Analyzing Caching Benefits for YouTube Traffic in Edge Networks - A Measurement-Based Evaluation

Lothar Braun, Alexander Klein, Georg Carle
{braun,klein,carle}@net.in.tum.de
Chair for Network Architectures and Services
Technische Universität München

Helmut Reiser
reiser@lrz.de
Leibniz Supercomputing Center

Jochen Eisl
jochen.eisl@nsn.com
Nokia Siemens Networks

***Abstract*—Recent studies observed video download platforms which contribute a large share to the overall traffic mix in today's operator networks. Traffic related to video downloads has reached a level where operators, network equipment vendors, and standardization organizations such as the IETF start to explore methods in order to reduce the traffic load in the network. Success or failure of these techniques depend on caching potentials of the target applications' traffic patterns.**

Our work aims at providing detailed insight into caching potentials of one of the leading video serving platforms: YouTube. We monitored interactions of users of a large operator network with the YouTube video distribution infrastructure for the time period of one month. From these traffic observations, we examine parameters that are relevant to the operation and effectiveness of an in-network cache deployed in an edge-network. Furthermore, we use our monitoring data as input for a simulation and determine the caching benefits that could have been observed if caching had been deployed.

I. Introduction

The past few years have seen the rise of web services and user generated content. HTTP traffic has become one of the dominant protocols in current networks, both seen from a large Internet wide view [1] as well from a local networks' point of view [2], [3]. One flavor of user generated content that has come to a special interest of network operators and network researchers is video content. Video platforms such as YouTube, are known to be responsible for a large share of the overall amount of traffic exchanged in certain networks. A recent study from Labovitz et al. [1] quantifies this share to be as large as 20–40% of all HTTP traffic. With YouTube being one of the most popular video sharing applications, researchers conducted a lot of work that aims at understanding the platform and its impact on network traffic.

Over-the-top (OTT) video services such as YouTube are challenging for network operators, as these contribute a large amount of traffic in their (access) networks. Furthermore, video traffic is expected to experience a strong increase for mobile device usage [4]. As video content is typically static, and video popularity on platforms like YouTube is assumed to be Zipf distributed (except for the tail) [5], [6]. Thus, video content is seen to be a good candidate for caching. However, several properties of YouTube traffic might have negative impact on cache hit rates and cache performance: One important factor is video popularity. Global popularity of YouTube videos, measured in view counts, must not necessarily match local peculiarities in a specific operator network [7]. Popularity distributions may differ slightly depending on the network that a cache has to serve.

Factors such as video encodings or user behavior can impact cache performance, too. YouTube introduced several video encoding formats for its videos starting in 2007. Due to these different encodings, otherwise equal content is transformed into a complete different (as seen from a caches' perspective) video. Users can switch between encodings while watching videos, they can abort video download before completion, or can start watching a particular video from a certain position in the video file.

Our work aims at quantifying the caching potential of video traffic in end-networks. We picked the YouTube video platform for our study because it is one of the major platforms for user generated video distribution, and has therefore received a lot of attention from researchers, operators and network equipment vendors. For our work, we monitored and analyzed all YouTube video traffic from an end-network with more than 120,000 users over a period of a month. We examine this traffic with respect to relevant parameters for network caches, and report our findings. Building on this traffic evaluation, we simulate an in-network cache for this network. We estimate the caching potential and the reduction of downstream network traffic that can be avoided because of such a cache.

The remainder of this paper is organized as follows. Following this introduction, Section II presents related work that studies YouTube or caching of video data. In addition, we show where our work extends and complements previous studies. Section III introduces the parts of the YouTube application that are relevant for understanding our work. We explain interactions between clients and the YouTube video servers which have impact on the operation of a video cache. Section IV presents our traffic monitoring setup and introduces the network in which our vantage point has been deployed. The obtained data sets span a period of one month and several general properties are discussed. Afterwards, we discuss several properties of this data sets which are relevant for caching, and present our evaluation on the benefits of an in-network video cache in Section V. Our paper is concluded in Section VI with a summary of our results and an outlook on future work.

II. Related Work

Related work can be grouped into several categories: Some papers discuss on the shares of YouTube traffic in the overall traffic mix. Others focus on YouTube traffic characteristics, YouTube's infrastructure, or caching of video content.

Popularity of YouTube videos has been studied from different view points. One branch of papers use active crawling of YouTube sites to determine popularity or try to find reasons for popularity of various videos [8]. Figueiredo et al. [9] focus on popularity development over time. Their findings conclude that a lot of videos show a *viral* popularity growth, indicating potentials for caching. Others find power-law patterns with truncated tails in global video popularity distributions and predict good caching potentials [5], [10].

Video popularity has also been studied from network local perspectives [11], and local and global video popularity have also been compared [7]. These studies show that the global popularity of video content does not have to match the local popularity. For example, Gill et al. [7] find that the Top 100 videos from global YouTube video rankings are viewed in their network but do not have any significant contribution to the overall amount of YouTube traffic observed. Caching strategies must therefore consider local popularity and view counts.

Other work tries to provide a better understanding of the YouTube web application or YouTubes' infrastructure Such work includes attempts to inspect YouTubes' policies for picking local data centers for video downloads [12], or describe load-balancing or traffic asymmetry from the view point of a Tier-1 provider [13]. Finamore et al. [14] assess YouTube traffic by evaluating several YouTube traces from different networks. They study traffic and video patterns for YouTube videos from mobile and PC-based devices, and show differences between traffic of these devices.

Caching potentials have been considered by Ager et al. for different protocols in [15] where they outline good potentials for caching HTTP in general. Zink et al. evaluate caching of YouTube videos [11]. In their study, the authors collect three one-week traces from a university network and use these as input for cache simulation. The authors consider client, P2P and proxy caches and estimate cache video hit rates. They conclude that high video hit rates can be achieved even with small caches. We extend their work by accounting more important factors such as video encoding formats, aborted video downloads and their impact on caches. Furthermore, we do not only consider video hit rates, but more sophisticated metrics such as content hit rates. Using these metrics, we can show that other caching strategies, such as chunk-wise caching strategies, provide better cache performance than the previously proposed and evaluated caching.

III. YouTube Explained

The YouTube video application has been studied in a variety of previous work. Our discussion of the YouTube video application considers those parts of the application that have direct impact on caching and are relevant for understanding our work. For a more detailed and complete description of the YouTube video application, we refer the reader to [14].

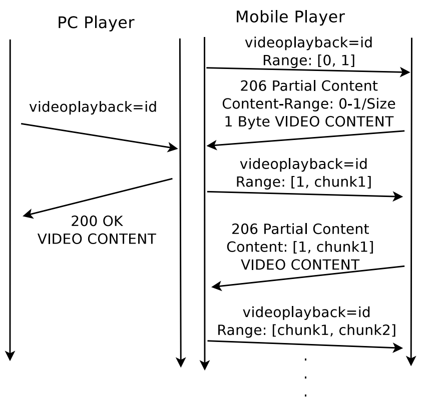


Fig. 1: Video downloads from PC and mobile players

When talking about YouTube, one has to be aware that there is not a single web application for delivering videos, but two different ones. The first application targets PC systems, the other is optimized for mobile devices [14]. Both behave different on the network level during video download phase, but share common behavior before downloading a video.

Before a video can be requested from a video server, a user has to become aware of this video. Users can find videos from the YouTube portals sites (www.youtube.com for PCs or m.youtube.com for mobile devices) or via other websites which embed a video within an iframe. The portal or the embedded iframe contain a reference to a video in form of a unique video ID, which is then requested by the player.

Video download differs significantly between mobile- and PC-based devices. As first difference, PC players most often display videos using the Adobe Flash Plugin, which does not exist on several mobile devices such as the iPhone. It is also possible to watch videos directly using an HTML5 capable browser. However, this feature is currently only in a trial phase and users need to explicitly opt-in in order to download videos using HTML5. Mobile devices, which connect to YouTube from a WiFi network, usually do not have access to the Adobe Flash Plugin. Instead, they request and obtain a MP4 container (or similar) which includes the video directly.

The most important difference, however, is the process of downloading a video. Figure 1 shows downloading procedures for a PC and a mobile player. PC players, which are expected to have access to much memory and/or disk space, download a video using a single HTTP connection. Mobile players on the other hand download a single video using multiple connections where each connection is used to download a different chunk. HTTPs chunking mechanism is used to implement this process. Chunks sizes are chosen by the client in the HTTP header, and servers reply with a *Partial Download* HTTP response. The first chunk size is always a request for a single byte of the video. Video servers respond with the first video byte along with the total size in bytes of the video. The client will then request subsequent video chunks of bigger sizes to download the video. Video encoding and start of the video are also requested by the client, which applies to both the PC and the mobile player.

IV. YouTube Traffic Data Sets

This section builds the base for our YouTube traffic study. We introduce our monitoring setup that we used to observe and analyze YouTube traffic in in Section IV-A. Afterwards, we describe general properties of our obtained data sets in Section IV-B.

A. Monitoring Setup

Our vantage point was deployed in the *Munich Scientific Research Network* (Münchner Wissenschaftsnetz, MWN) in Munich, Germany.The research network interconnects three major universities and several affiliated research institutions in the area in and around Munich, Germany. Furthermore, the network includes several student dorms that provide housing for the students enrolled in the universities in Munich. In total, the network hosts about 80,000 devices which are used by approximately 120,000 users. The *Leibniz Supercomputing Center* (Leibniz-Rechenzentrum, LRZ), who operates this network, provides Internet access for all its users via a 10 GBit/s link to its upstream provider the *German research network (DFN)*. Our vantage point was deployed on the border gateway between the MWN and its upstream service provider. Due to this deployment, we were able to observe both office related as well as residential video traffic.

Our monitoring setup was built around standard of the shelf PC hardware, operated by a Linux-based operating system. All traffic properties where calculated during an online monitoring run, as we where not able to store the many Terrabytes of YouTube traffic that were observed during our monitoring period. We used an optimized capturing setup, including our improvement presented in [16], based on TNAPI [17] to build a multi-core aware monitoring system. The measurement was conducted with the tstat [18] tool, which has been used for monitoring YouTube traffic before [14].

B. Data Set Properties

The monitoring setup was used to log information about the YouTube video downloads for a period of one month. We collected this data in order to be able to measure long-term statistics of caching relevant parameters.

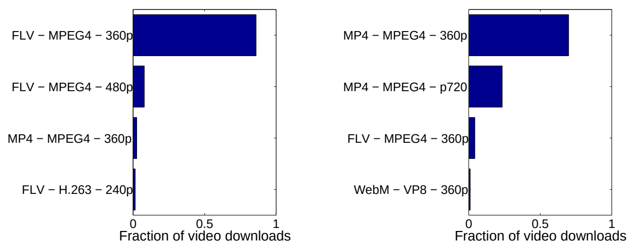
Table I describes the data set obtained throughout the monitoring process. The measurement was started in mid-July and was continually observing all video downloads until mid-August. We decided to distinguish between PC player and mobile player traffic, as shown in the table. Similar to [14], we use HTTP return codes for distinguishing between PC player and mobile players: Video requests from PC players are answered with a HTTP *200 OK* return code, while mobile video requests are answered by *206 Partial Content*.

Mobile downloads are only responsible for a small share of the overall video traffic (1.6 TB for mobile downloads vs. 40.3 TB for PC downloads) in our network. However,

TABLE I: Monitoring Data Overview

Property	Value
Start time	16-Jul-2011 12:57:33 UTC
End time	15-Aug-2011 13:47:10 UTC
PC Player	
# of video downloads	3,727,753
# of video IDs	1,235,676
# of videos with single encoding	1,129,548
# of videos with multiple encodings	106,128
Video traffic (PC player)	40.3 TB
Mobile Player	
# of download connections	2,480,703
# of video IDs	73,601
# of videos with single encoding	70,388
# of videos with multiple encodings	3,213
Video traffic	1.6 TB

mobile downloads are responsible for quite a large number of connections. This is due to the download procedure which downloads a single video via multiple connections, as described in Section III. We did not log the byte range requests from the clients requesting the actual chunks. Thus, we cannot give precise numbers about how many video downloads have been seen.



(a) PC player (b) Mobile Player

Fig. 2: Video encodings for PC and mobile traffic

One interesting difference between mobile and non-mobile traffic can be found in the video encoding statistics, as shown in Figure 2. Most watched videos on a PC platform are transmitted as MPEG-4 AVC (H.264) encoded video with a 360p resolution which is embedded into a flash container. Mobile videos are usually not embedded into flash containers, but are downloaded in a MP4 container. As for video content, the same encoding is used with a 360p resolution.

Hence, if the same video is watched with a mobile and a PC-based player, there is a high probability that a cache needs to deliver a completely different video (from a cache's point of view) for the same requested video. Operator networks that provide network access to an equal amount of mobile and PC-based devices, might therefore have to cache a lot of videos twice due to different encodings. Networks that have only one dominant type of video requests, such as the network we are looking at, might thus employ a smaller cache.

We were curious about the content types which have been requested most often by the users and therefore examined the most often viewed videos from PC players. The biggest share of the most popular videos where advertisements. Seven of TOP 10 videos can be placed in this category, with most of them being short ads for computer games. One particular popular video (Top 2) is part of a campaign advertising for a German pay TV station. We think that these videos were embedded into non-YouTube related sites and automatically downloaded by users who have visited those sites. The remaining two Top 10 videos are a news clip from CNN and a short fun movie. Advertisements or trailers for computer games, movies, cell phones, or beer dominate the Top 30 of the most often viewed videos. Each of these videos has a view count of more then 1500 views, and most of them are clips with a run time of less than two minutes. In the following section, we discuss several parameters that are relevant for caching this video traffic.

V. Evaluation

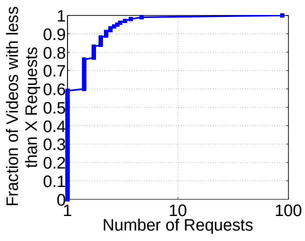
This section discusses video properties of the data obtained in the previous section. Relevant parameters for caching are discussed in Section V-A. Section V-B evaluates the benefits of such an in-network cache. For the sake of brevity and due to the fact that PC player traffic is dominant in our observations, we will restrict our further discussion to PC player traffic.

A. Relevant Videos Parameters for In-Network Caching

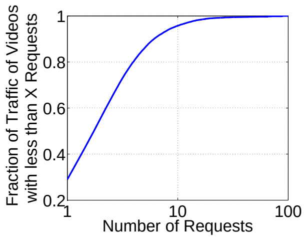
There are several important factors of video traffic that have large impact on a cache. These properties include video sizes, number of views of a single video or the inter-request times of multiple views, as caches can only provide benefit if videos or parts of videos are watched more than once. We distinguish between videos from a caches' point of view: Two videos are considered to be different if they have a different YouTube video id, or if they share the same video id but are encoded in different formats. In the following, we use the term video to address unique video content. Furthermore, the term request corresponds to a partial or full request of a video, while the term view indicates a full download of a video.

Figure 3a presents the share of videos out of the observed videos that have a particular number of requests. Our data reveals that about 60% of all videos are only requested once in our monitoring interval which results in an average number of 2.7 requests per video. The remaining 40% of the videos can be cached and delivered from the cache to other clients for subsequent requests. The majority of the videos have been requested ten or less times, but some of the videos are watched several hundred or even thousand times.

The huge share of videos that are viewed only once or a couple of times could lead to the assumption of only little potential for caching. However, if traffic volumes are considered, different trends can be observed: Figure 3b plots the amount of video content delivered from the YouTube servers for videos that have less than a certain amount of views. The majority of videos that have been requested only once are responsible for only approximately 30% of the video traffic. Videos that are watched more than once account for



(a) Requests per video



(b) Overall traffic depending on the number of requests

Fig. 3: Video request counts and their impact on generated traffic

the biggest part of the traffic, which emphasizes the potential of in-network caches.

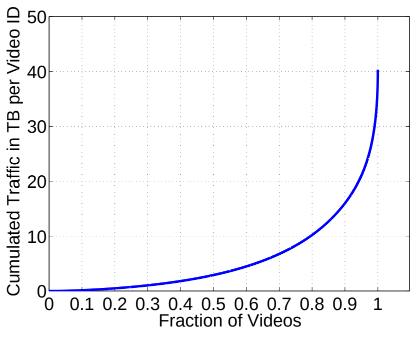


Fig. 4: Total requested data of all videos

Figure 4 summarizes the influence of individual videos on the overall amount of download traffic. The graph shows the sum of the requested data per video sorted by traffic size in order to outline the traffic contribution of the individual videos. One can see that 80% of the videos are responsible for about 10 TB of traffic, while the remaining 20% account for 30 TB of all downloaded video data. By identifying and caching such high-profile videos, a cache can significantly decrease the amount of download traffic and achieve good cache hit rates without the need of a large storage capacity, since 20% of the videos generate 75% of the video traffic.

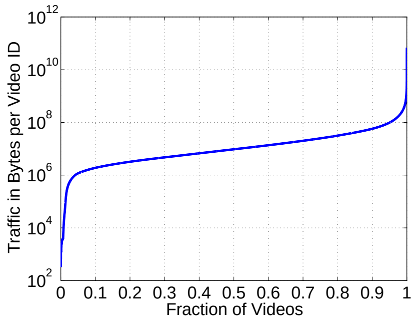


Fig. 5: Total requested data per video

If we focus on individual videos, we can recognize a similar trend. Figure 5 plots the amount of traffic per video sorted by traffic size. The amount of traffic is summed over all requests of the video. The plot indicates that a very small fraction of videos only contribute a very small share to the overall amount of video data. These videos were probably watched only for a single time and/or were aborted before being downloaded completely. 4.2% of the videos' download sizes are less than 1MB in data, while 4.9% of the video downloads generated more than 100 MB of traffic. Thus, more than 90% of the videos generate between 1 and 100 MB of traffic. The linear increase of the traffic load of these videos points out that their generated load can assumed to be uniform distributed.

Request size and number of requests per video are the two factors that contribute to the amount of traffic that is generated by a single video. Very large videos are responsible for a large amount of traffic even if they are watched only a few times. Small videos that are viewed very often can also accumulate a lot of traffic over a large time interval. From a caches' point of view, videos with high request rates are most beneficial for hit rates and cache efficiency. Moreover, high request rates in combination with large request sizes would further reduce traffic from the YouTube servers which results in a decrease of link load between the edge network and the Internet. Table II lists the amount of traffic and the number of views by the TOP 5 videos which generated the most download traffic[1].

TABLE II: TOP 5 Videos

TOP	Traffic (GB)	Request Count	videoID
1	68.3	7758	hJd9iCbpwuI
2	33.3	4204	zM41GVYYOMI
3	16.4	22	roFmDA2_yhg
4	16.3	2826	60ZO8fVkfH4
5	14.5	4944	Wsfgyyvs1tc

As one can see, the characteristics of the Top 5 videos in terms of generated traffic load and number of views differs significantly. The heterogeneity of the videos is indicated by the number of requests, e.g. the video that is ranked third is only requested 22 times but contributes 16.4 GB of traffic whereas the video on fifth place is requested 4944 times. The same characteristics can be recognized for the top 20 videos in terms of requests which we cannot list due to space limitations. However, the number of requests is still a reasonable decision factor whether a video should be cached or not since the average request size of all videos is 12.8MB.

Besides the amount of generated traffic per video, request patterns play an important role for the efficiency of a cache. Now, we take a closer look on the average inter-request time of the videos and the time period during which the videos were requested. Due to the heterogeneity of the videos, we decided to evaluate both characteristics for different groups of videos. The videos are grouped according to the number of requests which reflects their popularity. The following intervals were used to group the videos $[2, 10[$, $[10, 100[$, $[100, \infty[$. These groups are referred to as low, medium and high popular videos, respectively. Figure 6 reveals that the average inter-request time of videos differs significantly depending on the number of requests. 95% of the average inter-request-times

[1]View video at http://www.youtube.com/watch?v=$\$videoID$

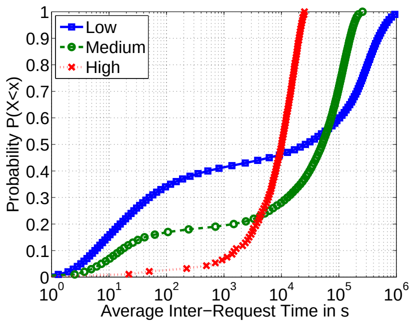


Fig. 6: CDF of average inter-request time depending on the number of requests

of high popular videos are between 1000 and 10000 seconds which is a strong contrast to those with less requests. The videos with low or medium popularity have a much higher variance of their inter-request times. Their results show that the majority of these videos have an average inter-request time of approximately 30 seconds. We assume that this represents a typical value for non-popular videos, which are posted in social networks. Thus, it is likely that friends will request the posted video resulting in a couple of full or partial downloads within a rather short period of time. In addition, a second peak can be recognized for videos with less than 100 requests for an average inter-request time of approximately one day which results in a bimodal distribution.

Another important characteristic for caches is represented by the request period which is the time difference between the first and the last request of a video. According to our definition, the maximum of the request period is limited by the monitoring period. The cumulative distribution function of the request period of videos with low, medium and high popularity are shown in Figure 7. The figure points out that

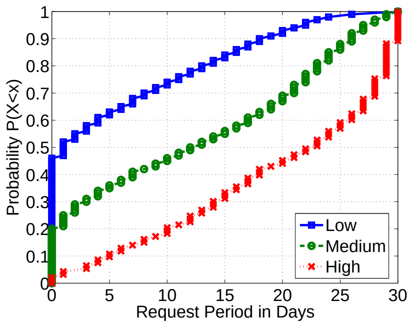


Fig. 7: Request period depending on the number of requests

videos with a smaller number of views tend to have a shorter request period. 44% of videos with less than 10 requests have a request period of less than a day. This share decreases for videos with medium and high popularity to 20% and 3%, respectively. More than 50% of the high popular videos have a request period of more than two weeks. Almost 12% of all videos were requested over the whole monitoring period which shows that a significant amount of videos are popular over a long time-period.

Cache sizes are very important for the estimation of caching benefits. Due to limitations in cache sizes, videos that are no longer watched need to be removed from a cache as soon as its disk is no longer able to store new videos. A video should not be removed if the probability for a subsequent request in the near future is still high. For this reason, we evaluated the probability that a video is requested at least one more time depending on the number of previous requests. Figure 8 shows the complementary probability of this event in order to provide a higher readability. The probability that a video is requested

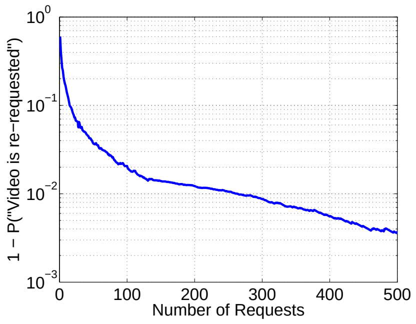
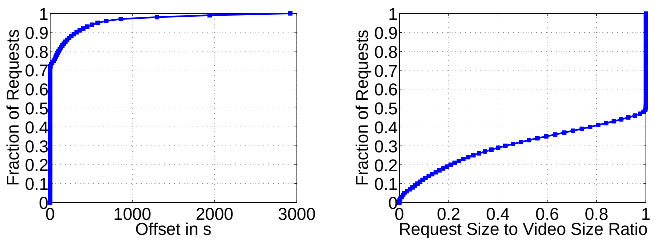


Fig. 8: Probability for a video not being requested one more time

at least one more time increases with the number of requests. The re-request probability of a video that was requested one time in the past is already 60%. This probability increases to 86% for videos that were requested 10 times and exceeds 98% for videos that were requested more than 100 times. The trend suggests that this probability converges against 99.9%. However, the number of videos with such a high number of requests was too low during our monitoring period to support such a statement with a sufficient level of significance.

YouTube users do not necessarily watch videos from the beginning since embedded YouTube videos can directly jump into a specific part of video by specifying a starting offset. Furthermore, if a user forwards a video to a not yet downloaded offset, a new TCP connection will be opened which starts a new download beginning from the chosen offset. Figure 9a shows the CDF of the offset of all requests. The figure reveals that 72% of all requests have an offset of zero This means the majority of the users request the beginning of a video, which is very beneficial for caches. Only 3.5% of all requests have an offset greater than 1000s which results from the fact that the average video duration is 331s.

In addition, users can abort a video download before the video has been downloaded completely. This can happen for several reasons, such as the user experiences bad download quality or is not interested in the video [14]. Therefore, we evaluate the behavior of the users by calculating the fraction of request size and video size in order to track how much



(a) CDF of request offset (b) Request size

Fig. 9: Request characteristic

of the video the user has watched. The results are plotted in Figure 9b. The figure shows that more than 50% of the requests request the whole video. Another 20% still request almost half of the video while only a very small fraction requests a small part of the video.

Videos that are not watched completely do not need to be fully cached. A cache has to decide whether the complete file is downloaded when a new video is requested, or if it only stores the requested bytes. We therefore examine whether certain parts of a video are watched with a higher probability, e.g. if the beginning of a video is more likely to be watched than the middle or the end of the video. Thus, we divided each video into chunks and calculated the probability for each chunk to be downloaded. As offsets are defined in milliseconds, we need to calculate byte offsets. For this calculation, we use meta information such as total video length (bytes and duration), which we could only obtain from the flash containers. As a result, we can only evaluate the chunk information for videos which where embedded in flash containers. Figure 10 shows the probability for different parts of the video to be watched in a given download. We observe

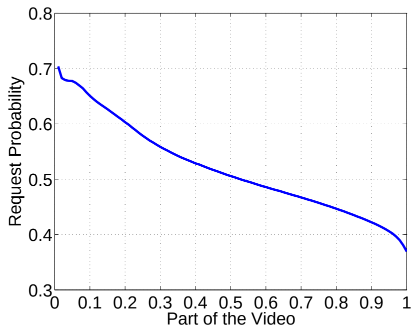
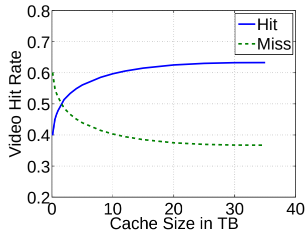
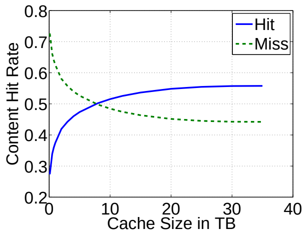


Fig. 10: Request probability of different video chunks

that not all chunks are viewed with the same probability. Video parts from the beginning of the video are more likely to be viewed than the latter parts of the video. The probability for a chunk to be viewed is decreasing with its distance from the start of the video, which is probably due to the fact that users abort a video before it is completely downloaded. We will study the effect of this finding in the following from a caches' point of view. If a cache loads and caches unnecessary chunks, it will on the one hand download too much content.



(a) Video hits



(b) Content hits

Fig. 11: Request characteristic

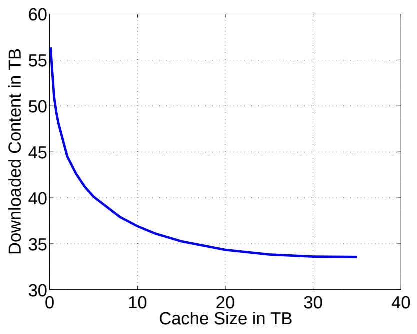


Fig. 12: Downloaded content from YouTube video servers

This content will on the other hand fill up disk space which is necessary for caching other relevant chunks.

B. Caching Benefits

For our evaluation of caching benefits, we use our monitoring data as input for a simulation. Our simulation aims at answering the question: "What if a YouTube video cache had been deployed in the network during our monitoring period?" We calculate benefits that could have been provided by different caches and caching strategies.

Caching strategies that define how videos are downloaded and replaced are very important. Another important factor is the disk size, which is a major limitation factor for cache performance. A caching strategy must decide for each user request, whether it will download the complete video or only those parts that have been requested by a user.

Zink et al. [11] propose to download complete videos upon user request and deliver subsequent requests from this video cache. They also propose a last recently used replacement scheme from the cache: If disk space is exhausted and a new video needs to be stored, the video that has not been requested for the longest time is removed from the cache. We implemented a simulation of this caching strategy and plotted the video and content hit rates for various disk sizes. A video hit is a user request for a video, which can be successfully answered from the cache. Video misses are user requests for videos that need to be fetched from the YouTube video servers. The same is applied to content hits and misses. Here we consider how many bytes of the request needed to be fetched from the YouTube servers and how many bytes could be delivered from the cache.

Figure 11a shows the cache hit and miss rates for all video requests during our monitoring interval depending on the cache size. We simulated caches with disks sizes between 100 GB and 35 TB, in order to determine hit an miss rates. Similar to Zink et al., we can see good hit rates. About 40% of all requests can be delivered from a cache with very small disk sizes (e.g. 100 GB). A cache with 2 TB disk space, could achieve a hit rate of more than 50%. Our maximum achievable video hit rate is more than 60% for a cache that is able to cache all requested content which corresponds to the video re-request probability for a video as shown in Figure 8.

However, a hit rate of more than 50% of the videos does not necessarily imply a high content hit rate. Figure 11b shows the content hit rate for caches of various sizes. We plot for each requested byte whether it could be delivered from the cache or whether it must be fetched from the YouTube infrastructure. Similar trends can be observed when looking at the hit and miss rates. However, 2 TB of disk space are not sufficient for a 50% hit rate in the cache. We need at least 8 TB in order to achieve a content hit rate of 50%. The maximum content hit rate is smaller than the video hit rate, but still exceeds 55%.

While these figures appear to be amazingly good, this caching strategy requires downloading the complete video. From our previous evaluation, we know that parts of the videos are not necessarily downloaded. Figure 12 shows the number of bytes that have been fetched from the YouTube video servers depending on the cache size. It can be seen, that this number is very high for small cache sizes and reduces to 33.6 TB with higher cache sizes. The reason for this is that all unique video content, if fully downloaded, results in 33.6 TB of traffic. However, users did not download this unique content completely, but only parts of it. This unnecessarily fetched data must be stored on disk and occupies disk space which is needed for videos that are requested completely or requested multiple times. For small cache sizes, many important videos are removed from the cache, and need therefore to be downloaded from YouTube several times for subsequent user requests. It is therefore important not only to look at cache hit rates, but also on the number of bytes which have to be fetched from the YouTube video infrastructure. One more important conclusion, according to our monitored data, is that a caching strategy which fetches the complete content instead of the requested content, is not an efficient strategy.

Thus, we evaluated a cache which only stores content chunk-wise (chunk strategy): Videos are separated into 100 chunks, and chunks are only cached on user request. For the reasons outlined before, we can only consider flash content for this evaluation.

Therefore, the numbers for video data and requested content change: The complete size of the flash videos is 29.5 TB (compared to 33.6 TB for all videos). 9.7 TB of this video sizes where not viewed at all, e.g. due to premature download aborts. Storing these parts of the videos in the cache would unnecessarily occupy valuable disk space. User requests to YouTube for flash content sum up to 34.4 TB of video

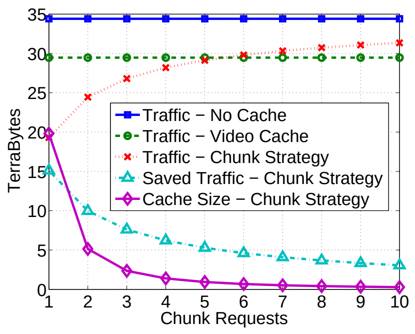


Fig. 13: Chunked caching strategies

downloads, if no cache is used. A cache which downloads the complete video content if a video is requested (as simulated before), will download 29.5 TB of flash content from the YouTube provided that it is able to cache all requested videos. These two numbers are therefore the base-line for our chunked caching strategy.

A caching strategy has to provide mechanisms that decide when to store a chunk. Each chunk can be stored when it is requested for the first, the second, or more times. This decision has large impact on storage requirements and download traffic reduction of a cache. Popular chunks need to be downloaded twice, three times or more before any cache hit can appear, thus reducing the benefits in download traffic. On the other hand, waiting for a chunk to be requested several times before caching reduces the required cache size.

We evaluated the effects and benefits of a chunked caching strategy and plotted the results in Figure 13. The figure shows the cache sizes that are required and the traffic to the YouTube infrastructure, depending on the number of requests of a chunk before this chunk is stored. If we store chunks at their first request, a cache needs disk space of 19.5 TB for storing all chunks, and generates the same amount of traffic to the YouTube servers. Hence, when deploying such a cache, the amount of downloads from YouTube can be reduced by 15 TB. If we cache chunks on the second occurrence of a chunk, the required cache size drops to 5 TB (diamond markers), and the amount of traffic to the YouTube servers increases to about 25 TB (cross markers). The amount of reduced download traffic drops by this 5 TB (triangle markers), since popular chunks need to be fetched twice. By comparing the results of the chunked caching strategy with the complete download strategy (triangle markers vs. dashed line), we can see that a properly configured chunked caching strategy performs much better than a properly configured strategy that downloads complete videos. Furthermore, the chunked strategy allows to deploy smaller caches to achieve this high caching benefits.

VI. Conclusion

In our study, we monitored and analyzed traffic between a large operator network and the YouTube video distribution site for over a month. We were able to find good local popularity values for YouTube videos, which result in high caching potential. User behavior, such as users not fully watching a video, can have significant negative impact on caching performance and effectiveness. Caches that do not take this behavior into account, experience caching performances penalties compared to properly configured caches. Our analysis revealed, that a chunked caching strategy provides very good performance for YouTube video content. In future work, we plan to extend our study to other video platforms than YouTube, and to propose concrete cache implementations that can be used with various video platforms.

References

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, “Internet Inter-Domain Traffic,” *Proceedings of the ACM SIGCOMM 2010 Conference on SIGCOMM*, 2010.

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman, “On Dominant Characteristics of Residential Broadband Internet Traffic,” pp. 1–13, Sep. 2009.

[3] G. Münz, S. Heckmüller, L. Braun, and G. Carle, “Improving Markov-based TCP Traffic Classification,” in *Proceedings of Communication in Distributed Systems (KiVS) 2011*, Kiel, Germany, Mar. 2011.

[4] Cisco Corporation, “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 20102015,” http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf, Feb. 2011.

[5] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, “Analyzing the Video Popularity Characteristics of large-scale User Generated Content Systems,” *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 5, pp. 1357–1370, Oct. 2009.

[6] R. Zhou, S. Khemmarat, and L. Gao, “The Impact of YouTube Recommendation System on Video Views,” *IMC ’10: Proceedings of the 10th annual conference on Internet measurement*, 2010.

[7] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, “Youtube traffic characterization: a view from the edge,” in *IMC ’07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement.* ACM Request Permissions, Oct. 2007.

[8] G. Chatzopoulou, C. Sheng, and M. Faloutsos, “A First Step Towards Understanding Popularity in YouTube,” in *INFOCOM IEEE Conference on Computer Communications Workshops , 2010*, 2010, pp. 1–6.

[9] F. Figueiredo, F. Benevenuto, and J. M. Almeida, “The Tube over Time: Characterizing Popularity Growth of Youtube Videos,” in *In Proceedings of the 4th ACM Conference on Web Search and Data Mining*, 2011.

[10] 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 Conference on Internet Measurements (IMC) 2007*, 2007.

[11] M. Zink, K. Suh, Y. Gu, and J. Kurose, “Watch Global, Cache Local: YouTube Network Traffic at a Campus Network-Measurements and Implications,” *Proceedings of the 15th SPIE/ACM Annual Multimedia Computing and Networking Conference (MMCN)*, 2008.

[12] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. Munafo, and S. Rao, “Dissecting Video Server Selection Strategies in the YouTube CDN,” in *Distributed Computing Systems (ICDCS), 2011 31st International Conference on.* IEEE Computer Society, 2011, pp. 248–257.

[13] V. K. Adhikari, S. Jain, and Z.-L. Zhang, “YouTube Traffic Dynamics and Its Interplay with a Tier-1 ISP: An ISP Perspective,” *IMC ’10: Proceedings of the 10th annual conference on Internet measurement*, 2010.

[14] A. Finamore, M. Mellia, M. Munafo, R. Torres, and S. Rao, “YouTube Everywhere: Impact of Device and Infrastructure Synergies on User Experience,” *Technical Report*, 2011.

[15] B. Ager, F. Schneider, J. Kim, and A. Feldmann, “Revisiting Cacheability in Times of User Generated Content,” in *INFOCOM IEEE Conference on Computer Communications Workshops , 2010*, 2010, pp. 1–6.

[16] L. Braun, A. Didebulidze, N. Kammenhuber, and G. Carle, “Comparing and improving current packet capturing solutions based on commodity hardware,” *IMC ’10: Proceedings of the 10th annual conference on Internet measurement*, Nov. 2010.

[17] L. Deri, “High Speed Network Traffic Analysis with Commodity Multi-Core Systems,” *IMC ’10: Proceedings of the 10th annual conference on Internet measurement*, Nov. 2010.

[18] Tstat Homepage, http://tstat.polito.it.