ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

cStream: Cloud based high performance video delivery network

KARTHIKA.M [1], M.MOHANAPRIYA [2]

1. P.G. Student, Dept. of MCA, VSB Engineering College, Karur, Tamilnadu, India
2. HoD, Dept. of MCA, VSB Engineering College, Karur, Tamilnadu, India

Abstract—Live streaming and video-on demand are increasing at a rapid pace. Global Over-the-top (OTT) video market is estimated to grow to $37.2 billion by 2017. However, live video streaming continues to suffer from high buffering ratios, high join times, high join failures and low average bit rates. The economic impact of these user experience metrics is huge. Recent studies have shown that traditional CDNs account for more than 20% of these join failure and bit rate degradation issues. In this paper, we present CSTREAM- a high performance cloud based live video delivery network. CSTREAM leverages the Cloud provider's global footprint, Cloud provider's high performance backbone network between different data centers, social media analytics and a UDP based fast data transfer protocol to optimize the quality of experience for end users, and the total cost incurred by the cloud provider in terms of network bandwidth and compute resources. CSTREAM allows a video broadcaster to be redirected to its closest publishing point (PP - hosted inside the "closest" Cloud data center), and then transfers the live stream at high speed using a UDP based fast protocol to one or more receiver side proxy (RSP) nodes (hosted on different Cloud data centers) worldwide, before being delivered to the eventual receiver devices.

1. INTRODUCTION

Video streaming is an increasingly popular Internet application. However, despite its popularity, real-time video streaming still remains a challenge in many scenarios. Limited home broadband bandwidth and mobile phone 3G bandwidth means many users stream videos at low quality and compromise on their user experience. To overcome this problem, we propose CStream, a system that aggregates bandwidth from multiple co-operating users in a neighborhood environment for better video streaming. CStream exploits the fact that wireless devices have multiple network interfaces and connects co-operating users with a wireless ad-hoc network to aggregate their unused downlink Internet bandwidth to improve video quality. CStream dynamically generates a streaming plan to stream a single video using multiple connections and continuously adapts to changes in the neighborhood and variations in the available bandwidth. The main objectives of the proposed system are,

- To built a system fot CStream and evaluated it on a controlled test-bed of computers with various performance measures.
- To show linear increase in throughput and improved video streaming quality as the

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

number of cooperating users in a neighborhood increase.

- CSTREAM does not make any changes to end devices and still results in significantly lower startup times, low buffering ratios and high average bit rates all key QoE parameters that determine user stickiness.

2 RELATED WORK

The cache node will check for the content (static web page fragments, images, JavaScript files, etc) being requested in its local disk (or local area network) and if found, it will serve it locally. If not found, it will pull a copy of the content from the origin server. This is the typical "pull mode" of operation of a CDN (similar to the use of hierarchical memory caches on traditional computing systems). It is also possible to push content proactively (referred to as "push mode") to all or selected cache nodes. A CDN can reduce the rendering time of most web pages drastically as it serves most content from a location that is much closer the client than the origin server. B. Live and Video On Demand (VoD) on the Internet Video On Demand (VoD) refers to video that is hosted on one or more servers in the cloud and is streamed to a viewer when it explicitly requests for that video. Live Internet video, on the other hand, refers to scenarios wherein a live event is broadcast to one or more viewers across the world over the Internet. This is becoming increasingly popular for sports events [5], [6] and for user generated content shared over social media [7]. Live video is typically captured by a camera (which could be on a mobile phone), converted into one or more desired formats at predetermined bit rates, and then either sent directly to the viewers or first transferred to one or more origin servers on the cloud. Once a live video feed is hosted on an origin server, it is streamed almost like Video on Demand. Typically live Internet video has a streaming delay of around 30 seconds due to the time taken in sending the feed first to a cloud server, transcoding and then streaming it. C. ABR Streaming and HTTP Progressive Downloads RTSP (Real Time Streaming Protocol) and HTTP are the two most popular protocols for streaming live video and VoD. HTTP is widely supported across all devices and firewalls and hence is usually the most preferred way of streaming video (e.g. HTTP Live Streaming (HLS)). Both live and VoD use ABR (Adaptive Bit Rate) streaming (or HTTP progressive

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

downloads in case of HTTP). A video file is first broken into small chunks or segments through a process called ABR chunking. Each chunk is of a small duration (typically less than 10 seconds), and is encoded at different bit rates. The client video player requests the initial chunk based on the observed bandwidth to the server (there are various ways to detect the available bandwidth between the client and the video server). As the video plays out, if the channel conditions between the client and the video server improve, client would request the next chunk at a higher bit rate and if they worsen, the client would request the next chunk at a lower bit rate. This adaption of bit rates help in providing seamless quality-of-experience and graceful degradation in presence of jittery network conditions. Breaking a video file into small chunks in case of HTTP progressive downloads, also makes the streaming process CDN friendly since these small chunks can be cached and served by traditional CDNs just like any other web content. Both VoD and live video can make of ABR streaming and HTTP progressive downloads. However, in case of live videos, traditional CDNs can distribute the live video feed only after the feed is available on an origin server on the cloud. This implies, that if there is a live event in Australia, and the origin server is in the US, even to serve a local client in Australia the live feed has to be first transferred to the origin server in the US (in practice, it is very likely that there will be more than 1 origin servers).

2.1 EXISTING SYSTEM

Traditional CDNs face many challenges in live and ondemand video delivery: CDNs are already responsible for a significant fraction of video quality problems for both live and VoD (including 20% of join failures and 22% of bitrate degradation). Furthermore, live video is shifting to viral user-created streams. Traditional CDNs were meant for serving static files and web page fragments which were typically pulled by the web site visitors all over the world. However, a live video stream has to travel from a live event location to its viewers all around the world in real time. CDNs need to maximize video-specific quality metrics (e.g., high bitrate and low join time) for both popular and unpopular streams while simultaneously coping with unexpected shifts in popularity/network conditions

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

Another issue is the use of pre-determined encoding options. Pre-determined encoding options can lead to inefficient use of bandwidth or reduced user experience. Fine grained bit rate adaptation can alleviate several buffering and join problems.

2.1.1 Disadvantages of Existing System

CDNs for live video require extensive CPU and storage resources that are typically associated with cloud providers. These resources are required both at the ingress and egress points in the cloud network to host various video processing and handling tasks Cloud demand elasticity enables the cloud provider to reuse these resources for other workloads when the live video workload reduces in volume.

The buffered unwatched video may be wasted if the user turns off the video player or switches to other videos.

Instead, ideally, when one bit is in error, the effect on the reconstructed video should be unperceivable, with minimal overhead. In addition, the perceived video quality should gracefully and proportionally degrade with decreasing channel quality.

2.2 PROPOSED SYSTEM

The system proposed the CSTREAM- a high performance cloud based live video delivery network. CSTREAM leverages the Cloud provider's global footprint, Cloud provider's high performance backbone network between different data centers,social media analytics and a UDP based fast data transfer protocol to optimize the quality of experience for end users, and the total cost incurred by the cloud provider in terms of network bandwidth and compute resources. CSTREAM allows a video broadcaster to be redirected to its closest publishing point (PP - hosted inside the "closest" Cloud data center), and then transfers the live stream at high speed using a UDP based fast protocol to one or more receiver side proxy (RSP) nodes (hosted on different Cloud data centers) worldwide, before being delivered to the eventual receiver devices. An multicast overlay is created to optimize the internal Cloud network bandwidth, while using a fast data transfer protocol like Aspera FASP between different nodes in the multicast overlay. Unlike Traditional Content Delivery Networks (CDNs) that are primarily receiver driven and comprise of passive cache nodes that act as passive servers that serve files, CSTREAM has active publishing points (PP) and receiver side proxy (RSP) nodes that are effective for both senders and receivers.

Advantage:

- ✓ Smooth and high quality video streaming.
- ✓ Avoid playback interruption and achieve better smoothness and quality.
- ✓ CSTREAM does not make any changes to end devices and still results in significantly lower startup times, low buffering ratios and high average bit rates all key QoE parameters that determine user stickiness.

3. SYSTEM DESCRIPTION

A content delivery network typically consists of a large number of cache nodes or "points of presence" (PoPs) distributed all over the world and interconnected by a medium to high bandwidth network. A website host (also referred to as the "origin host") will redirect an incoming client request to its nearest cache nodes based on its geographical location. The cache node will check for the content (static web page fragments, images, JavaScript files, etc) being requested in its local disk (or local area network) and if found, it will serve it locally. If not found, it will pull a copy of the content from the origin server. This is the typical "pull mode" of operation of a CDN (similar to the use of hierarchical memory caches on traditional computing systems). It is also possible to push content proactively (referred to as "push mode") to all or selected cache nodes. A CDN can reduce the rendering time of most web pages drastically as it serves most content from a location that is much closer the client than the origin server.

Video On Demand (VoD) refers to video that is hosted on one or more servers in the cloud and is streamed to a viewer when it explicitly requests for that video. Live Internet video, on the other hand, refers to scenarios wherein a live event is broadcast to one or more viewers across the world over the Internet. This is becoming increasingly popular for sports events and for user generated content shared over social media. Live video is typically captured by a camera (which could be on a mobile phone), converted into one or more desired formats at predetermined bit rates, and then either sent directly to the viewers or first transferred to one or more origin servers on the cloud. Once a live video feed is hosted on an origin server, it is streamed almost like Video on Demand. Typically live Internet video has a streaming delay of around 30 seconds due to the time taken in sending the feed first to a cloud server, transcoding and then streaming it.

RTSP (Real Time Streaming Protocol) and HTTP are the two most popular protocols for streaming live video and VoD. HTTP is widely supported across all devices and firewalls and hence is usually the most preferred way of streaming video (e.g. HTTP Live Streaming (HLS)). Both live and VoD use ABR (Adaptive Bit Rate) streaming (or HTTP progressive downloads in case of HTTP). A video file is first broken into small chunks or segments through a process called ABR chunking. Each chunk is of a small duration (typically less than 10 seconds), and is encoded at different bit rates. The client video player requests the initial chunk based on the observed bandwidth to the server (there are various ways to detect the available bandwidth between the client and the video server). As the video plays out, if the channel conditions between the client and the video server improve, client would request the next chunk at a higher bit rate and if they worsen, the client would request the next chunk at a lower bit rate. This adaption of bit rates help in providing seamless quality-of-experience and graceful degradation in presence of jittery network conditions. Breaking a video file into small chunks in case of HTTP progressive downloads, also makes the streaming process CDN friendly since these small chunks can be cached and served by traditional CDNs just like any other web content. Both VoD and live video can make of ABR streaming and HTTP progressive downloads. However, in case of live videos, traditional CDNs can distribute the live video feed only after the feed is available on an origin server on the cloud. This implies, that if there is a live event in Australia, and the origin server is in the US, even to serve a local client in Australia the live feed has to be first transferred to the origin server in the US (in practice, it is very likely that there will be more than 1 origin servers).

These considerations formed the basis of the design principles for CSTREAM.

Video Processing Nodes (VPNs) - PP and RSP

Video Processing Nodes (VPNs) primarily comprise of a video streaming engine and a custom data transfer stack. PP and RSP nodes are identical in terms of functional components, since all RSP also act as a publishing point for nearby RSPs who request a video feed from them. The system uses ffserver as the video streaming server and ffmpeg as the video upload/download engine. ffserver and ffmpeg have out-of-the-box support for a few video and audio encoders and decoders and more can be compiled and added separately. ffserver uses a configuration file to list all the video files it can stream. ffserver code was modified to ensure that new video feeds can be added dynamically to its configuration file without restarting it. The system uses byte streaming APIs to send requested video feeds from PP nodes to RSP nodes, and from RSP nodes to other RSP nodes.

Orchestrator Node(s)

Orchestrator Node has API handlers for download and upload requests. The upload request is a GET request. Upload request handler on the Orchestrator Node invokes the PP selector algorithm and sends the URL of the PP node in its response to the sender. The sender then sends the same GET request to the PP node, and a handler at the PP node first adds the video to be uploaded to the ffserver configuration file and to the Cloudant database so that it is available for RSPs immediately. The handler then sends the URI of the ffserver to the client in its response, who it turn issues a POST request to upload its video feed directly to its selected PP (running the ffserver). Similarly, the download request (issued by a player interestedin viewing the live video feed) is also a GET request. Download request handler figures out the client location using the geolite database and searches for the 5 closest RSP location ids and retrieves their IPs from the Cloudant database using the those ids.

RSP selector algorithm is invoked, and the client is redirected to the selected RSP. When the handler at the RSP receives a /download request, it checks if it already has the requested stream by querying the Cloudant database. If yes, then it redirects the receiver to the ffserver URI with that stream. If not, then it adds the feed and stream to the ffserver configuration file and then runs the ffmpeg command to get the stream from some other data center (which could be a PP or another close by RSP) that it gets by querying the Cloudant database for the server with that stream. The live video

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

feed thus received from another RSP or the PP, is simultaneously streamed using the ffserver on the RSP (after adding the feed to the ffserver configuration file).

InfraMonitor

CSTREAM InfraMonitor gathers two kinds of data — available bandwidth on the path from end clients (sender or receivers) to the potential PP or RSP nodes, and resource utilization information from the servers that are part of CSTREAM

Input: V: set of all servers that are part of CSTREAM, L: set of network links

```
1: function INFRAMONITOR(V, L)
2: /*Collect statistics*/
3: COLLECTBWSTATS(V)
4: COLLECTCPUUTIL(V)
5: COLLECTMEMUTIL(V)
6: /*Notify listeners of update()*/
7: NOTIFY()
8: end function
```

InfraMonitor measures available bandwidth from the "Q" potential PP or RSP host cloud data centers to the sender or receiver using a variant of "packet pair algorithms. This is because ping tests may not suffice as latency is not a true indicator of available bandwidth. Some variants of packet pair algorithms can converge in less than a second. InfraMonitor makes use of native cloud APIs to monitor the resource (CPU, memory and network) usage of the various physical servers in each data center (that have been pre-provisioned for CSTREAM).

Score Manager Service

InfraMonitor also comprises of a Score Manager Service. Based on the measurements gathered by the InfraMonitor, the Score Manager Service computes scores for all provisioned servers whenever it receives an update notification from the InfraMonitor. This score is indicative of the resources available at the server's disposal, which include (a) the available bandwidth on the path(s) to the server from a sender or a receiver, (b) % CPU utilization of the server, and (c) % memory utilization of the server. The lower the usage levels of a member's resources, the lower its score.

Input: V: set of all candidate servers

Output: _: set of scores

```
1: function SCOREMANAGER(V)
2: for all v 2 V do
3: Pv   GETPATHS(v)
4: for all P   Pv do
5: /*Compute scores*/
6: _   GETBWUTILIZATION(v)
7: _   GETCPUUTILIZATION(v)
```

```
8:   GETMEMUTILIZATION(v)
9: _   GETCOST(v)
10: _v;P   GETSCORE(_, _, , _)
11: _   _ [ _v;P
12: end for
13: end for
14: /*Notify listeners of update()*/
15: NOTIFY()
16: return _
17: end function
```

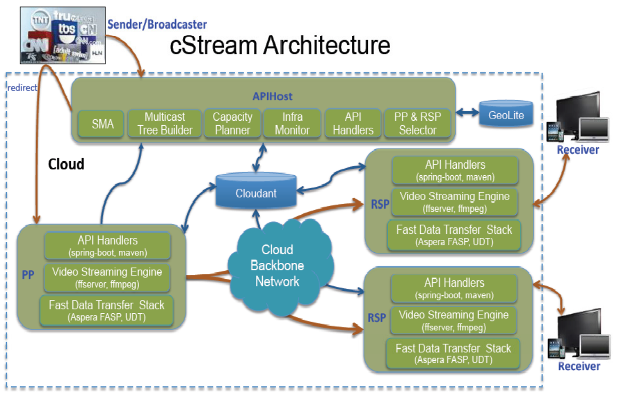
Sender/Broadcaster
cStream Architecture
redirect
APIHost
SMA
Multicast Tree Builder
Capacity Planner
Infra Monitor
API Handlers
PP & RSP Selector
GeoLite
Receiver
Cloud
Cloudant
API Handlers (spring-boot, maven)
Video Streaming Engine (ffserver, ffmpeg)
Fast Data Transfer Stack (Aspera FASP, UDT)
RSP
PP
Cloud Backbone Network
API Handlers (spring-boot, maven)
Video Streaming Engine (ffserver, ffmpeg)
Fast Data Transfer Stack (Aspera FASP, UDT)
RSP
API Handlers (spring-boot, maven)
Video Streaming Engine (ffserver, ffmpeg)
Fast Data Transfer Stack (Aspera FASP, UDT)
Receiver

Fig.1 cSTREAM Architecture

PP and RSP Selector

PP and RSP selector first gets a list of cloud data centers that are in close proximity to the sender or receiver. It uses the client IP addresses to query a local instance of the GeoLite database to get client's location, and then queries the CSTREAM meta data in Cloudant to find out the Q nearest cloud data centers. Q is a configuration parameter, and is typically set to less than 10 (5 being the default value). To find the most effective publishing point (PP) and receiver side proxy nodes (RSP), nearest geo-locations (in terms of network latency) may not suffice.

3.1 MODULE DESCRIPTION

PUBLISHING POINT (PP)

A PP node is an example of a Video Processing Node (VPN). PP nodes are Docker containers running video streaming engine and custom data transfer stack. They are hosted inside a cloud data center that exists in physical proximity to the sender of the live video stream. For each live video stream, CSTREAM will chose a PP node based on physical location, overall resource utilization (CPU, memory and network) and cost of a data center. PP nodes comprise of a video streaming engine

RECEIVER SIDE PROXY (RSP)

An RSP node is an example of a Video Processing Node (VPN). RSP nodes are Docker containers identical to PP nodes except that they are hosted inside cloud data centers that exist in close physical proximity to their respective receivers of a given live video stream. For each live video stream, CSTREAM will chose one or more RSP

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

nodes based on physical location, overall resource utilization (CPU, memory and network) and cost of a data center.

ORCHESTRATOR NODE (ON)

Orchestrator node is where all the business logic and intelligence of CSTREAM resides. ONs host the APIs to be used by senders and receivers. Senders use a REST API to upload a live video and the ON redirects the sender stream to the selected PP node. Similarly, receivers use a REST API to join a live video stream, and the ON redirects them to their respective selected RSP nodes. ON comprises of all the logic for effective PP and RSP node selection, infrastructure monitoring, capacity planning and auto scaling.

4. CONCLUSION

In this paper it has been presented the design and implementation of STREAM- a high performance cloud based live video delivery network that leverages the Cloud providers global footprint, Cloud providers high performance backbone network between different data centers, social media analytics and a UDP based fast data transfer protocol. CSTREAM can improve throughput and transfer times by up to 14 times for a transmission across the globe In this project, it has been formulated and studies a practical problem for large VoD streaming service providers: how to smartly utilize bandwidth resource to improve streaming QoE and peak load bandwidth requirements. It shows that these two goals are highly coupled, and if you can cut down the bandwidth waste you can use the saved bandwidth to improve QoE as well as save peak load bandwidth costs. The key is to understand user early departure behavior

4.1 FUTURE WORK

For future work, there are many interesting directions. On the analytical side, one can think it is possible to further refine the abstract model for analyzing smart streaming at an abstract level. This would extend the insights into the problem. On the experimental side, it may be extended that evaluation from single video to multiple videos, with different lengths and multiple resolutions. Finally, it may be considered to work towards deploying CSTREAM on a real large scale cloud.

REFERENCES

[1]. M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," IEEE Pervasive Computing, vol. 8, pp.14–23, 2009.

[2]. S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in Proc. of IEEE INFOCOM, 2012.

[3]. W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," IEEE Signal Processing Magazine, vol. 28, pp. 59–69, 2011.

[4]. T. Coppens, L. Trappeniners, and M. Godon, "AmigoTV: towards a social TV experience," in Proc. of EuroITV, 2004.

[5]. N. Ducheneaut, R. J. Moore, L. Oehlberg, J. D. Thornton, and E. Nickell, "Social TV: Designing for Distributed, Sociable Television Viewing," International Journal of Human-Computer Interaction, vol. 24, no. 2, pp. 136–154, 2008.

[6]. A. Carroll and G. Heiser, "An analysis of power consumption in as smartphone," in Proc. of USENIXATC, 2010.

[7] A. Balachandran, V. Sekar, A. Akella, and S. Seshan, "Analyzing the Potential Benefits of CDN Augmentation Strategies for Internet Video Workloads," in IMC 2013.