Internet Scale User-Generated Live Video Streaming: The Twitch Case

Conference Paper *in* Lecture Notes in Computer Science · February 2017

DOI: 10.1007/978-3-319-54328-4_5

CITATIONS
23

READS
1,893

4 authors, including:

Gareth Tyson
Queen Mary, University of London
146 PUBLICATIONS **1,769** CITATIONS
SEE PROFILE

Félix Cuadrado
Queen Mary, University of London
62 PUBLICATIONS **611** CITATIONS
SEE PROFILE

Steve Uhlig
Queen Mary, University of London
208 PUBLICATIONS **7,792** CITATIONS
SEE PROFILE

Internet Scale User-Generated Live Video Streaming: The Twitch Case

Jie Deng, Gareth Tyson, Felix Cuadrado, and Steve Uhlig

Queen Mary University of London, UK
{j.deng,gareth.tyson,felix.cuadrado,steve.uhlig}@qmul.ac.uk

Abstract. Twitch is a live video streaming platform used for broadcasting video gameplay, ranging from amateur players to eSports tournaments. This platform has gathered a substantial world wide community, reaching more than 1.7 million broadcasters and 100 million visitors every month. Twitch is fundamentally different from "static" content distribution platforms such as YouTube and Netflix, as streams are generated and consumed in real time. In this paper, we explore the Twitch infrastructure to understand how it manages live streaming delivery to an Internet-wide audience. We found Twitch manages a geo-distributed infrastructure, with presence in four continents. Our findings show that Twitch dynamically allocates servers to channels depending on their popularity. Additionally, we explore the redirection strategy of clients to servers depending on their region and the specific channel.

Keywords: Twitch.tv; live video streaming; video streaming infrastructure

1 Introduction

Online live streaming has long been a popular application. However, recently, there has been an interesting evolution, whereby everyday users provide streams of their own activities, *e.g.,* Facebook Live, Periscope, Meerkat. This is termed *user-generated live streaming*, and unlike other platforms (*e.g.,* YouTube [16,15] and Netflix [10,7]), often involves things like live social interaction. Thus, these platforms introduce two core innovations: (*i*) Any user can provide a personal live stream (potentially to millions of viewers); and (*ii*) This upload must occur in realtime due to live social interaction between consumers and producers. One of the most popular examples of this is *Twitch* [3,22]. This live broadcast platform is oriented towards video games, allowing users to broadcast their gameplay, as well as to watch large eSports tournaments with professional players. Though others have started similar services (*e.g.,* YouTube Gaming), they are yet to experience the demand of Twitch [18,19], which delivered 35k streams to over 2 million concurrent users in real time during its peak [5].

The rapid expansion of user-generated live streaming platforms, like Twitch, comes with fundamental challenges for the management of infrastructure and traffic delivery.[1] For example, in Twitch it is impossible to time-shift (cache)

[1] Note that Twitch is the fourth largest source of peak traffic in the US [4].

video content, and often uploaders are not geographically near or well connected to their subscribers. Further, live social interaction (*e.g.,* via web cams and chat feeds [17]) means that the *real-time* constraints are very strict. Thus, we argue that Twitch might offer some important insights into how such challenges can be overcome.

In this paper, we perform a large-scale measurement study of Twitch. Taking advantage of a global network of proxy servers, we map the infrastructure used by Twitch. We explore its content replication and server selection strategies, correlating them with both viewer and broadcaster location. Note that broadcaster selection is a unique aspect of personalised video streaming, as prior systems lack the concept of user-generated live broadcasters. In this paper, we analyse how Twitch has managed to scale-up to deal with its huge demand. In summary, we make the following contributions:

- We map the infrastructure and internetworking of Twitch. Unlike YouTube or Netflix which deploy thousands of caches in edge networks, Twitch serves millions of users directly from relatively few server locations in North America (NA), Europe (EU) and Asia (AS) (§3).
- Based on this, we expose how streams are hosted by Twitch at different locations (§4); we explore how Twitch scales-up depending on channel popularity, and how clients are redirected to Twitch servers.
- We evaluate the client redirection strategy (§5) on a global scale. We find multiple factors affecting the redirection policy, including channel popularity and the client network configuration (peering). Due to the lack of peering in Asia, 50% of the clients are exclusively served by NA servers.

2 Measurement Methodology

We begin by presenting our measurement methodology, which is driven by three goals. First, we wish to discover the location and number of servers in Twitch's infrastructure. Second, we want to know how Twitch allocates individual live streams onto these severs (note that this is a very different model to static video content, which is usually reactively cached wherever it is requested). Third, we want to understand how users are mapped to servers so that they can watch the stream they are interested in.

We built a Python crawler that allows us to automatically request video streams from Twitch channels. The responses to these requests allow us to inspect which server the client has been redirected to.[2] In order to comprehensively sample the infrastructure, and explore how different clients are redirected to Twitch servers, we ran this crawler in many geographic locations to achieve global coverage of Twitch's infrastructure. To achieve this, we utilised a global network of open HTTP proxies[3] to launch the video requests from around the

[2] We distinguish unique servers based on their IP address — we note that each IP address is also allocated a unique domain name.

[3] These are servers that allow us to proxy web requests through them, thereby appearing as it our requests come from them: `https://incloak.com/`

world. We validated that the client IP address exposed to the server is the proxy address, thus we can expect the Twitch server to redirect based on the proxy location. In total, we routed through 806 proxies, from 287 ASes located in 50 countries from Europe (154), Asia (372), Africa (24), Australia (4), North America (138) and South America (114). Though there are several limitations with using open proxies (*e.g.,* unevenly distributed locations and no accurate feedback of the video streaming latency), we argue that the proxy platform provides sufficient information on Twitch infrastructure at scale.

We observed that Twitch frequently redirects a client to different servers when requesting the same channel multiple times, thus evidencing some mechanism of load balancing. For each channel we sent the request multiple times from each proxy in order to comprehensively sample the servers offered from that location. Each channel was requested a variable number of times (from 15 to 300) based on how many unique servers our queries discovered. We first ran the crawler for 5 months from December 2015 to April 2016. We continuously launched requests to all online channels listed from public Twitch API,[4] and collected over 700k requests indicating the Twitch servers that clients in that region are redirected to.

Once we acquired the list of Twitch servers, we began to explore the strategy that maps streams onto servers. First, we requested all online channels via proxy servers in the countries in which Twitch servers are located; also each channel was requested multiple times to discover as many servers hosting the stream as possible. Second, we carried out the same experiment for around 30 selected popular channels every 5 minutes. This was done to observe how the most popular channels are managed over an extended period of time. A total of 1m requests were collected from these two experiments.

Finally, to further understand Twitch's client redirection strategy on a global scale, we also requested all online channels through all proxies one-by-one. We then captured which server each proxy is redirected to. For each proxy, we requested the channels only once to emulate a typical client. This resulted in a further 1m requests collected between April to June 2016.

3 Geographic Deployment of Twitch Infrastructure

We start the exploration of Twitch's infrastructure by describing the locations of its servers, as well as how they are connected to the Internet. Our logs show that all Twitch video streams are served from hls.ttvnw.net subdomains. Each domain consists of a server name with an airport code, hinting at a geographical location. For example, video11.fra01.hls.ttvnw.net is a server in Frankfurt (fra), Germany. We confirmed that there is a one-to-one mapping between each domain and an IP address by performing global DNS queries from locations around the world. In total, we discovered 876 servers distributed over 21 airport code subdomains from 12 countries.

[4] https://github.com/justintv/Twitch-API

It is unclear how accurate these location-embedded domains are and, therefore, we compare the airport codes against the locations returned by three IP geodatabases: ipinfo.io, DP-IP and Maxmind GeoLiteCity. Although the airport locations embedded within the domains are always in the same continent, we note that they are inconsistent with the locations returned from the databases. Instead, the geodatabases report that Twitch operates a centralised infrastructure. All servers were mapped to just 4 countries: Switzerland (Europe), Hong Kong (Asia), US (North America) and Sydney (Oceania). In total, our traces reveal 360 servers in the North America (NA), 257 servers in Europe (EU), 119 in Asia (AS) and 47 in Oceania (OC).

To explore the discrepancy between the databases and airport codes, we performed a TCP-based traceroute and ping campaign from 10 sites in East and West US, Europe, Asia Pacific and South America. From the traceroute path we see that servers sharing a prefix also pass through the same router when entering Twitch's AS, with only the last three hops differing. This, however, does not confirm physical locations. Hence, we also check the Round Trip Time (RTT) to each server using TCP ping. This shows a clear boundary between servers with different airport codes. Servers inside the same sub-domains tend to differ by under 5ms; for servers on the same continent, the difference is within 50ms; for servers on different continents, this increases beyond 100ms. We found a minimal RTT of under 3ms when accessing servers sharing the same country code. This suggests that the airport country codes are a good indicator of physical location. In othe words, this highlights inaccuracy in the geolocation databases (this is perhaps reasonable, as geodatabases are well known to suffer limitations such as address registration [11]).

We gain additional confidence in our findings by checking the BGP routing tables.[5] Unlike other large content providers, we fail to find any third party hosting, as seen in other larger CDNs like Google [11] or Netflix. Instead, all servers are located within Twitch's own Autonomous System (AS46489). Importantly, we find the prefixes are only announced in their appropriate continents. For example, 185.42.204.0/22 is only announced in Europe and 45.113.128.0/22 is only announced in Asia. Thus, we are confident that the geolocations are at least accurate on a continent-level granularity

Finally, to dig deeper into the BGP interconnectivity of Twitch's AS, we utilise PeeringDB [2] to extract the locations of advertised public and private peering facilities used by the 153 Twitch peers listed in [1]. Fig. 1 presents the number of *potential* peers that are collocated with Twitch in Internet Exchange Points (IXPs) and private peering facilities. Unsurprisingly, we find a tendency for more peering in countries where we also discover Twitch servers. For example, most of the *potential* peerings are located in IXPs in the Netherlands (AMS-IX), US (Equinix), UK (LONAP) and Germany (DE-CIX Frankfurt). Noteworthy is that the number of *potential* peerings in Asia is actually quite small, with the bulk in America and Europe (we acknowledge this could be caused by inaccu-

[5] http://routeserver.org/

racies in PeeringDB). We find from BGP route records[6] that the IP prefix for the Asia presence was first advertised in June 2015. This recency could explain the low number of peers. The same is for Oceania, which first was advertised in November 2015. The low number of peers could affect the performance in redirection, as we will illustrate later in §5.

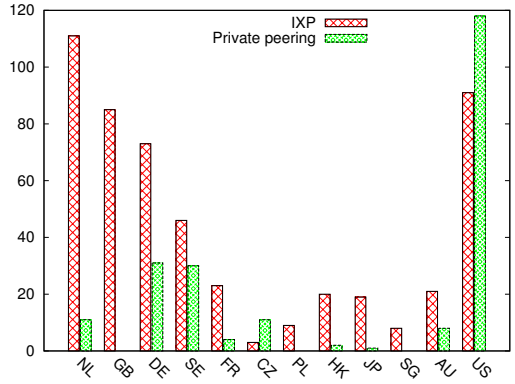

Fig. 1: Number of peers collocated with Twitch AS46489 at Internet Exchange Points and private peering facilities in each country (from PeeringDB). There is more peering in countries where Twitch servers are based.

The above results only allow us to definitively state that geolocations are accurate on a per-continent basis. Hence, for the rest of this paper, we focus our analysis on *continent-level* geolocation; where countries are mentioned, we use airport codes as the ground truth. Due to the low utilisation of Oceania servers, we will mainly focus on NA, EU and AS in the following sections.

4 Stream Hosting Strategy

The previous section has explored the location of Twitch's infrastructure. However, this says little about how it is used to serve its dynamic workload. Next, we look at how streams are allocated to Twitch's servers.

4.1 How important is channel popularity?

We first look at the number of servers a channel is hosted on, based on how many viewers it receives (*i.e.,* popularity). It might be expected that the number of servers hosting a channel scales linearly with the number of viewers. However, we find this is not the case for Twitch. Fig. 2 presents the number of servers hosting a channel against the instant number of viewers per channel. Live viewer

[6] https://stat.ripe.net/

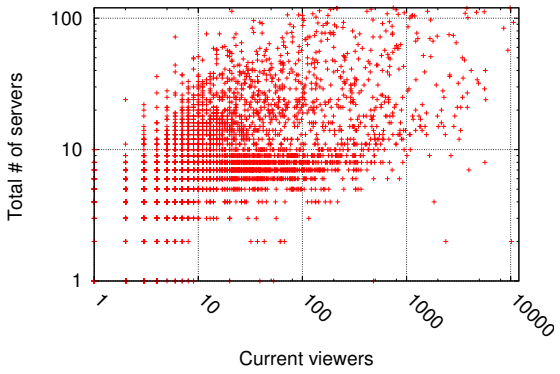

Fig. 2: Number of unique servers hosting each channel (found using requests from multiple vantage points all over the world) against number of current viewers. Channels with high view counts are replicated on a larger number of servers.

figures are acquired from the Twitch API. Although there is an upward trend, it is not that distinct (highest correlation is just 0.41). We also explored the total number of viewers (accumulated viewers over time), however the correlation with number of servers was not higher.

The low correlation suggests a more sophisticated methodology is used to manage the scaling — it is not solely based on the number of viewers. To understand this better, we take a temporal perspective to see how the number of servers utilised for a channel evolves over time. We manually selected 30 popular streamers from different countries and repeatedly requested their channels every 5 minutes from the proxies.

Fig. 3 presents example results from a US streamer and a Chinese streamer. Both channels have an initial allocation of 3 servers when they start the streaming session. As more viewers join, the popularity is followed by an increase in the number of servers provisioned by Twitch. The figure also shows how drops in viewing figures are accompanied by a decrease in the number of servers. When looking at the number of servers per continent, it can be seen that the capacity is adjusted independently per region, with the Chinese streamer having only 3 instances in Europe and America. Again, this confirms that Twitch scales dynamically the number of servers allocated to a channel, depending on the view count. Moreover, it indicates that each region is scaled independently based on the number of viewers in that region.

4.2 Scaling of Servers Across Continents

The previous section shows that the number of servers hosting the channel is correlated with the number of viewers watching the channel per region. We next investigate how the scaling works across continents. Fig. 4 presents the fraction of servers found in each continent for each channel (based on its number of

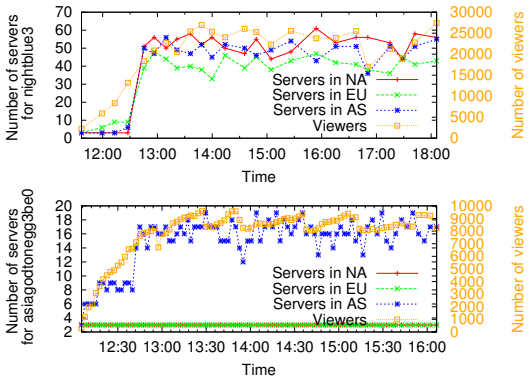

Fig. 3: (a) Number of servers found for channel nightblue3 (US streamer) as a timeseries; (b) Number of servers found for channel asiagodtonegg3be0 (Asian streamer) as a timeseries. The number of servers are scaled independently in each region.

viewers). We present both the bottom 70% and top 10% of all channels during one snapshot.

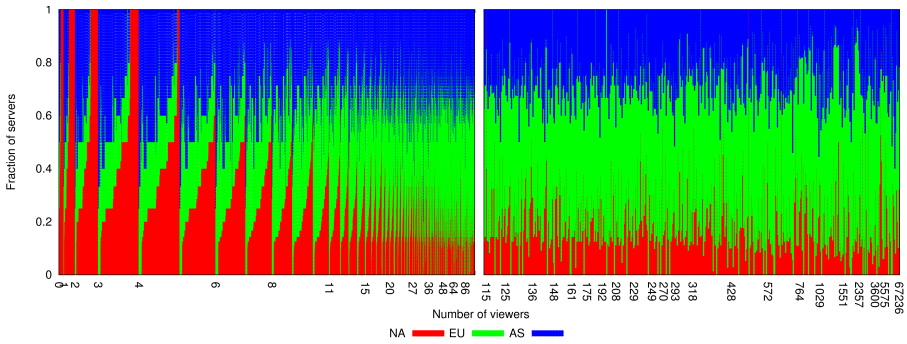

Fig. 4: Fraction of servers found from NA, EU and AS cluster for the bottom 70% (left) and top 10% channels (right). Only popular channels are replicated outside of NA

We can see from Fig. 4 that channels with a small number of viewers tend to be predominantly served from NA only (red). 67% of channels with 0 viewers are exclusively hosted in the US; this drops to 63% for 1 viewer, 48% for 2 viewers, 40% for 4 viewers, and just 24% for 5 viewers. As the number of viewers increases, the fraction of US servers hosting the stream decreases (to be replaced by both EU and AS servers). Channels with over 50 viewers are nearly always

served from all three continents. Fig. 4 also shows the server distribution of the top 10% channels, with 21% of servers in NA, 53% in EU and 26% in AS overall.

Briefly, we also see distinct patterns within each continent. For example, in NA, channels are always first hosted in San Francisco (sfo) before being scaled out to other server locations in the region. The same occurs in EU and AS, with Amsterdam (ams) and Seoul (sel) usually hosting a stream before other continental locations.

5 Client Redirection and Traffic Localisation

The previous section has shown that Twitch tries to adapt to the global demand by progressively pushing streams to multiple servers on multiple continents. In this section, we explore the mapping of clients to these regions by utilising our full set of proxies. We perform a full channel crawl from each location, and see where the clients are redirected to (*cf.* §2). Table 1 provides a breakdown of the redirections between different continents. In the majority of cases, Twitch assigns a server from the nearest continent: 99.4% of the requests in North America and 96% of requests in South America are handled by servers in NA; 82% of the requests in Europe and 78.2% of the requests in Africa are served by EU servers.

Table 1: Traffic distribution of Twitch clusters globally.

Fraction(%)	NA cluster	EU cluster	AS cluster
North America	99.4	0.6	0
South America	96	4	0.01
Europe	17	82	1
Africa	21.8	78.2	0
Asia	34.4	20	45.6

Our results also contain some noticeable outliers. Asian servers handle only 45.6% of requests from Asian clients; more than one third of the requests are handled by NA servers. That said, the NA cluster also absorbs the vast majority of requests from other regions that are not resolved to their local servers, including AS and EU. In order to explore the reasons behind this apparent mismatch, we investigate for each proxy the fraction of redirections to its local (continental) servers when requesting the full list of channels. Fig. 5 shows the empirical CDF of the fraction of local servers observed by each proxy. We separate the plots into each continent for comparison. A clear contrast can be seen among the three different regions: nearly 90% of the clients in North America are always served by NA servers; and almost 40% of the clients in Europe are always served by EU servers. However, for Asia, 50% of the clients are never served by the Asian servers, and only 10% are entirely served by Asian servers.

As previously noted, the number of servers that host a stream is closely related to the stream's popularity. Hence, we also inspect the relationship between

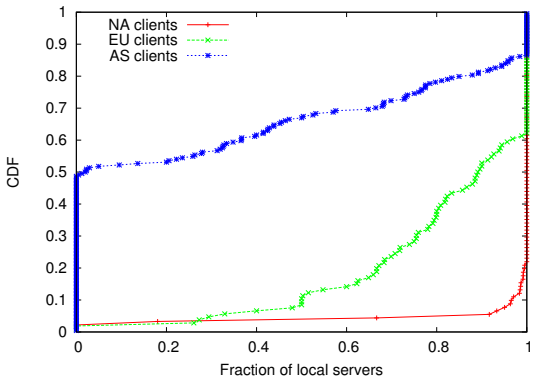

Fig. 5: Fraction of local servers observed for each proxy. Clients are grouped by continents for comparison. NA users are usually served locally, whereas most AS clients must contact servers outside of AS.

channel popularity and the ability of clients to access streams from their local cluster. Fig. 6 presents the fraction of requests that are redirected to a cluster on the same continent, plotted against the popularity of the channels. Again, it can be seen that European clients get far more local redirects, whilst Asian requests regularly leave the continent. This is consistent across all channel popularities, although in both cases, more popular channels receive a large number of local redirects.

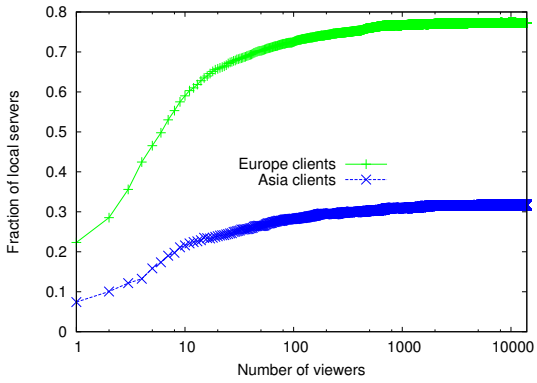

Fig. 6: The fraction of local servers used *vs.* the total number of viewer for a channel. More popular channels are more likely to be locally available on a continent.

An obvious question is why do the Asian clients suffer from such poorly localised redirects. Only 15% of our Asian clients exclusively utilise Asian servers; 50% are *never* redirected within Asia. To analyse why this might be the case, we revisit the peering policies of those particular networks. When inspecting the 15% of Asian clients that exclusively rely on Asian servers, we see that they all share the same private peering facilities with Twitch (based on PeeringDB). For example, AS36351, AS9381 and Twitch are all registered in Equinix, Hong Kong. In contrast, the remaining networks do not peer. Therefore, it is likely that Asia fails to localise its requests because of these poor existing peering arrangements (§3). Even if the servers in Asia are geographically nearby, their network distance might be higher. Similar scenarios can be found in previous work [14], highlighting that topology and peering is far more important than geographic distance.

6 Related Work

Live video streaming is challenging due to the size of video content and the time constraints involved. Various architectures have been developed to support these challenges. Peer-to-Peer (P2P) video streaming has emerged as one promising solution, leveraging the resources of end users. For example, LiveSky [24] and PPLive (CoolStreaming [23]) are two examples of deployed systems, relying on P2P assistance. Other approaches rely on cloud assistance; Chen *et al.* used Amazon Cloud, Microsoft Azure and Planetlab nodes to build an elastic system to support various loads in live video streaming [12].

To date, this is the first work revealing the content delivery infrastructure of Twitch; we believe this could be very influential when designing future Twitch-like systems. That said, there has been a wealth of work looking, more generally, at content delivery infrastructures in Video on Demand and live video streaming. For example, in [8], the authors use PlanetLab nodes to measure YouTube's infrastructure. They found that YouTube uses many different cache servers hosted inside edge networks. Torres *et al.* [20] captured traces from a campus network, showing that the server selected in the YouTube CDN is usually the closest one to the user. There has also been work looking at various other systems, *e.g.,* Netflix [10,7], YouPorn [21] and Hulu [6]. However, whereas previous work has focussed on platforms in which static (*i.e.,* non-live) content is being delivered, Twitch suffers from far greater constraints due to its live real time nature (making caching redundant). Critically, Twitch is the first major platform to employ *user generated* live video streaming. In our past work [13], we explored the nature of channel and game popularity to confirm the significant scale of Twitch (channel peaks exceeding 1.2 million viewers).

7 Conclusion

In this paper, we have studied Twitch as an example of modern user generated live streaming services. We have made a number of findings, which reveal how

Twitch's infrastructure differs from traditional "static" streaming platforms like YouTube. Through empirical measurements, we have shown that Twitch operates a much more centralised infrastructure — in a single AS with POPs on four continents (compared to the thousands used by YouTube). This is likely because the benefits of using highly decentralised caches are less than for that of live streaming (as time-shifted caching cannot take place for live streams). These design choices naturally lead to a different scale-up strategy to that of content delivery networks like YouTube, which typically rely on reactive caching. Driven by the delay sensitivity of live streaming, Twitch progressively and proactively replicates streams across servers only after sufficient demand is observed. Critically, this occurs on a pre-region basis, dynamically replicating streams based on *local* demand. This more centralised approach places a much greater reliance on effective peering and interconnection strategies (as Twitch does not place caches inside other networks). We observed the challenges this brings in Asia, where clients were redirected to NA due to poor local interconnectivity with Twitch's AS.

Although Twitch is only one example of user generated live streaming, we believe its scale and success indicates that its architecture could be an effective design choice for other similar platforms. Hence, there are a number of future lines of work that can build on this study. We are interested in exploring a range of system improvements for Twitch-like platforms, including a more sophisticated control plane that redirects on several factors, expanding their multicast design, introducing peer-to-peer techniques, or addressing issues with peering. We would also like to expand our study by measuring realtime streaming performance and comparing with other platforms, such as YouTube's recent gaming service. Only through this will it be possible to evaluate the best architecture(s) for future user generated streaming platforms.

References

1. AS46489 Twitch.tv IPv4 Peers. `http://bgp.he.net/AS46489#_peers`
2. PeeringDB - AS46489 Twitch.tv. `https://www.peeringdb.com/net/1956`
3. Twitch. `https://www.twitch.tv/`
4. Twitch is 4th in Peak US Internet Traffic. `https://blog.twitch.tv/`
5. Twitch: THE 2015 RETROSPECTIVE. `https://www.twitch.tv/year/2015`
6. Adhikari, V.K., Guo, Y., Hao, F., Hilt, V., Zhang, Z.L.: A tale of three cdns: An active measurement study of hulu and its cdns. In: Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on. pp. 7–12. IEEE (2012)
7. Adhikari, V.K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., Zhang, Z.L.: Unreeling netflix: Understanding and improving multi-cdn movie delivery. In: INFOCOM, 2012 Proceedings IEEE. pp. 1620–1628. IEEE (2012)
8. Adhikari, V.K., Jain, S., Chen, Y., Zhang, Z.L.: Vivisecting youtube: An active measurement study. In: INFOCOM, 2012 Proceedings IEEE. pp. 2521–2525. IEEE (2012)
9. Ahmad, S., Bouras, C., Buyukkaya, E., Hamzaoui, R., Papazois, A., Shani, A., Simon, G., Zhou, F.: Peer-to-peer live streaming for massively multiplayer online

games. In: Peer-to-Peer Computing (P2P), 2012 IEEE 12th International Conference on. pp. 67–68. IEEE (2012)
10. Böttger, T., Cuadrado, F., Tyson, G., Castro, I., Uhlig, S.: Open connect everywhere: A glimpse at the internet ecosystem through the lens of the netflix cdn. arXiv:1606.05519 (2016)
11. Calder, M., Fan, X., Hu, Z., Katz-Bassett, E., Heidemann, J., Govindan, R.: Mapping the expansion of google's serving infrastructure. In: Proceedings of the 2013 ACM Conference on Internet Measurement (IMC '13). pp. 313–326. ACM (2013)
12. Chen, F., Zhang, C., Wang, F., Liu, J., Wang, X., Liu, Y.: Cloud-assisted live streaming for crowdsourced multimedia content. Multimedia, IEEE Transactions on 17(9), 1471–1483 (2015)
13. Deng, J., Cuadrado, F., Tyson, G., Uhlig, S.: Behind the game: Exploring the twitch streaming platform. In: Network and Systems Support for Games (NetGames), 2015 14th Annual Workshop on. IEEE (2015)
14. Fanou, R., Tyson, G., Francois, P., Sathiaseelan, A., et al.: Pushing the frontier: Exploring the african web ecosystem. In: Proceedings of the 25th International Conference on World Wide Web (WWW '16). International World Wide Web Conferences Steering Committee (2016)
15. Finamore, A., Mellia, M., Munafò, M.M., Torres, R., Rao, S.G.: Youtube everywhere: Impact of device and infrastructure synergies on user experience. In: Proceedings of the 2011 ACM Conference on Internet Measurement (IMC '11). pp. 345–360. ACM (2011)
16. Gill, P., Arlitt, M., Li, Z., Mahanti, A.: Youtube traffic characterization: a view from the edge. In: Proceedings of the 2007 ACM Conference on Internet Measurement (IMC '07). pp. 15–28. ACM (2007)
17. Hamilton, W.A., Garretson, O., Kerne, A.: Streaming on twitch: fostering participatory communities of play within live mixed media. In: Proceedings of the 32nd annual ACM conference on Human factors in computing systems. pp. 1315–1324. ACM (2014)
18. Pires, K., Simon, G.: Youtube live and twitch: A tour of user-generated live streaming systems. In: Proceedings of the 6th ACM Multimedia Systems Conference. pp. 225–230. MMSys '15, ACM, New York, NY, USA (2015)
19. Siekkinen, M., Masala, E., Kämäräinen, T.: A first look at quality of mobile live streaming experience: the case of periscope. In: Proceedings of the 2016 ACM on Internet Measurement Conference. pp. 477–483. ACM (2016)
20. Torres, R., Finamore, A., Kim, J.R., Mellia, M., Munafo, M.M., Rao, S.: Dissecting video server selection strategies in the youtube cdn. In: Distributed Computing Systems (ICDCS), 2011 31st International Conference on. pp. 248–257. IEEE (2011)
21. Tyson, G., El Khatib, Y., Sastry, N., Uhlig, S.: Measurements and analysis of a major porn 2.0 portal. ACM Transactions on Multimedia Computing, Communications, and Applications (ACM ToMM) (2016)
22. Wang, B., Zhang, X., Wang, G., Zheng, H., Zhao, B.Y.: Anatomy of a personalized livestreaming system. In: Proceedings of the 2016 ACM on Internet Measurement Conference. pp. 485–498. ACM (2016)
23. Xie, S., Li, B., Keung, G.Y., Zhang, X.: Coolstreaming: Design, theory, and practice. Multimedia, IEEE Transactions on 9(8), 1661–1671 (2007)
24. Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., Li, B.: Design and deployment of a hybrid cdn-p2p system for live video streaming: experiences with livesky. In: Proceedings of the 17th ACM international conference on Multimedia. pp. 25–34. ACM (2009)