due to different scene complexity in the content. This results in the higher amount of bitrate switches in Figure 5 (a) and (b). In contrast to this, DASH over CCN in (c) is relatively constant which is caused by the fixed overhead, as already shown in Figure 4, and the usage of UDP instead of TCP.

Figure 6 shows the results for RTT = 150 ms. Although CCN comes together with a higher overhead, it achieves a better performance than HTTP 1.0 with a 1% higher average bitrate. However, when comparing the buffer fill levels one may see that CCN has more problems to reach the maximum buffer level. In these evaluations, one clearly sees the advantages of HTTP 1.1 in Figure 6 (b), which still maintains nearly the same average media bitrate as in the other experiment with lower RTT. It also adjusts much faster to the available bandwidth conditions than HTTP 1.0 and CCN, as one can notice at the beginning of the session between seconds 0 and 10 as well as 80 and 85. This lower performance of HTTP 1.0 is caused, by the lower link utilization due to the two RTTs needed for TCP connection establishment and HTTP GET request for each DASH segment. With higher RTT in Figure 6, CCN has more and more problems reaching the maximum buffer level of 40 seconds of content. This is caused by the relative high header overhead of CCN data packets and a poor link utilization of CCN at high RTTs.

Considering that CCN is a new as well as experimental concept and the used CCNx implementation is a prototype and not integrated, e.g., into the system's kernel like TCP is, the overall performance of DASH over CCN in terms of average media bitrate is relatively good as shown in Figure 7. The difference of 117 kbps, or 6 %, to HTTP 1.0 and of 219 kbps, or 11 %, to HTTP 1.1 in the case of 0 ms RTT is lower than expected, especially when considering the previously shown protocol overhead as well as the computational overhead at the CCN nodes, introduced by cache lookup, bloom filters, etc. Furthermore, the performance of DASH over CCN decreases monotonically in contrast to conventional DASH over HTTP 1.0, which finally leads to a 1 % better performance of CCN than HTTP 1.0 when the RTT increases to 150 ms. However, the CCN performance at this high network delay is still 735 kbps, or 39 % lower than HTTP 1.1.

The performance drop of DASH over HTTP 1.0 results from establishing a new TCP connection for each segment as well for requesting the segments via a subsequent HTTP GET request. The used TCP connection is closed again after the HTTP response. This also influences the bandwidth utilization, since the TCP slow start uses the default initial congestion window of the system for each connection establishment. Furthermore, two times the RTT are needed for establishing the TCP/IP connection and sending the HTTP request for the desired segment until the first bytes of it are received, which is also responsible for the lower bandwidth utilization.

The experimental results showed the good performance of HTTP 1.1 in environments with high RTT, like in mobile networks. When the RTT is increased, the reduction of the average bitrate for HTTP 1.1 is significantly lower than for HTTP 1.0 and CCN, which is caused by the usage of persistent TCP connections and pipelining in HTTP1.1. In this case the client does maintain one TCP connection for the whole streaming session and requests the subsequent DASH segment while downloading the previous one. Due to this, the downlink is always fully utilized, which causes the good performance with high RTTs. Furthermore, the TCP connection window is initialized only at the beginning of the streaming session, i.e., for the first segment

According to these results, the current CCN implementation has the potential to compete with DASH over HTTP 1.0, however, it definitely needs some work to come closer to the performance of HTTP 1.1. Therefore, the high overhead and the lower link utilization have to be addressed. In particular, the link utilization of DASH over CCN is influenced directly by the network delay. This influence can be reduced by improving the pipelining of CCN interest packets on the transport layer, as it is done by TCP. Furthermore, the performance can be increased by eliminating unnecessary interest packets at the end of the data

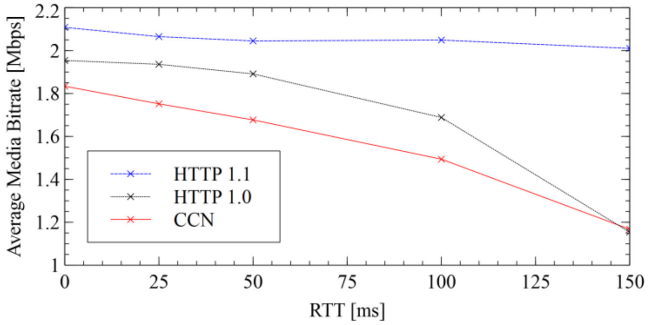

Figure 7: Average Media Throughput for different RTTs

transfer of a DASH segment, which was already mentioned in the overhead evaluation in Section 5. The interest packets are requested in a pipelined manner, so non-existing data packets are requested while the last data packet containing the *FinalBlockID* field is received and processed by the client. These unnecessary interest packets can be easily avoided by sending the *FinalBlockID* field in an earlier data packet, e.g., the first one of the transfer, to notify the requesting node which is the last data packet. Of course, this also effects the bandwidth utilization in networks with higher delays, as instead of sending unnecessary interest packets, the client can already request data packets of the subsequent DASH segment and, therefore, reduce the time in which the link is unused. Additionally, research has to focus on a more efficient possibility for content encryption and signing which is currently mainly responsible for most of the header size and as a consequence also for the protocol overhead.

7. CONCLUSIONS

This paper proposed and evaluated the combination of CCN with DASH. As both concepts maintain several elements in common, like, e.g., the content in different versions being dealt with in segments, a deep integration of DASH and CCN can be achieved. Out of several implementation possibilities, a direct integration in the DASH client has been proposed and evaluated in different experiments focusing on dynamic multimedia streaming against the same client based on HTTP 1.0 and HTTP 1.1.

The first evaluations analyzed the protocol overhead introduced by CCN and its chunk-based data retrieval in comparison to HTTP 1.0 and HTTP 1.1. As CCN separates data to fixed size chunks of 4 kB, identified by URIs and equipped with signing information, the protocol overhead is significantly higher than in the case of HTTP 1.0/1.1, where the overhead caused by TCP/IP and the HTTP headers is relatively low in comparison. Nevertheless, the CCN header information enables further possibilities which are not possible with IP-based protocols such as HTTP, e.g., the automatic retrieval of content via the fastest available link with an intrinsic error resilience w.r.t. the network, as show in our previous work in [21].

Additionally, we evaluated the dynamic adaptation characteristics and the performance in terms of media throughput for all solutions. This has been conducted based on a given test scenario with different bandwidth variations, which has been analyzed under different network delays. We showed that streaming adaptive media using DASH on top of CCN is possible. As expected, CCN cannot compete with HTTP 1.1 and its efficient usage of one TCP connection for the whole streaming session as well as the pipelining of HTTP requests. However, HTTP 1.1 is not supported by every Web server and causes problems in combination with proxies like, e.g., the Head-of-Line blocking where a range of responses can be delayed by, e.g., only one pending response [20]. DASH over CCN can definitely compete with HTTP 1.0, showing the same RTT-sensibility characteristics and slightly outperforming it already in scenarios with high network delay. Considering the prototype implementation of CCN and the modification possibilities proposed in this paper, DASH over CCN has the possibility to outperform HTTP 1.0 and strive towards the efficiency of HTTP 1.1., while maintaining its advantages like, e.g., its efficient caching and intrinsic multicast support.

8. REFERENCES

[1] J. Pan, S. Paul, and R. Jain, "A Survey of the Research on Future Internet Architectures", In *IEEE Communications Magazine,* Vol. 49, Issue 7, 26 – 36, 2011.

[2] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content", In *Proc. of the 5th Int. Conf. on Emerging Netw. Experiments and technologies* (CoNEXT '09), ACM, New York, NY, USA, 1-12, 2009.

[3] Sandvine, *Global Internet Phenomena Report 1H 2012,* Sandvine Intelligent Broadband Networks, 2012.

[4] ISO/IEC DIS 23009-1.2, *Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats*

[5] S. Lederer, C. Mueller and C. Timmerer, „Dynamic adaptive streaming over HTTP dataset", In *Proc. of the 3rd Multimedia Systems Conf.* (MMSys '12). ACM, New York, NY, USA, 89-94, 2012.

[6] CCNx Projects V. 0.6.1, URL: http://www.ccnx.org (last access: Dec. 2012).

[7] V. Jacobson, D. Smetters, N. Briggs, M. Plass, P. Stewart, J. Thornton and R. Braynard, "VoCCN: voice-over content-centric networks", In *Proc. of the 2009 Works. on Re-architecting the Internet* (ReArch '09), ACM, New York, USA, 1-6, 2009.

[8] 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 *Proc. of the Int. Symp. on a World of Wireless, Mobile and Multimedia Networks* (WoWMoM '12), IEEE, San Francisco, CA, USA, 1-3, 2012.

[9] D. Perino and M. Varvello, "A Reality Check for Content Centric Networking", In *Proc. of the 10th Int. Conf. on Networks* (ICN '11), Canada, Toronto, 44-49, 2011.

[10] Z. Chen, R. Chen and J. Cao, "Live Streaming with Content Centric Networking", In *Proc. of the 3rd Int. Conf. on Netw. and Distributed Computing*, Hangzhou, China, 2012.

[11] *Transmission Control Protocol*, URL: http://tools.ietf.org/html/rfc793(last access: Dec. 2012),

[12] *Internet Protocol*, RFC 791, URL: http://tools.ietf.org/html/rfc791(last access: Dec. 2012)

[13] *IEEE 802.3-2800 Ethernet*

[14] J. Postel, *User Datagram Protocol*, RFC 768, URL: http://tools.ietf.org/html/rfc768 (last access: Dec. 2012).

[15] T. Berners-Lee, R. Fielding, R. and H. Frystyk, *Hypertext Transfer Protocol -- HTTP/1.0,* URL: http://www.w3.org/Protocols/HTTP/1.0/spec.html (last access: Dec. 2012)

[16] C. Mueller and C. Timmerer, "A VLC Media Player Plugin enabling Dynamic Adaptive Streaming over HTTP", In *Proc. of the 19th ACM Int. Conf. on Multimedia* (ACM MM '11). ACM, New York, NY, USA, 723-726, 2011

[17] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", URL: http://www.w3.org/Protocols/rfc2616/rfc2616.html (last access: Dec. 2012).

[18] K. J. Grinnemo, T. Andersson, A. Brunstrom, "Performance Benefits of avoiding head-of-line blocking in SCTP," *In Proceedings of ICAS/ICNS*, pp.44, Tahiti, 2005.

[19] S. Lederer, C. Müller, B. Rainer, C. Timmerer, and H. Hellwagner, "Adaptive Streaming over Content Centric Networks in Mobile Networks using Multiple Links", in *Proc. of the IEEE Int. Workshop on Immersive & Interactive Multimedia Comm. over the Future Internet*, Budapest, Hungary, June, 2013.