An analysis of the YouTube CDN

Max Crone

max.crone@aalto.fi

Jack Henschel

jack.henschel@aalto.fi

***Abstract*—This report is a research into the video distribution and caching strategies employed by YouTube. We collect data from multiple continents for a multitude of videos. Our analyses find geolocations of cache servers used by YouTube, measure their performance and interpret the cache host names. Our data did not support any clear strategy based on time of the week or region, with the exception of the case for uploading a video, in which the region of upload had a clear advantage for the first hour after publishing compared to other regions. We conclude by formulating our expectation that the distribution and caching in YouTube's infrastructure is currently largely governed by machine learning models and therefore our research could not find a clearly discernible strategy.**

***Index Terms*—video, distribution, cdn, youtube**

I. Introduction

In this report we formulate hypotheses on the strategies employed by YouTube in their distribution and caching of videos around the world, based on the data we will collect from different geographical vantage points. The intention of this research is to gain an understanding of the infrastructure of YouTube and its workings, which are arguably a blackbox in the current public knowledge.

First, we need to know how the basic setup of YouTube's video and load distribution works. When a client requests a video from YouTube's website youtube.com, the web frontend returns a rendered HTML webpage which contains a link pointing to a specific cache server (e.g. r4---sn-4g5e6nsd.googlevideo.com). YouTube does not disclose how and based on which parameters this cache server URL is generated, but by querying many different videos from different client locations we can infer a part of their strategy. Second, the client has to resolve the given hostname to an IP address. Depending on the particular DNS configuration, the DNS server will also refer to different IP addresses for different clients for the same hostname. After connecting to this IP address, the client can stream the video from YouTube's CDN infrastructure (backend servers). This entire process is illustrated in Figure 1.

II. Part 1

In the first part of this report, we are investigating how YouTube dynamically returns different video cache servers for different users all over the world. We also run additional measurements to see how well this infrastructure works to improve the quality of experience (QoE) for the users. We will formulate conclusions about the video caching strategies of YouTube based on our findings.

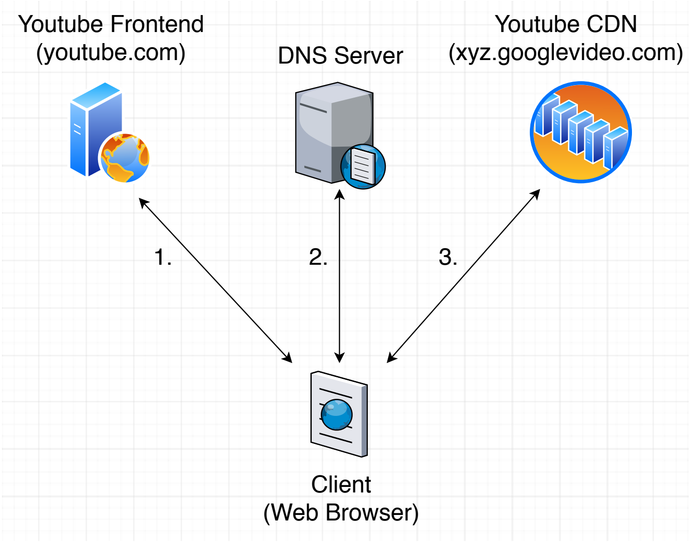

Fig. 1. High-level overview of YouTube video streaming

A. Experimentation setup

Firstly, we composed a diverse list of videos which we will use in our experiment. We decided to use two videos for each of the major continents: one video that is very popular on the continent and another one that is rather obscure. Thus, we have a total of ten videos for our measurements (Table I).

TABLE I
Measurement Videos

Video ID	Continent	Type	Views
_WuWQn0SZ54	Asia	Popular	500k
ty19DTniYR0	Asia	Obscure	9
8mMAHPEkmuM	Africa	Popular	750k
6eGaPzFAUdU	Africa	Obscure	400
Jzl_nrTkfIM	South America	Popular	500m
gnBQ_kc6Zxg	South America	Obscure	66
OEsfbBHdqAc	North America	Popular	2.5m
9hj9eNvjLtE	North America	Obscure	160
8ma7afWF7r0	Europe	Popular	3m
m4QacHOQHwI	Europe	Obscure	100

We use virtual machines deployed across four major continents to conduct our measurements. The locations are Europe **EU** (Frankfurt), North America **NA** (Iowa), South America **SA** (Sao Paolo) and Asia **AS** (Taiwan). Ideally, we would have also liked to deploy a VM in Africa, but our cloud provider of choice is not offering any resources there.

Our measurements are automatically collected every four hours: at 0:00, 4:00, 8:00, 12:00, 16:00 and 20:00 o'clock (six measurements per day). The time between these runs allows the videos to be evicted from the network caches again, thus our measurements should not be of any influence to the observed performance in our experiments. We run these measurements continuously for one week, so we can also capture differences in the video cache distribution strategies employed by YouTube at different days in the week.

Thus, in total we collected measurements on 10 videos from 4 geographically different vantage points with 6 runs per day on 7 consecutive days, resulting in 1,680 unique measurements.

To collect the various important variables for video streaming (such as server hostnames, IP addresses, latency etc.), we use the Pytomo tool. It was initially developed in 2011 to analyze YouTube's infrastructure by setting up a lot of monitoring agents on home internet connections [1]. Since it is a relatively old program, we had to apply a few modifications in order to have it function correctly in the current landscape of YouTube's infrastructure (YouTube refuses any kind of unencrypted HTTP connection, but Pytomo has these hardcoded in a lot of places).

Furthermore, Pytomo's way of saving the data is inadequate for data analysis. For each run, Pytomo creates a new database named according to the current time. In this database file there is a table, that is also named according to the time of the run, but in a slightly different format. For example, during the week of our data collection Pytomo created 42 individual database files containing one unique table each, instead of reusing the same database, possibly with different table names.

To analyze all our measurements (which were conducted at different times and on different machines), we wrote a custom script to download and merge all the databases into one. Equipped with this database, we can analyze our data effectively.

B. Analysis

1) IP Geolocation: We started by mapping the IP addresses of the cache servers that returned the video data to our clients to geolocations. This would help us gain insight into the locations from where YouTube serves the videos, which will prove useful in determining their distribution strategy. Precisely determining the geographical location of a particular IP address is hard. Even though there are many online services providing IP to geolocation mappings, in our experience they locate many of the IP addresses in North America, even though we had good reason to assume that these servers actually reside in other continents (proximity measurements with ping). Thus the geolocation services we had access to are not precise enough to support our analysis. Despite these issues, we present our process either way, to show how to conduct this analysis properly in case the geolocations are accurate.

Our approach was as follows:

1) Count and group IP addresses of cache servers for every one of the four continents.
2) Map every IP to a geolocation.
3) Count occurence of IP addresses in a certain region.

This gives us the graph presented in Figure 2. This graph clearly supports our earlier observation: the inaccuracy of IP geolocation databases is a well known problem [3] that we also encounter with the database we used. [2] When performing manual RTT measurements, many of the IPs assigned to the US by the geolocation database have a ping too low to resemble an intercontinental connection. Our hypothesis is that due to the volatility of "cloud" systems (like Google's video CDN), IP address assignments can change quickly and are no longer geographically bound. This is especially true for IPv4 addresses which are a scarce resource these days.

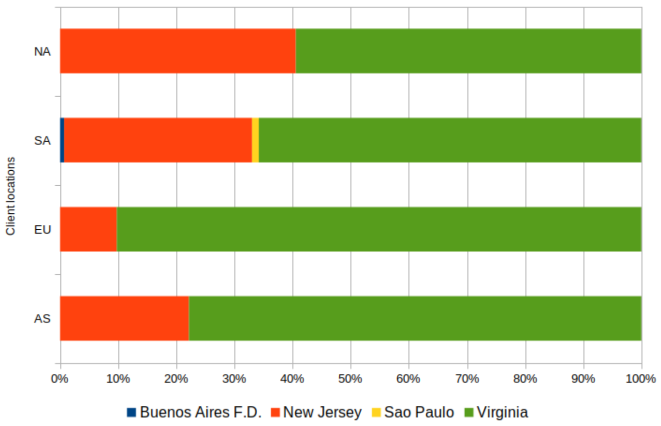

Fig. 2. Origin of cache servers used to serve requests from our four client location

We came up with a second method of approximating the geolocation of the cache servers. The *traceroute* command provides the route taken by packets across the network. By studying the number of hops it takes to reach any of the IP addresses of the cache servers we found, we are able to estimate how far away these servers are. All our traceroute operations were conducted from the Aalto University network. Thus from this perspective we would expect the number of hops to be the lowest for European servers, while they ought to be the highest for Asian or South American servers. We performed traceroutes on every one of the 104 unique IP addresses we found during our data gathering phase, grouped by origin continent. The results are plotted in a boxplot in Figure 3. This shows us that there is a very clear separation in the average number of hops it takes to get to a certain continent. The results are also summed up in Table II.

From these results we conclude that there can be two reasons for the occurence of servers with a higher hops distance in Asia and North America. On the one hand the larger distance could mean that the route goes to a server still in the same continent, but just a less easily reachable one. On the other hand the larger distance in hops could mean that the cache server is not even located in the continent that it served,

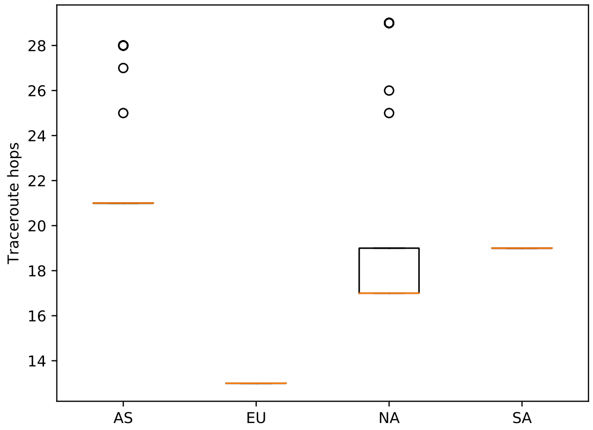

Fig. 3. Boxplot of the number of hops in the network from Aalto University campus to cache servers in the designated regions.

TABLE II
AVERAGE NUMBER OF NETWORK HOPS FROM AALTO UNIVERSITY CAMPUS TO THE CACHE SERVERS USED IN THE GIVEN REGIONS

Region	Average number of hops
Asia	21.2
Europe	13
North America	18.0
South America	19

but instead resides on another continent. In this case, Asia is the furthest away in terms of network hops. So this could mean that some of the cache servers that served requests made from North America which were found to be of distance 29, 26 or 25 hops away could reasonably be cache servers located in Asia. We found that the cache server of distance 29 hops from Aalto University campus has served twelve requests in total to the client from North America only. The only video it served, was the popular Asian video. Thus it seems plausible that this server is a cache in Asia that occasionally serves a video to a North American client. However, if this would truly be a cache server located in Asia, then we would expect it to also serve Asian clients. Our data show no record of any of the requests originating from Asia to be served by this particular cache server. So instead, this particular server could also be a somewhat more isolated datacenter still residing in North America. The same analogy can be applied to all of these outlier cases, causing uncertainty for the geolocation of these particular cache servers.

As a conclusion to our traceroute analysis we state that the large majority if not all requests made by a client on a certain continent will be served by cache servers on that same continent. The clear distinction in number of network hops for all continents supports this conclusion, with the rare outlier not being straightforward to interpret.

2) Proximity to YouTube: We then looked at the proximity between our VMs and YouTube's infrastructure. For this purpose we used the average measured ping to the server. From Figure 4 we can see that our system in North America had by far the "longest" connection to YouTube, though compared to most residential connections an average ping of 11 ms is still very quick. The other clients had a very good connection to YouTube's servers (within a few milliseconds). We suspect this is due to the fact that we rented elastic cloud VMs which are run on servers in datacenters that are extremely well connected with all major and important networks worldwide.

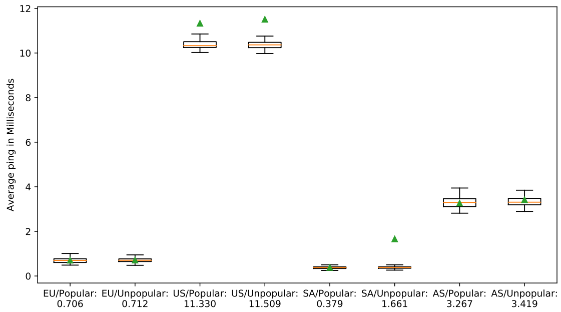

Fig. 4. Boxplots of average ping between VM and YouTube infrastructure for each location

Additionally, Figure 4 also shows that the metric does not differ significantly when querying either a popular video or an obscure video. A few outliers are not shown in the boxplots to make the rest of the graph more readable. These outliers most likely correspond to sporadic network failures or congestion issues and are not statistically relevant.

3) Streaming Startup Delay: Next we analyzed the "TimeToGetFirstByte" metric which refers to the time it took the cache server to respond to the clients streaming request with a first chuck of video. We expected there to be a significant difference between popular and obscure videos, based on our assumption that YouTube caches popular videos more aggressively and at the same time evicts obscure videos relatively quickly from its caches. Our findings in Figure 5 do not show behavior in line with our hypotheses, however. One plausible explanation is that YouTube has tweaked and optimized its infrastructure a lot to minimize the startup delay for its clients.

Figure 4 and 5 consistently show that the client location in North America (Iowa) has the "longest" connection to YouTube's infrastructure compared to the other vantage points.

Another comparison we deemed interesting was the correlation between the geographical proximity to YouTube's servers and the video streaming startup delay. We can use our ping measurements as an estimate for the proximity and the `TimeTogetFirstByte` for the startup delay.

With these measurement we can calculate the correlation value according to the following formula (with X as the proximity and Y as the delay):

$$\rho_{X,Y} = \mathrm{corr}(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$$

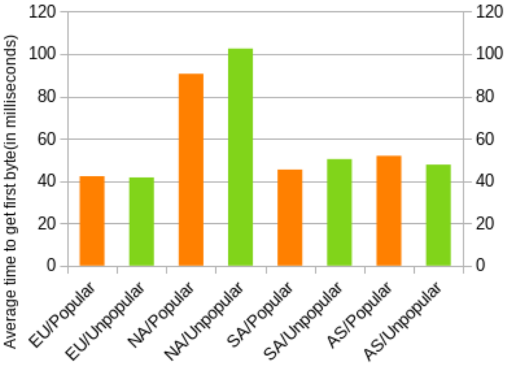

Fig. 5. Average streaming startup delay, for each video type and client location

Having done this for each client location and video type individually, we obtain Table III. In the case of the vantage points in Asia (AS) and South America (SA) there is a strong correlation between the proximity and the startup delay for unpopular videos. The EU and NA cases however are more inconsistent. The EU client location shows very little correlation between these two values, while the US location shows only a strong correlation for popular videos. Thus we cannot draw any clear conclusion from this metric.

TABLE III
CORRELATION BETWEEN PROXIMITY AND DELAY

	EU	NA	SA	AS
Popular	0.1292	0.6398	0.0181	-0.0334
Obscure	-0.0881	0.3226	0.8598	0.7406

4) Timeseries: We also looked into how the response times from the cache servers behave over the course of a week. Figure 6 and 7 show the “TimeTogetFirstByte” metric for two of our videos (Y axis) from all our vantage points (Legend) for each of our measurements (X axis). These two diagrams confirm again that the compute instance in North America had the slowest connection to Youtube’s video servers. The high spikes in both diagrams are likely just outliers (probably influenced by other network traffic) and can thus safely be ignored.

The measurements of the popular video (Figure 6) are a lot more volatile than the obscure video (Figure 7) which is what we would expect from a video with very high demand. Unfortunately, in both figures we can not really identify a clear pattern describing when the TimeToGetFirstByte will be low or high.

Figure 8 plots the delay of the answer from the cache server for the popular Asian video. Most notably, the client in South America has to wait the longest for an answer from the cache server, whereas the client in Asia gets the quickest response. Again, there are no time-based patterns in the plot which is what we initially expected to find (e.g. high load in the evening influences delay from cache server).

5) Hostname analysis: After analysis of the gathered data, we propose an interpretation of the structure of cache URLs. Consider the following URL:
`https://r5---sn-4g5ednle.googlevideo.com`

The last part of the subdomain, `4g5ednle` in this URL, is thought to identify a cluster of cache servers in geographic vicinity of each other. Let us call this a *service node ID*. The first part of the subdomain, `r5` in this example, refers to a specific ingress point to that cluster. There are `r1` through to `r6` ingress points in each cluster. `sn` most likely stands for *service node*, based on the terminology used in Google infrastructure. [4]

While we believe the service node IDs to be of random structure, we have found similarities in their prefixes that coincide with the region served by the service nodes. These are summarized in Table IV.

TABLE IV
CACHE URL PREFIXES PER REGION

Region	Prefixes
AS	`un57`
EU	`4g5e`
NA	`vgq, qxo`
SA	`bg0`

This indicates that YouTube has a structure for generating cache URLs for regions, despite the lack of an encoded name of a specific region. Based on our data we can draw no conclusions about the selection of the remaining characters in the service node ID. Thus as far as we can say, they are randomly generated to satisify the requirement for unique IDs.

These prefixes will not be the only ones in use for the YouTube infrastructure. This experiment could after all only capture a small part of the total infrastructure. Our data also show a few instances of other service node ID prefixes, for example when an occasional redirection to a different cache URL occurred.

6) Redirects: Of the 1680 measurements we collected, only 40 got a redirect from the cache server. This means the cache server redirects the client to a different cache server with HTTP status code 302 instead of serving the video file directly. Of these 40 redirects, 20 were for popular videos and again 20 for obscure videos. There are two possible scenarios in which a cache server would redirect the client: a) the cache server is overloaded with requests, thus directs the client to a different server or b) the cache server does not have the requested video file available and also can not fetch it in the background. Looking at our data, we could not find a clear indicator for when these redirects occur.

There emerged two different redirect URL patterns from our data. Either the client first queries https://redirector.googlevideo.com to be redirected to an actual video cache server (such as https://r3---sn-vgqsrnek.googlevideo.com) or the client queries an actual cache server but gets redirected to a different instance (e.g. from https://r3---sn-vgqs7nlk.googlevideo.com to https://r6---sn-vgqs7nlk.googlevideo.com). The latter type of redirect

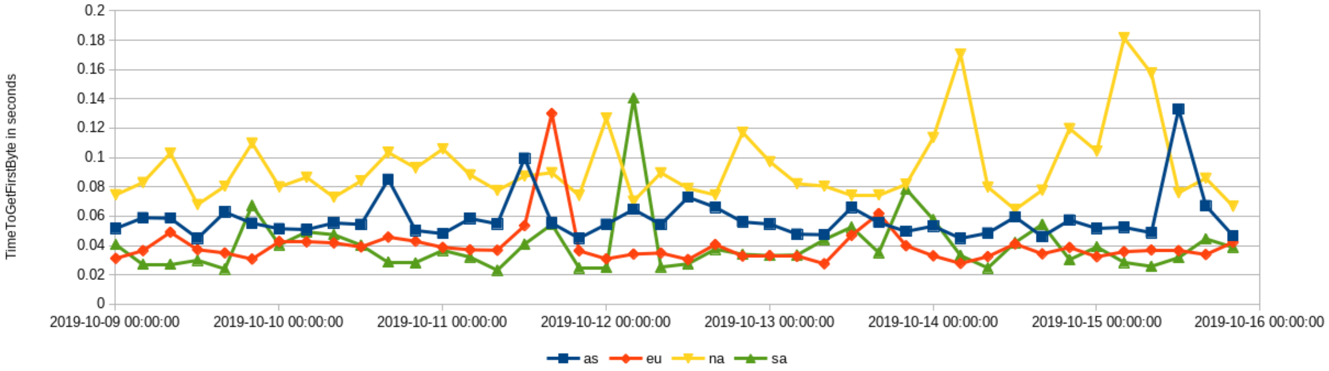

Fig. 6. TimeToGetFirstByte of popular European video

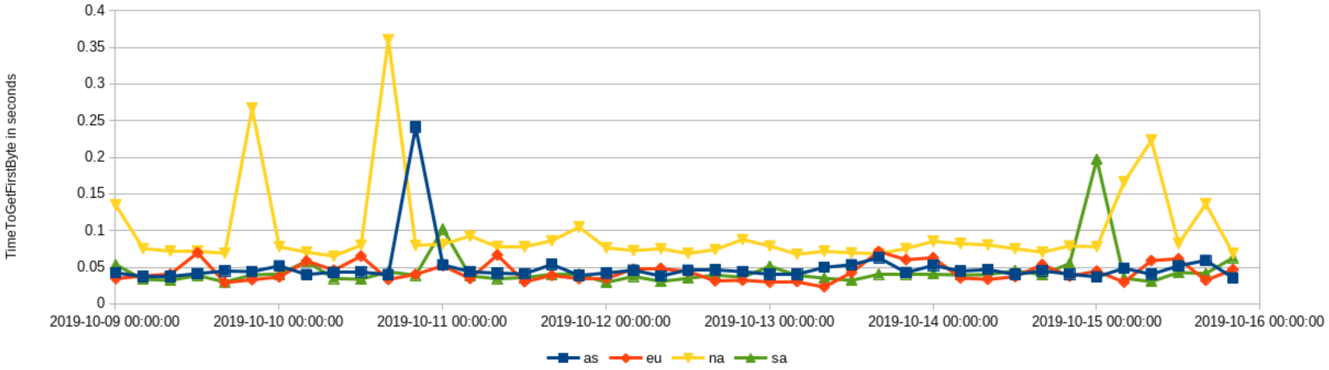

Fig. 7. TimeToGetFirstByte of obscure South American video

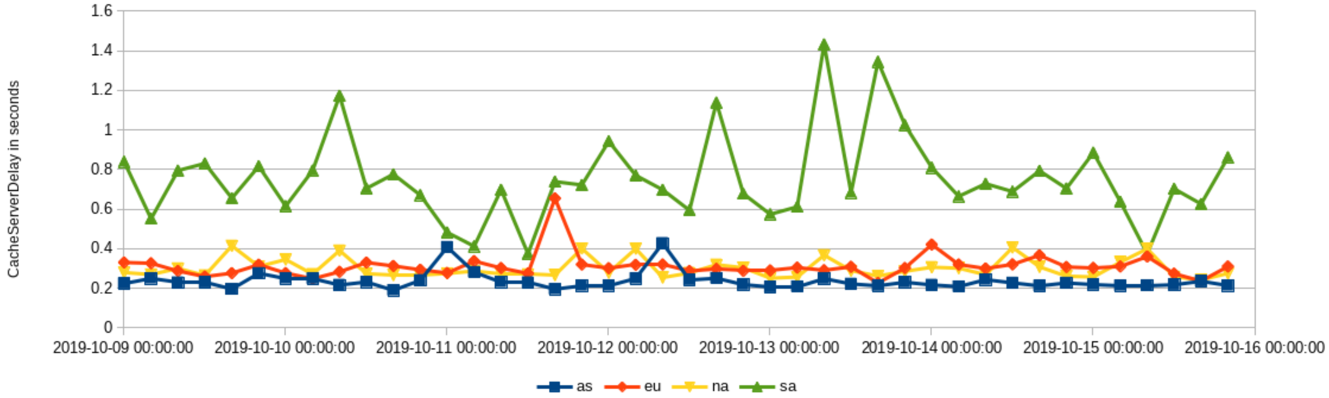

Fig. 8. CacheServerDelay of popular Asian video

only occurs within the same cluster of cache servers (e.g. from `r3` to `r6`). Our hypothesis is that all cache servers within one cluster (same service node ID, e.g. `vgqs7nlk`) work together as a unit (i.e. serving different videos to different users).

III. Part 2

This part explores the distribution and caching strategies YouTube employs on their platform for newly uploaded videos.

A. Experimentation setup

First we created a unique, high definition video and uploaded it via the YouTube web interface from the Aalto University campus in Finland. We again employ four virtual machines at the different physical locations around the world as expanded upon earlier. On all four machines, we run the Pytomo tool for our newly uploaded video at different intervals after the original time of upload. These are as follows: 1 minute, 5 minutes, 15 minutes, 30 minutes, 60 minutes, 2 hours, 6 hours, 12 hours. Because these are being conducted on four continents simultaneously, it will result in 32 unique measurements. Our intention by selecting these delays is to gain an understanding of the completeness of YouTube's video caching and distribution strategy over time.

Additionally, we look at the Developer Tools integrated into modern web browsers. The "Networking" tab shows the requests a website makes in the background and thus also reveals information about the CDN infrastructure.

B. Analysis

Figure 9 shows how the TimeToGetFirstByte metric behaves in the different locations over the course of 12 hours after the video has been uploaded. Figure 10 illustrates the download time of our uploaded video, again for the different locations over the course of 12 hours. In both figures the EU client location consistently has the best performance. This matches our expectation since the source video was uploaded from the EU region (Helsinki). Thus, logically YouTube will first distribute the video there before also replicating it to other continents. After as little as half an hour this process is already completed. We can infer this from the figures since all client locations exhibit the same performance after half an hour.

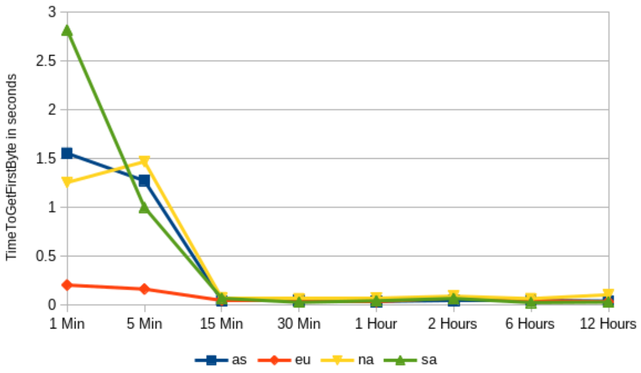

Fig. 9. Time to get first byte for custom uploaded video

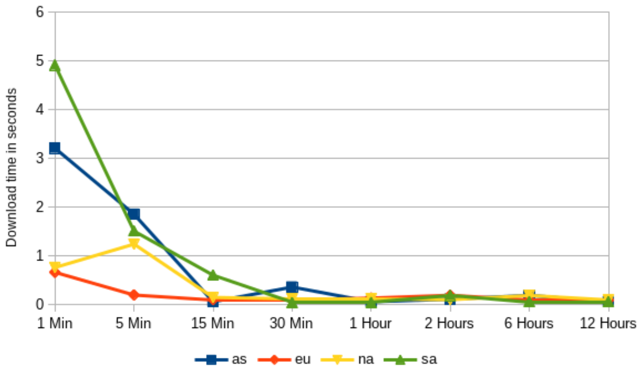

Fig. 10. Download time for custom uploaded video

To make these results more sound, we ran a second experiment: this time uploading a video from a client location in South America (Sao Paolo). Again, we used all our client locations to generate measurements for this particular video. The results in are the same as before: Both Figure 11 and 12 clearly show that the region where the video was initially uploaded to consistently has the best performance (in this case: SA). The access times from the other locations (AS, EU, NA) are a lot higher at first (multiple seconds), but after roughly half an hour the performance of all client locations converges to the same level (few hundred milliseconds).

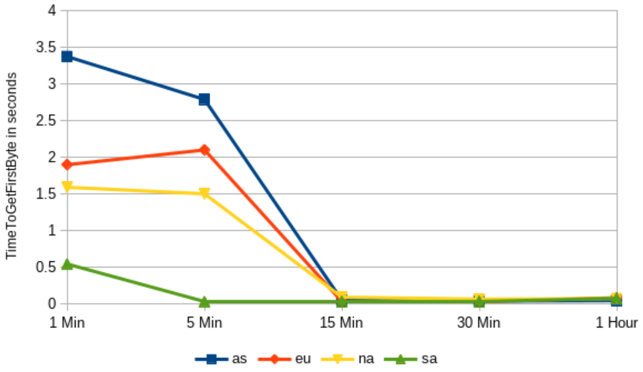

Fig. 11. Time to get first byte for second uploaded video

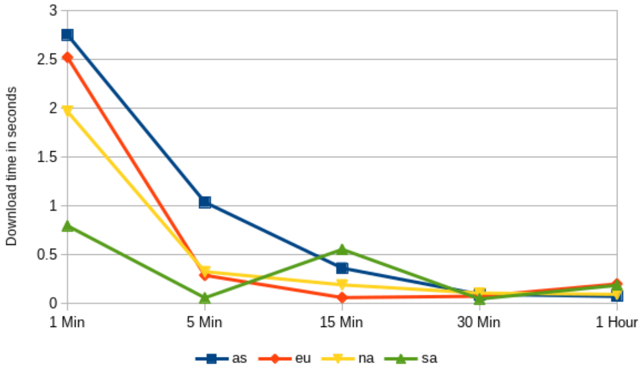

Fig. 12. Download time for second uploaded video

Another interesting observation we made while uploading videos through YouTube's web interface was that the video server cache URLs differ between the video uploader and other users or viewers. The uploader gets a URL of the form https://r1---sn-bg07dnsr.c.youtube.com, whereas others get URL likes https://r1---sn-5hne6nsd.googlevideo.com. There seem to be two different caching services at work here. We assume the first one (*.c.youtube.com) is more direct, meaning it always returns the most current version of the video to the content creator (e.g. when YouTube is processing the video). This cache can likely only handle less load compared to *.googlevideo.com which is used to serve all other users.

IV. Limitations

The biggest challenge for this report was trying to look inside Google's Autonomous System (AS) which is where YouTube's services run. An autonomous system (AS) is a network or a collection of networks that are all managed and supervised by a single entity or organization. As soon as our traffic to YouTube entered Google's AS it became extremely hard to estimate server location, due to the fact that both the hostnames of the systems are cryptic as well as the IP addresses could not be pin pointed.

Inaccurate geolocation IP databases were the second big impediment in conducting this research. Most of the freely available databases seem to consist of old, incomplete or inaccurate data. We verified this by doing manual proximity measurements with ping. Most of the IP addresses we collected were assumed to be from the US (according to the IP databases), but our measurements quickly confirmed that the servers where either much closer (e.g. Europe) or much further away (e.g. South America and Asia). We assume this is due to the fact that in the cloud landscape IP address assignments can quickly and unexpectedly change location.

The fact that we had to use cloud-based virtual machines may be another limitation of our experimental setup. We had however no other option if we wanted to conduct our experiments world wide. But since these VMs are located in extremely well interconnected datacenters, their proximity to YouTube's server infrastructure was extremely close. Much closer than what an average home internet connection would have. Therefore any observed differences would be much smaller and thus easier to overlook.

V. Conclusion

In this report we conducted active measurements of the distribution of videos by YouTube to its end users by simulating users viewing a multitude of different videos from four different continents. The intention was to gain an understanding of the strategies employed by YouTube for its video distribution and caching.

Our analysis established that videos are most likely always served from cache servers residing on the same continent or in the same region as the end user, even for content that is mostly popular in a completely different part of the world, or when it is not popular at all. We conclude from this that YouTube's global infrastructure is so powerful that it no longer has to focus on savings on a continental scale. For intracontinental connections, the delay between an end user and YouTube's infrastructure is also low, which contributes to the overall fast availability of the video services. Even though we found that popular videos were started faster on average than unpopular ones, the practical difference for an end user is almost negligible. We also established some structure and found some patterns in the format YouTube uses for their cache server host names. When uploading our own videos, we noticed a very clear advantage in availability for the region from where we uploaded the video. Thus a video spreads from its region of upload across the rest of the global infrastructure of YouTube.

Our overall observation is that YouTube offers a nearly uniform quality of service for viewing their videos globally. Our research shows no distinct differences over time of the week nor for any specific regions in any of our test cases. From our data only arises a general tendency of popular videos being served slightly faster than more obscure content. This uniform experience and lack of clearly discernable strategies leads us to believe that much of the video distribution and caching are currently being managed by machine learning and game theoretic models that are highly optimized for general quality of service and quality of experience parameters. [5]

References

[1] P. Juluri, L. Plissonneau, D. Medhi, "Pytomo: A Tool for Analyzing Playback Quality of YouTube Videos", Proceedings of the 23rd International Teletraffic Congress, 2011.

[2] ip-api.com, https://ip-api.com, 2019

[3] Poese, I., Uhlig, S., Kaafar, M. A., Donnet, B., & Gueye, B. (2011). IP geolocation databases: Unreliable?. ACM SIGCOMM Computer Communication Review, 41(2), 53-56.

[4] Google, "Google Edge Network Infrastructure", retrieved from https://peering.google.com/#/infrastructure on 2019-11-02.

[5] W. Hoiles, O. N. Gharehshiran, V. Krishnamurthy, N.-D. Dào, H. Zhang, "Adaptive Caching in the YouTube Content Distribution Network: A Revealed Preference Game-Theoretic Learning Approach", IEEE Transactions on Cognitive Communications and Networking, Vol. 1, No. 1, March 2015, 71-85.