Big Data: Real-Time Video Streaming and Log Analytic for Improving Quality of Experience

Reza S. Kalan

Digiturk beIN Media Group, Istanbul, Turkey
Email: reza.shokri@hotmail.com; reza.kalan@istinye.edu.tr (R.S.K)
Manuscript received March 1, 2024; revised April 5, 2024; accepted May 10, 2024; published June 3, 2024.

***Abstract*—Client-side adaptive bitrate algorithms are designed to optimize human-perceived Quality of Experience (QoE). However, network heterogeneity at the edge makes it difficult to provide the same video quality to all end users. Even the best Content Delivery Networks (CDNs) or Internet Service Providers (ISPs) have poor quality in certain regions or times of the day. In addition to network dynamism, online clients continuously switch between video channels that stream via different CDNs. The volume of video logs and network dynamics make it very difficult to analyze client-side video quality or monitor network performance and thus make timely decisions. The concept of big data analytics is a successful and cost-effective data mining tool and application that offers deep analytics, high agility, and massive scalability with low latency. Recently, with the advent of distributed computing technologies, the analysis of big video data in the cloud has attracted the attention of researchers and practitioners. Resource-rich edge or cloud servers have become popular destinations for video streaming and log analytics. In this paper, we discuss the change in the requirements for video streaming and illustrate the difficulty of big data log analytics at the edge. We then show the advantage of log analytics in the cloud and its impact on improving users' QoE and reducing CDN traffic distribution costs by detecting and removing illegal streaming along with CDN switching.**

***Keywords*—big data, log analytic, adaptive video streaming, quality of experience**

I. Introduction

Nowadays, the IP network has become part of the critical infrastructure of our businesses. With the rapid development of smart devices, the end user can play video anytime and anywhere. The popularity of video streaming brings new business fashion; therefore, all video service providers have started big data analytics. Due to the volume, variety, and velocity of big data, retrieving information in a short period that is suitable for our decision-making becomes a concern [1]. Big data analytics technologies are designed to extract value from big data in trustworthy, fast, and economical ways. Some of the most significant potentials of big data come from the bridge between resource and data streams. Cloud services (e.g., Amazon and Azure) strive to provide the robustness, scalability, and reliability necessary for big data analytics [2] [3]. Regardless of the infrastructure, cloud services are used to improve the delivery of big data to the end user.

HTTP Adaptive Streaming (HAS) technology is widely used in video streaming to select the appropriate bitrate that matches the network bandwidth. Users connect to the network using various devices and through different networks and play online videos. This heterogeneity of networks and devices makes it difficult to provide high-performance streaming to end users. Meanwhile, in a competitive marketplace, Over-the-Top (OTT) companies try to understand customer behaviors and offer customized services by mining and exploring hidden relations between different attributes. To this end, we need to find new ways to interact with big data.

Real-time streaming analytics has some significant features that help to better decision making that include i) Analyze and take action on real-time data ii) Risk analysis before they occur. iii) Predict new business opportunities and revenue streams by analyzing end-user behavior [4]. Due to on premise system limitations, distributed systems are used to store and process large data. The scalability and flexibility of cloud analytics solutions have become popular destinations to solve limitations of on-premise resources, but there are still challenges. Although cloud solutions are cost-effective for short tasks, it is more costly for long-term operations. Another problem is the data source. Data is an asset and must be purchased. The bottom line is, that to provide better live video service, video service providers must have complete insight into Content Delivery Network (CDN) performance at the edge and delivery network levels, as well as the quality of video experienced by the end user. Nevertheless, the nature of video makes it difficult because of the volume and velocity of logs. (e.g., 2 GB raw logs for a minute video length).

Previous works such as [5–7] discussed comprehensive architecture that has been proposed for intelligent video analytics in the cloud. However, none of them discuss the actual effects of the application on end-user quality and video provider decision-making in the real world. The motivation behind this study is to improve users' Quality of Experience (QoE) with data analysis and real-time reaction. Typically, CDNs report data logs with delays of almost 15 minutes. However, by analyzing raw logs that are delivered by CDN logs stream services it is possible to fast log analyses and adapt to react in less than 3 minutes.

The remainder of the study is organized as follows: background and system architecture are introduced in Section II. The system architecture is discussed in Section III. The case study and experimental results are discussed in Section IV. In Section V, we conclude the paper and draw future paths.

II. Background and Related Works

A. Background

HTTP Adaptive Streaming: As shown in Fig. 1, in the HAS technology, a single video file encoded at multiple bitrates called "representations" enables smooth adaptation between different video qualities during display time [8]. In adaptive streaming, a player needs to download the entire video file

before starting to play. Toward a fast start, a video file is divided into small 'segments' (or chunks). A segment generally contains less than 10 seconds of video. This architecture benefits from pull technology, where distributed caches on the CDN store contents at the edge points, and clients connect to the edge and download video content.

Content Delivery Network: Using the flexibility of CDN, combined with edge caching features, allows for enhancing the quality of video streaming. Clients connect to the CDN network at the edge and adapt to a suitable bitrate considering network throughput or buffer fullness. Serving client requests at the CDN edge reduces the load on the original media servers by eliminating sending client requests to the origin server. This offloads the origin server and reduces network traffic. As a result, it improves the response time of the origin server and reduces the download time.

Client Adaptation Mechanism: The intention of adaptive streaming is a smooth adaptation between different video qualities where the network download speed changes during the display time. To this end, a video file is encoded with different bitrates (representation) and divided into small-size video segments. Manifest file carries video file information including the number of representations, number of video segments, segment length, audio, and subtitle information. As shown in Fig. 2, clients connect to the nearest CDN-edge and start video streaming by requesting an HLS playlist or DASH manifest file. Meanwhile, clients continuously detect the network performance (download speed) with the Adaptive Bitrate (ABR) algorithm.

Network heterogeneity at the edge makes it difficult to provide the same bandwidth for all clients. Therefore, each client requests video segments sequentially considering the network throughout and manifest file information. Suppose a client requests segment 's' with bitrate 'b' adaptively. If the request (s, b) has already been requested by another client and is available in the CDN cache, the requested video segment is returned to the client by CDN-edge. Otherwise, the client's request is sent to the origin server, which is generally located in a different geography and consequently leads to a slower response due to a larger Round-Trip Time (RRT).

The origin server does just-in-time packaging (s , b) and sends it to the client via CDN. CDN forwards the received video segment to the client and meanwhile caches them for future incoming requests. This has two advantages: i) fast response time, and ii) offloading the origin server. On the client side, to avoid premature buffering, the video player fills the buffer with downloaded video segments up to a predefined threshold value (for example, 10 seconds of video) and then starts playing the video. In the case of rebuffering, the client starts requesting lower bitrate video segments that are small in size (but poor in quality) and can be downloaded faster.

B. Related Work

The high data rate, low delivery latency, and scalability are the Key Performance Indicators (KPI) of adaptive streaming technology, widely used by video management and service providers for live or on-demand video streaming [9]. An overview of distributed networks and clients' experiences helps provide end users with better video service. Such insight needs real-time log analytics. However, due to the data intensive and resource-hungry nature of large-scale video logs, processing and extracting insights from the video log quickly is a challenging task. With the popularity, resource flexibility, and scalability of cloud computing, big data video analytics has become a reality.

Most recent studies [10–13] focus on resource utilization rather than human-perceived quality optimization. Unlike end-user quality experience, video analytics algorithms can tolerate dropped frames and decayed image quality, which are very important from a QoE perspective. Zhang *et al.* [10] proposed a configuration adaptive streaming framework designed for live video analysis. The proposed model trains a deep reinforcement learning that makes no assumptions about the environment but learns to make configuration choices through its own experiences. Canel *et al.* [11] proposed "FilterForward" an edge cloud desertion method to reduce bandwidth use by an order of magnitude without sacrificing accuracy. Both proposed methods use Deep Neural Networks (DNN) to extract general features of video frames. Like previous works, a neural network solution is used in [12] for enabling edge-cloud video analytics for robotics applications rather than human-perceived adaptive streaming. Zhang *et al.* [13] introduced 'Awstream' adaptive streaming analytic uses network throughput and behaves cautiously to avoid building up queues: automatically learns a Pareto-optimal policy or strategy for when and how to invoke different degradation functions. However, the proposed solution is more suitable for IoT devices rather than adaptive streaming.

Many recent efforts have been made to improve QoE parameters for ABR. However, most of them are designed as adaptive algorithms or performed in limited input and unrealistic domains. A QoE for adaptive video streaming is discussed in [14], where authors build a database dedicated to the subjective evaluation of HAS videos under realistic conditions. A good study related to human-perceived quality is discussed in 'Pensieve' [15]. Pensieve uses modern Reinforcement Learning (RL) techniques to learn a control policy for bitrate adaptation purely through experience. However, Pensieve is an ABR algorithm running in clients and optimizes its policy for different network characteristics and QoE metrics directly from experience. There is no mechanism for changing CDN or edge points when the network quality drops significantly. Recall, even the best CDN or ISP has poor quality in certain regions or times of the day.

III. System Architecture

In this section, video streaming technologies as well as the proposed system architecture are discussed.

A. Architecture Design and Implementation

- *Origin Side:* The origin server is the source of the video content, where the video files are encoded and packaged Just-in-Time for delivery. Typically, a video file is compressed by using H.264/AVC or H.265/HEVC codec, then ingested in a packager. Packager is responsible for creating different video streaming formats including Apple HTTP Live Streaming (HLS) [16], Microsoft Smooth Streaming (MSS) [17], and Dynamic Adaptive Streaming over HTTP (DASH) [18].

OTT uses a combination of multiple streaming technologies to satisfy the experience of their connected heterogeneous clients [19, 20], which complicates log analysis tasks.

- *CDN Side:* Our content distribution system relies on a multi-CDN architecture, where different CDNs are used to deliver video content to end users connected to edge points. However, each CDN has a specific log format. Therefore, defining a common template for mapping all CDN formats improves system efficiency.
- *Client Side:* Satisfying legitimate clients is the majority of client-side data analytics. In the context of client satisfaction, data analysis helps us provide better video services by detecting playback problems and troubleshooting.

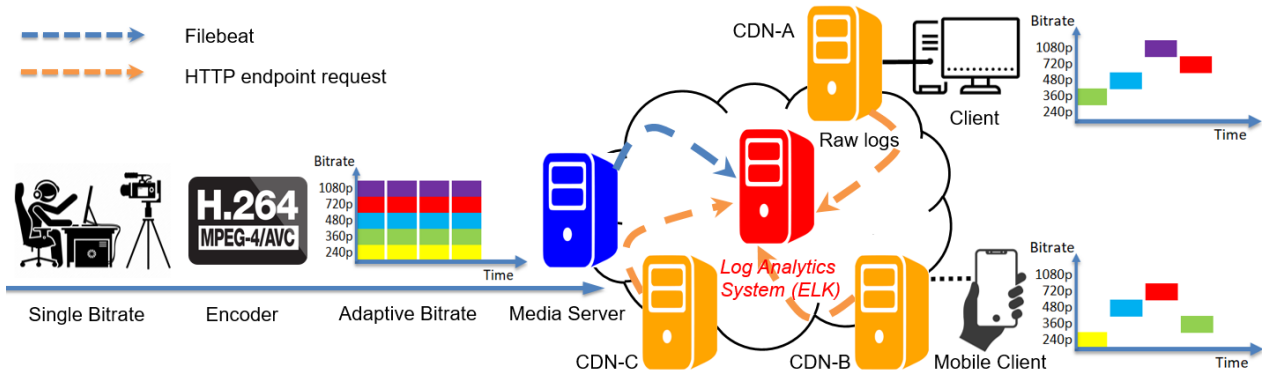

Fig. 1. Conceptual view of adaptive live streaming and log analytics system – life cycle.

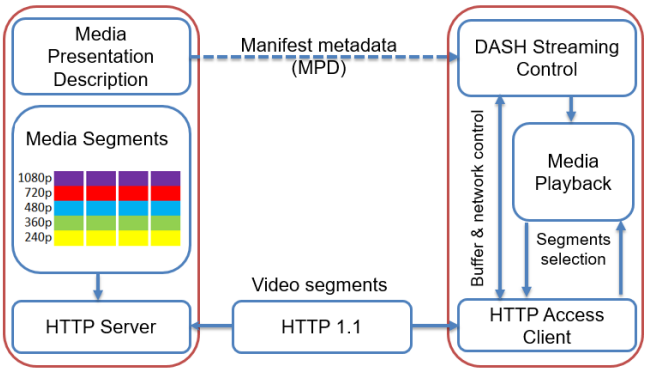

Fig. 2. Dynamic Adaptive Streaming over HTTP (DASH).

B. Log Analytic System

Log analytics is the process of gathering, investigating, and visualizing the data produced by computer systems. Real-time streaming analytic tools are critical for video log analysis where thousands of heterogeneous clients are connected to the Internet and display video content. Log content includes all involved end-to-end components information including, but not limited to the origin server, video, telecom, network, CDN, and client information. To better understand the volume of video logs, only consider video metadata information for three different display formats. Each video stream has at least manifest, video, audio, and subtitle files. Let's say the length of the video segment is 2 seconds. Therefore, the media server receives 4 requests (manifest, video, audio, subtitle) per segment per client. That means 2 requests per second or 120 requests per minute. Each request contains detailed information about the client, network, CDN, etc. Thousands of online users stream videos in different formats through worldwide heterogeneous networks during the live event.

The characteristics of video streaming logs (volume, velocity, and variety) make it difficult to send, receive, and process them in real time. Therefore, in the absence of sufficient resources, even processing a part of the logs helps to make better decisions. Products such as Amazon Kinesis and Azure Stream Analytic are widely used in the industry. For this purpose, we leveraged role-based Elastic log analytic tools. We leveraged the ELK ecosystem which is easy to use, scalable, and flexible. This ecosystem embers three open-source projects:

- *Elasticsearch:* is a prominent search and analytics engine.
- *Logstash:* is a server-side data processing pipeline that simultaneously ingests data from multiple sources, transforms, and sends it to Elasticsearch.
- *Kibana:* visualize data with charts and graphs in Elasticsearch.

C. ELK Ecosystem

ELK stack is a collection of Elasticsearch, Logstash, and Kibana. Elasticsearch is a full-text search and analysis engine. Logstash is a log aggregator that collects data from multiple sources, processes, transforms, and sends it to various destinations, such as Elasticsearch. Finally, Kibana provides a user interface that allows users to visualize, query, and analyze their data through graphs and charts. ELK is a log management platform that works by collecting massive amounts of log data from different endpoints in one place and then searching, analyzing, and visualizing it in real time. ELK's most common use cases include monitoring, troubleshooting, security analysis, and fraud detection.

Fig. 3 shows an abstract view of the data log analytic system and behavior of the ELK stack in collecting summarization and visualization of logs coming from different endpoints. End users connect to CDN-edge from different geolocations using different devices, each playing different video formats. This is where the ELK stack shines in collection, aggregation, analysis, visualization, and monitoring. Logstash is a log aggregator and processor that reads data from many sources (such as a CDN) and sends it to one or more destinations for caching or storage (such as Elasticsearch). Logstash has a rich library of plugins, allowing it to collect, convert, and enrich various log types,

from error logs, app logs, system logs, and web server logs.

The flexibility, distributed, and multi-tenancy nature of Elasticsearch provides a scalable, near-real-time search solution. Elasticsearch fetches and stores unstructured data from various locations, indexes it based on a user-specified mapping, and makes it searchable. While the ELK stack is deployed and running in the cloud, there are technically no limitations in terms of storage, processing, and network bandwidth. Therefore, the scalability of Elasticsearch allows a scale-up system on demand. ELK's simplicity and robustness make it possible to manage massive volumes of data and easily scale on a cloud platform without sacrificing performance.

The Beats family including *Filebeat*, *Metricbeat*, and *Packetbeat* are lightweight tools for collecting and shaping logs, metrics, or network data. They sit on servers or are deployed as functions and then centralize the data in Elasticsearch. In the proposed system, filebeat includes modules that simplify the collection, analysis, and visualization of data from origin media servers which are installed in different data centers. This is achieved by combining automatic default routes based on the operating system (NGINX), with Elasticsearch and Kibana dashboards.

Kibana shapes any data (structured and unstructured) indexed in Elasticsearch. Kibana is preferred for visualizing large volumes of logs stored in Elasticsearch, which sit on top of Elasticsearch and Logstash in the ELK stack. Dashboard features, various interactive charts, efficient shaping, and visualization provide optimized search experiences.

D. Cloud-based Log Analytics

As illustrated in Fig. 4, integration of cloud computing and log analytic systems offers numeric advances, addressing the challenges associated with big data analytics. Some of the key benefits of this integration include:

- *Scalability:* Scalability is critical for video streaming platforms where hundreds of thousands of users simultaneously connect to media servers and display video. It is imperative to provide an efficient service to multiple users connected through a heterogeneous network and different devices. However, collecting, transforming, and summarizing such a big volume of log data is difficult and takes more time, while video streaming services require fast and real-time response. Taking advantage of cloud features, including distributed processing and providing sufficient storage space and bandwidth, makes it possible to scale the system based on demand.
- *Flexibility:* The cloud offers different classes of services and tools that allow the development of frameworks, infrastructures, and environments suitable for specific needs.
- *Cost Effectiveness:* Cloud-based log analytic platforms facilitate scaling systems with optimal cost by leveraging the pay-as-you-go model and cost optimization strategies.

The ELK ecosystem is a combination of multiple servers running on the cloud where logs are collected from each CDN's endpoint. We dedicated and configured a separate log analysis server for each CDN as well as the origin servers. Raw logs are parsed and summarized into valuable pieces of information, taking into account KPI metrics. We measure different KPIs from different end-to-end points to provide high-quality services to end users. On the CDN side, KPI parameters include QoS metrics (e.g., network bandwidth, latency, first-byte response, etc.) and QoE metrics (e.g., buffering time, startup delay, received bit rate, quality oscillation, etc.). On the origin media server side, we monitor server performance including the number of incoming requests, outgoing bandwidth, CPU performance, read/write speed, etc.

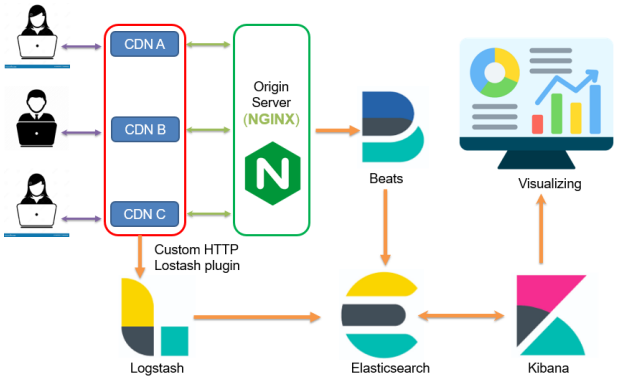

Fig. 3. ELK and data log analytic architecture overview.

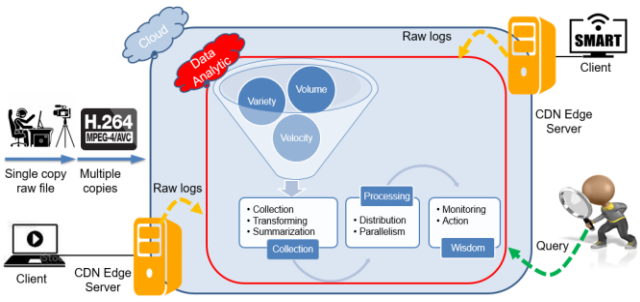

Fig. 4. Abstract architecture view of integration ELK stack and cloud platform.

IV. Case Study and Experimental Results

A. CDN Switching

The distributed nature of a CDN helps content be delivered to end users in a faster and more efficient manner. However, depending on the geographic location and the daytime, CDNs can perform differently. Leveraging a multi-CDN strategy has several advantages:

- *Reliability:* Technically, there is no unlimited capacity (storage, bandwidth, PoP, and upstream data center) for CDNs that provide large business services. Therefore, unexpected peaks in traffic may overwhelm the CDN, which can negatively impact the end user's QoE. In multi-CDN architectures, traffic is redirected to another CDN if a problem occurs.
- *Flexibility:* Even the best CDN performs well in some times and regions and not so well in others. The OTT providers can switch to the best-performing CDN considering this fact. Remind that switching to another CDN has an extra cost (e.g., increasing the re-buffering ratio for a short period).
- *Cost Efficiency:* In a competitive market, CDN providers may have variable prices for different regions

depending on the services and features offered. Content type, volume, and traffic (regular or peak traffic) are other important factors that can affect pricing.

- *Security:* Security is another case that we can consider for CDN switching. In addition to general protection against hacking and unauthorized access to media servers, content protection, and illegal streaming are other concerns. To protect the content, content providers use a combination of multiple Digital Rights Management (DRM) and watermarking technologies [9]. The CDN must support all necessary security features.

B. Fraud Detection

Identifying and reacting to changing operating conditions is critical for OTT providers. Any subtle differences in the underlying media server's operating conditions or network bandwidth usage can indicate unacceptable levels of application risk. Live video applications are more sensitive to latency, and an increase in fraud users leads to more network traffic. Detecting fraud and access revoking is necessary to provide better services. Logs are widely utilized to detect anomalies in modern large-scale distributed systems. The process of log analysis involves four main steps: log collection, log parsing, feature extraction, and anomaly detection.

At the media server level, the origin server is responsible for packaging incoming requests. Connecting more clients to the network means making more requests to the origin servers. Overloading the media server results in slow response time and thus greater latency. This delay also increases at the network level that carries the traffic.

At the network level, an increase in fraudulent users can significantly increase bandwidth usage. During peak times (for example, during a live event such as a football derby match) more users connect to the network at the same time and start streaming video. Even the best CDNs and ISPs have limited network resources. Thus, unexpected traffic generally results in poor service delivery. It is more evident in some regions which depends on many factors including customer density, connection types, and infrastructure network characteristics.

C. Experimental Results

In the existing adaptive streaming scheme, video players use adaptive bitrate algorithms (buffer-based or throughput-based) for QoE optimization, which inevitably fail to achieve optimal performance under a wide set of network conditions and QoE objectives [15]. Clients do not have any information about edge traffic except the bandwidth of connection points. They have no insight about other clients using the same bandwidth. Monitoring and analyzing logs coming from multiple CDNs in real time provides insight into switching traffic between CDNs. *Resource-rich edge or cloud servers have become popular destinations for streaming analytics* [10]. Edge analytics at the CDN level can be an alternative solution for moving big volumes of raw log data to the cloud. Unfortunately, providing such a service through an API leads to additional latency and is not suitable for real-time reactions. Therefore, the flexibility and scalability of the cloud are an alternative solution for analyzing raw logs.

Raw logs coming from each CDN carry two types of information i) Client logs that contain quality KPIs and deal with end-user detail information including client type, received bitrate, startup delay, etc. ii) CDN-edge information including edge response time, cache performance, ISP performance, number of connected users, traffic pattern, etc. This means that the analysis of the data logs, taking into account the client's QoE history at different CDNs, and the current traffic patterns and performance of each CDN enables the distribution of the client's video traffic to different CDNs. Unfortunately, the quality parameters decrease during the hand-off time, when forwarding a client from one CDN to another.

Typically, a client connects to the nearest CDN-edge and adapts to the appropriate bitrate according to the ABR algorithm decision. The majority of ABR algorithms are making bitrate decisions based on buffer occupancy, network throughput, or a hybrid model with a combination of both, The CDN returns the requested video segment to the client if it is cached on the CDN side (Hit scenario). Otherwise, in the case of 'Miss', forwards the client's request to the origin servers. This has two drawbacks that negatively affect QoE: i) usually, the origin server is far away from the client. Hence, slow response times can lead to a buffer drain on the client side. As a result, the user experiences buffering video on the screen. ii) On the origin server side, if more requests are received, the origin server becomes overwhelmed. Thus, it takes more time to prepare and package the requested segments. Subsequently, clients experience re-buffering (freezing video on the screen) and start requesting lower bitrate video segments for faster retrieval.

Fig. 5 shows the effects of CDN switching on client side re-buffering when some error rate increases. Note that re-buffering happens in the middle of playback when the client's buffers become empty, resulting in the video freezing on the screen. Typically, moving clients from one CDN to another CDN results in high re-buffering because of cache efficiency, which can negatively affect CDN-edge and origin response time and result in low video quality. A 'Miss' occurs by redirecting clients to another CDN-edge, where no video segments are cached. Therefore, the client's requests are sent to the origin server. Usually, the origin is far away from the client. Hence, slow response times can lead to a buffer drain on the client side. As a result, clients experience re-buffering and start requesting lower bitrate video segments for faster retrieval. Consequently, users experience low video quality on the screen. For this purpose, we consider streaming a percentage of the traffic through the second-best CDN to prefetch and warm the cache. Results show that on a real platform with approximately 200K concurrent users routing almost 10% of traffic through another CDN reduces re-buffering effects during CDN switching.

The origin server has limited resources compared to CDN which has a distributed nature and accommodates more resources. Therefore, forwarding more requests to the origin media server leads to slow response and bandwidth bottlenecks. Fig. 6 shows the effect of CDN switching on the origin server side. When a portion of the traffic is distributed through an alternate CDN, fewer requests will be sent to the origin server by switching to an alternate CDN. This is due to the prefetching and high efficiency of the cache at the

alternative CDN-edge, where requested segments are already cached and available for quick response. On the origin server side, we also noticed that disk I/O speed affects response time, which can significantly impact the overall performance. Since the origin server packages different video formats with different representations (bitrates), disk write speed is very important. This helps the origin server to respond more quickly to incoming requests.

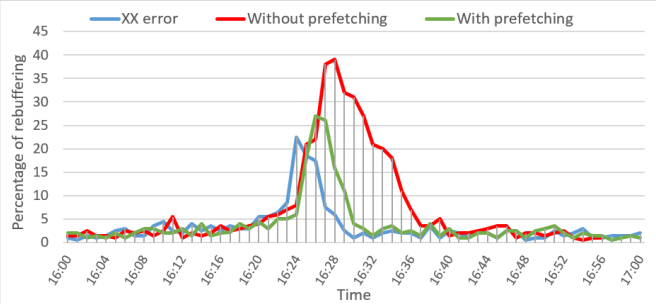

Fig. 5. Effect of CDN switching on re-buffering with/without prefetching.

Fig. 6. Requests received per second on the origin media server during CDN switching time. Warming up edge caches by a prefetching mechanism could offload origin servers in CDN switching time.

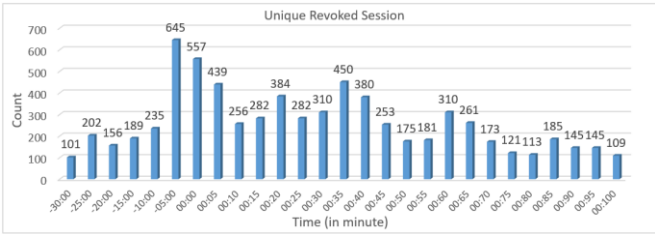

Fig. 7. Revoked unique session in the 90-minute live sports event. Exactly before starting the live event more illegal users try to use video service. This led to more requests to the edge and origin server and, consequently, more traffic at the edge, where bandwidth is shared with legal clients.

The last part of the experimental results is related to fraud detection. Illegal access to streaming video presents two main challenges. First, it fills the shared capacities of CDN resources. Even well-performing CDNs have limited resources that are shared and distributed among different customers (e.g., OTT, insurance, banking, financial services, etc.). Second, it increases billing costs. The first challenge is more critical because it directly affects end users' QoE. Regarding different stream protection mechanisms, a remarkable group of people attempts to display video in an illegal form. Therefore, detecting and revocation fraud users is essential for each OTT provider to better service delivery and improve end users' QoE. Timely fraud detection allows video service providers to pinpoint issues promptly and resolve them immediately, thereby improving system efficiency while saving revenue. Our fraud detection system uses some of the validation methods including, but not limited to:

- *Query Parameters: E*ach CDN has a specific request format. Considering this fact any request which does not meet the CDN query parameters is directly blocked.
- *User Anomaly: T*he control mechanism checks the unique user ID and blocks users who use the same ID but connect via different IPs, user agents, or user different tokens. In other words, there is no chance for a user to share account information with others.
- *Blocked List:* During real-time monitoring and observing clients' behavior, illegal sessions using different Autonomous System Numbers (ASNs), agents, and HTTP referrers are inserted into a blacklist. The blocked list progressively updates and all matched user IDs are blocked by the revoke system.

Fig. 7 shows the number of revoked sessions during a live sports event. Fraud clients start connecting to the edge servers before starting the match and this effort is boosted at the start of the event. In this example, approximately 32 K fraud sessions were detected. Overall, 6 users per second were removed from the live stream. While the total number of legal users is approximately 280 K, it means that 9% of shared bandwidth is saved. It is worth emphasizing that our system architecture is scalable, but each CDN can provide dedicated resources. For extra resource allocation, we need to proactively inform CDN.

V. Conclusion

In this study, we addressed the complexity of real-time video streaming analysis and emphasized that the implementation and execution of analytical tools on the cloud enables the achievement of results in a short time, resulting in quick decision-making and reaction. A cloud solution is easy to use and scalable on demand. We developed cloud-based ELK analytics tools to capture, transform, and summarize log streams from multiple CDNs as well as origin servers. Decoupling origin servers and CDN logs help to fast reaction and perform well in terms of providing a reliable and cost-effective video stream without compromising QoE. While each CDN has a specific log format, converting all CDN logs into a unique template is a way for quick retrieval and easy interpretation.

We implemented multi-CDN log analytics in the context of monitoring QoE and fraud detection. Our system provides an advanced online defense in depth for video analytic applications. The benefit is two-fold: i) Proving better live video service specifically in peak time considering user side KPI values including rebuffering time, received bitrate, and startup delay. ii) Saving network valuable bandwidth and reducing CDN costs by detecting fraud users and revoking them.

The current business model does not meet the requirements of intelligent multi-CDN integration. For example, legacy clients do not meet the requirements of dynamic switching between CDNs. To this end, we are revising video delivery tools, applications, and platforms. As a future work, we are planning to launch a machine learning approach to efficiently deliver content through multiple CDNs. The proposed solution benefits from content-aware encoding on the origin side and tracking and learning each client's streaming habits on the CDN side for better video

delivery. AI-based reinforcement algorithm used for multi-CDN switching considering quality, security, and cost of delivery service. This algorithm learns the performance of each client (device) on different ISPs and networks and automatically switches clients to the appropriate CDN and ISP.

CONFLICT OF INTEREST

The author declares no conflict of interest.

ACKNOWLEDGMENT

This research has been supported by the Digiturk, beIN Media Group platform (https://digiturk.com.tr/), in close cooperation with the R&D team and Istinye University (https://www.istinye.edu.tr).

REFERENCES

[1] R. S. Kalan and M. O. Unalir, "Leveraging big data technology for small and medium-sized enterprises," in *Proc. 2016 6th International Conference on Computer and Knowledge Engineering*, p. 9, 2016.
[2] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: A reference roadmap," *HCIS*, vol. 8, pp. 1–31, 2018.
[3] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, "Profitaware application placement for integrated fog–cloud computing environments," *JPDC*, vol. 135, pp. 177–190, 2020.
[4] J. Shim and K. Nisar, "Real-time streaming technology and analytics for insights," in *Proc. Conferences – Association for Information Systems*, vol. 1, 2021.
[5] A. Alam, I. Ullah, and Y. K. Lee, "Video big data analytics in the cloud: A reference architecture, survey, opportunities, and open research issues," *IEEE Access*, vol. 8, pp. 377–152, 2020.
[6] M. A. Uddin, A. Alam, N. A. Tu, M. S. Islam, and Y.-K. Lee, "Siat: A distributed video analytics framework for intelligent video surveillance," *Symmetry*, vol. 11, no. 7, p. 911, 2019.
[7] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman, "Live video analytics at scale with approximation and {Delay-Tolerance}," in *Proc. 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, 2017, pp. 377–392.
[8] R. S. Kalan and M. Sayit, "Sdn assisted codec, path and quality selection for http adaptive streaming," *IEEE Access*, vol. 9, 2021.
[9] R. S. Kalan and E. Karslı, "Illegal broadcasting: A way for protecting revenue," in *Proc. 2023 14th International Conference on Network of the Future (NoF)*, 2023, pp. 57–61.
[10] M. Zhang, F. Wang, and J. Liu, "Casva: Configuration-adaptive streaming for live video analytics," in *Proc IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, 2022, pp. 2168–2177.
[11] C. Canel, T. Kim, G. Zhou, C. Li, H. Lim, D. G. Andersen, M. Kaminsky, and S. Dulloor, "Scaling video analytics on constrained edge nodes," in *Proc. Machine Learning and Systems*, vol. 1, 2019.
[12] Y. Wang, W. Wang, D. Liu, X. Jin, J. Jiang, and K. Chen, "Enabling edge-cloud video analytics for robotics applications," *IEEE Transactions on Cloud Computing*, pp. 1500–1513, 2022.
[13] B. Zhang, X. Jin, S. Ratnasamy, J. Wawrzynek, and E. A. Lee, "Awstream: Adaptive wide-area streaming analytics," in *Proc. 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018, pp. 236–252.
[14] Z. Duanmu, A. Rehman, and Z. Wang, "A quality-of-experience database for adaptive video streaming," *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 474–487, 2018.
[15] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proc. Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 197–210.
[16] Apple. Apple http live streaming. [Online]. Available: https://developer.apple.com/streaming/
[17] Microsoft. Microsoft smooth streaming. [Online]. Available: http://www.iis.net/downloads/microsoft/smooth-streaming
[18] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, 2011.
[19] R. S. Kalan, S. Clayman, and M. Sayıt, "Vdane: Using virtualization for improving video quality with server and network assisted dash," *International Journal of Network Management*, vol. 32, no. 5, 2022.
[20] R. S. Kalan, M. Sayit, and A. C. Begen, "Implementation of sand architecture using sdn," in *Proc. 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, p. 3, 2018.