ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Secure Video Streaming Through Cloud and CDN: A Community Clustering Method

Subisha C.B[1], Hemand E.P[2]

M.Tech, Dept. of Computer Science, KMCT College of Engineering, Kozhikode, Kerala, India [1]

Assistant Professor, Dept. of Information Technology, KMCT College of Engineering, Kozhikode, Kerala, India [2]

ABSTRACT: This paper presents security extension of Video server platform for establishing secure media content delivery to authenticated end viewers. We develop a back end server platform provider technology for client, they want to store and deliver Videos to end viewers in secure manner. Our video streaming performed in video on demand (VOD) system. What is video stream? Video constantly download and received to an end viewer when requested for a video. In this paper, motivated by the social community classification, we present a social video replication and user request dispatching mechanism in the cloud CDN architecture to reduce the system operational cost, while guaranteeing the averaged service latency. Specifically, we first present a community classification method, which clusters social users with social relationships, close geo-locations, and similar video watching interests into various communities**.** Our technology platform provide server space for client to upload videos and distribute that videos in secure. Video server platform is based on Wowza server services with enhanced player with additional security features which support hls protocol so that it can deliver to variety of devices. Key features is implementation of modified Secure Token and encrypted delivery to devices for content cache we use a wowza module called media cache**.**

KEYWORDS: Cloud Storage, Open Stack Swift, CDN Network, Community Detection ,Wowza, Streaming Security.

I. INTRODUCTION

Streaming technologies are popular way how to deliver multimedia to almost any internet connected device. It is one of factor why network providers are increasing link speeds to end users, why mobile devices are growing in display sizes. It is good that this growth of multimedia delivery. In streaming as data is generated in real time and must be received on time, the development looks towards technologies which are less and less tolerant of data loss and which are capable of delivering data content in the shortest time possible.

The critical issue of video streaming is that of secure IP Flow access and content security. Access must be permitted to authorized personnel only. It is also necessary to confirm the real identity of a person and whether they Are authorized to access these information. These are discussed in this article.

We develop back end server platform provider Consists of cloud and CDN node. Content source placed in cloud for content replication we use open stack swift technology for high redundency.CDN node is the Edge server, technology used in that node is wowza. Wowza is media server software , It's for delivering video to variety of devices.

II. RELATED WORK

In this section, we investigate relevant literatures categorized into content delivery architecture, social media propagation, and social content replication.

A. Content Delivery Architecture

In general, there are three types of architectures proposed for video delivery, including CDN architecture, P2P architecture, and hybrid framework of CDN and P2P. In the CDN architecture [2], videos are dynamically replicated to CDN nodes at different geographical regions, and serve users all over the world. The P2P architecture [3] relies on the end-users (peer nodes) to store videos and assist video sharing among peers. The hybrid framework aims to integrate

DOI: 10.15680/IJIRCCE.2017. 0503330

advantages of both the CDN architecture and P2P architecture [4], where CDN nodes store video to guarantee stream quality and provide low startup latency, and peer nodes are used to provide scalability. Due to the huge volume and close-to-uniform popularity profile of UGCs, these architectures are not unsuitable. Recently, cloud CDN architecture sheds new lights into this field. Cloud CDN is built on the infrastructure of cloud storage providers, such as Amazon S3, Nirvanix, and Windows Azure. These cloud providers operate data centers that offer Internet scale content storage and delivery capabilities, and encapsulate these services in the form of virtual machines. Companies, who desire to build their own CDN infrastructure, can rent virtual machines from cloud storage providers and deploy services over such cloud architecture (referred as cloud CDN).In comparison with the traditional CDN, this novel architecture provides two benefits: 1) One can customize his CDN infrastructure without the high cost of owning or operating geographically dispersed data centers; 2) Small companies can rent cloud CDN service from cloud storage providers, and adopt the pay-as-demand way to save the bill. Chen *et al.* [5] discussed building the CDN networking over the cloud infrastructure. Wang *et al.* [1] and Wu *et al.* [6]explored demands and user access patterns in the traditional VOD application, and deployed VOD services onto a cloud platform. Jin *et al.* [7] proposed to deliver user-generated videos over a cloud-centric content delivery platform.

B. Social Media Propagation

Recently years have witnessed the growth of various online social networks. Based on traces of Flickr, Youtube, and Orkut, Mislove *et al.* [8] studied behaviors of Twitter users, and identified the geographic growth patterns of the social network. used the epidemic model to study the information propagation, where a piece of information is regarded as an infective disease that spreads via the social connections.

C. Social Content Replication

Some literatures utilized social features to distribute contents with better QoS. Considered the problem of placing social communities to different servers in a data center. Liu *et al.* [9] considered the visit/update rates of different types of user data, and decided which parts of data should be replicated. Proposed the propagation based replication strategy for social videos by jointly utilizing edge clouds and P2P clients. Wittie *et al.* [10] revealed the locality of interest of social communities, and proposed to build regional servers to cache data when they are first visited. Jiao *et al.*Our previous work presented the concept of utilizing social community for content placement and request dispatching. Many works utilize similar idea of community division to improve the content distribution over p2p networks. Our paper is distinct from those works in: 1) the context of our work is to utilize information from online social networks to cluster users, which needs to jointly consider several features; 2) our work does not enable video sharing among social users. In contrast, we utilize the community clustering result to map social users to CDN nodes, and conduct content replication and request dispatching to improve the utilization of replicas deployed on CDN nodes.

III. **PROPOSED SYSTEM**

In the proposed system , we first present the architecture of social video replication and request dispatching scheme.

System Architecture*:*

Our system architecture consists of two parts, including a Cloud CDN infrastructure for social video storage streaming, and a community-based request scheduling.

***1. Cloud CDN infrastructure*:** To serve social users from a variety of regions, our system rents resources in terms of virtual machines from multiple server service providers located at different regions for content caching and video streaming. All the resources rented from server service form a CDN node to serve the nearby users, i.e., each region correspond to a CDN node which provides video service with the lowest latency. All the CDN nodes are inter-connected over a WAN and constitute our cloud CDN infrastructure, in Fig. 1. Considering the video popularity dynamics and monetary cost, each CDN node can only store a small portion of videos from content source. When a user requests a video, he/she can download it either from a CDN node or from the remote content source.

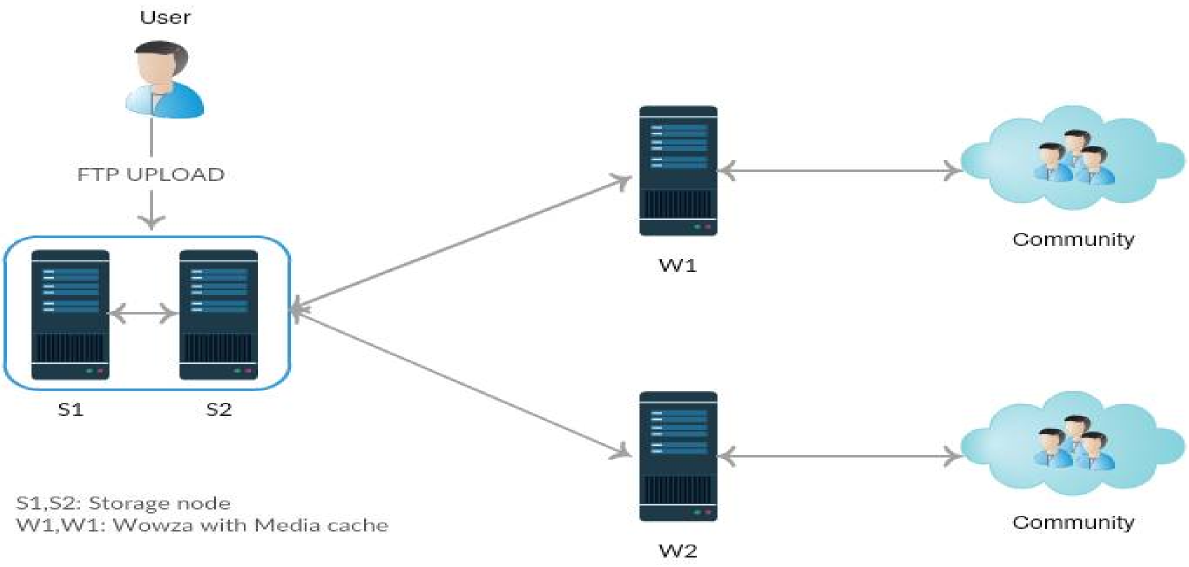

Fig. 1. Cloud CDN infrastructure for social video replication. Social users are grouped into various communities, and social users within a community can access videos from the corresponding CDN nodes.

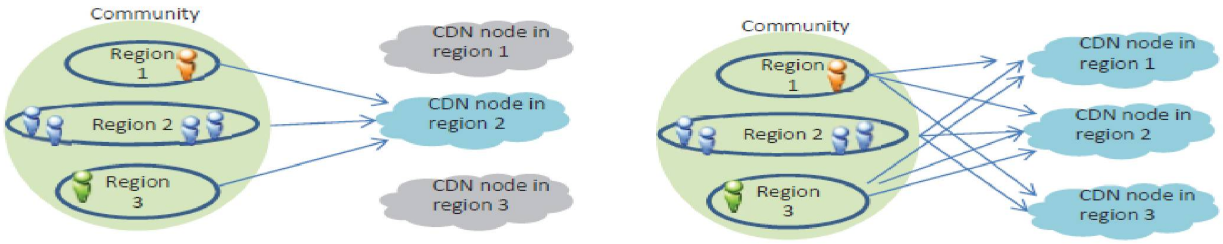

Fig.2. Community based request dispatching. (a) Requests from a community can be scheduled to a CDN node; (b) Requests from a community can be scheduled to a set of corresponding CDN nodes.

2. *Request dispatching:* Our system groups users into various communities and schedules requests based on community. However users within a community may come from several regions, each of which has its closest CDN node. The problem is to determine the corresponding set of CDN nodes for each community. There are two straight forward strategies as shown in Fig. 2. In the first strategy (Fig. 2(a)), we identify the region that contributes most users to a community, and the CDN node corresponding to this region is selected to serve this community. In the second strategy (Fig. 2(b)), users within a community can access a set of CDN nodes corresponding to regions covered by this community. A general video request dispatching procedure works as follows: When a social user initiates a video request, multiple queries to the corresponding CDN nodes will be sent out to search the available video replicas. Upon the response, the request will be served by the replica with the least time delay. Once there is no replica available, this request will be re-directed to the content source.

Technologies Used In Proposed System*:*

We develop back end server platform provider Using cloud and CDN node for secure video streaming. The technology provides server space for client to secure video uploading and Delivering. No: of technologies use in my proposed system. That technology implement in cloud server and Edge server. To secure video streaming, we place the content sources in cloud server that contains all the videos. Use CDN node as edge server that caches the videos. All the nodes are interconnected by CDN Network. CDN architecture is used for dynamic replication of videos from node to another node.

A. Cloud Server:

In the proposed system we use a cloud server for placing content source for content replication. A new technology, open stack swift, is introduced in our proposed System for high redundancy and automatic content replication. User uploads videos to content source via FTP.

i. Open Stack Swift

Open stack swift which is the object store service. Swift can be used to store data with high redundancy. The nodes in swift can be broadly classified in two categories:

1. Proxy Node: This is a public facing node. It handles the entire http request for various Swift operations like uploading, managing and modifying metadata. We can setup multiple proxy nodes and then load balance them using a standard load balancer.

2. Storage Node: This node actually stores data. It is recommended to make this node private, only accessible via proxy node but not directly. Other than storage service, this node also houses container service and account service which are used for managing mapping of containers and accounts respectively.

Open Stack Swift architecture

Open Stack Object Storage (Swift) provides redundant, scalable distributed object storage using clusters of standardized servers. “Distributed” means that each piece of the data is replicated across a cluster of storage nodes. The number of replicas is configurable, but should be set to at least three for production infrastructures .Objects in Swift are accessed via the REST interface, and can be stored, retrieved, and updated on demand. The object store can be easily scaled across a large number of servers.

Each object’s access path consists of exactly three elements:***/account/container/object***

The *object* is the exact data input by the user. *Accounts* and *containers* provide a way of grouping objects. Swift software is divided into components, which include *account servers*, *container servers*, and *object servers* that handle storage, replication, and management of objects, containers, and accounts. Apart from that, yet another machine called a *proxy server* exposes the swift API to users and streams objects to and from the client upon request. Account servers provide container listings for a given account. Container servers provide object listings in given containers. Object servers simply return or store the object itself, given its full path.

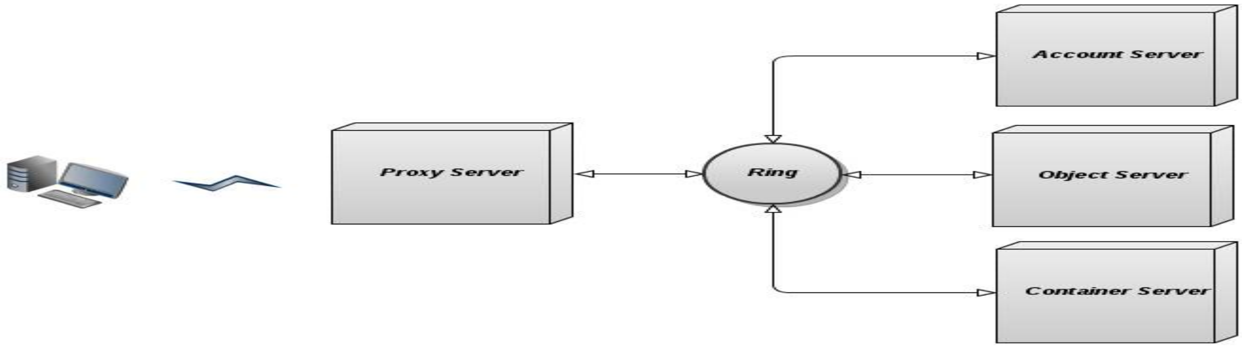

Fig3:Open Stack Swift Architecture

Rings

Since user data is distributed over a set of machines, it is essential to keep track of where they reside. Swift does this by maintaining internal data structures called “rings.” The rings are replicated on all the Swift cluster nodes (both storage and proxy). This way Swift avoids a flaw of many distributed file systems that rely on a centralized metadata server, as typically, that metadata store that becomes a chokepoint for calls to reference metadata. There are separate rings for account databases, container databases, and individual objects, but each ring works in the same way. In short for a given account, container, object name, the ring returns information on its physical location within storage nodes.

Proxy Server

The proxy server exposes public API and serves requests to storage entities. For each request, the proxy server looks up the location of the account, container, and object using the ring. Once it has their location, it routes the request accordingly. Objects are streamed between the proxy server and client directly and no buffering is enabled.

Object server
This is a simple BLOB storage server that can store, retrieve, and delete objects. Objects are stored as binary files on the storage nodes with metadata stored in the file's extended attributes (xattrs). This requires that the underlying file system choice for object servers support xattrs on files. Each object is stored using a path derived from the object name's hash and the operation's time stamp.

Container server
The container server handles listings of objects. It doesn't know where those objects are, just what objects are in a specific container. The listings are stored as sqlite3 database files, and replicated across the cluster in a way similar to objects. Statistics are also tracked that include the total number of objects, and total storage usage for that container. A special process swift container updater continuously sweeps the container databases on the node it works on, and updates the account database if a container's data changes. It uses the ring to locate the account to update.

Account server
The Account server handles container listings.

ii. FTP for cloud
To upload videos, we can provide FTP access to cloud storage using a proxy tool.

B: Edge Server:
CDN node used an edge server. software used in edge server is wowza media server with required module. Wowza is media server software. It supports almost all protocols. So it can deliver video to variety of devices.

i. Wowza
Wowza Streaming Engine (known as Wowza Media Server) is a unified streaming media server software developed by Wowza Media Systems. The server is used for streaming of live and on-demand video, audio, and rich Internet applications over IP networks to desktop, laptop, and tablet computers, mobile devices, IPTV set-top boxes, internet-connected TV sets,game consoles, and other network-connected devices. In the proposed system we use wowza. It supports almost all protocols. So it can deliver video to variety of devices. Supporting protocols RTMP, HLS, Rtsp, DASH, HDS, smooth. Operating system-Java-based, platform agnostic: Linux, Mac OS X, Solaris, UNIX, Windows. It is an adaptive bitrate streaming technology. High quality streaming media over the internet delivered from http web server works by breaking the content into a sequence of a small http based file segments, each segment containing a short interval play-back time that is potentially many hours in duration, such as movie or the live broadcast of a sports event. The content is made available at a variety of different bit rates, i.e., alternative segments encoded at different bit rates covering aligned short intervals of play back time are made available.

IV. RESULT ANALYSIS

In the proposed system , After implementing cloud the next step is to select best delivery servers available in today for Media delivery. After lot of studies, we choose to select wowza for media delivery.
Reason for wowza : (1)The Main purpose of this device is to transfer multimedia content to client devices. We choose HLS protocol for delivery purposes. This protocol was chosen for several reasons, mainly because the data is transmitted via HTTP, also it is possible to use encryption algorithm AES to encrypt video content. Streaming protocol HLS is supported by most of Mobile OS like Android, Blackberry. IOS as well as most of standard HTTP web players.
(2)The next challenge is to configure wowza for delivery. For doing this wowza need to pull the content from storage cloud using an wowza AddOn called media cache.
After discussing with one of wowza engineer we came to know that this module doesn't have a failover feature. So our biggest challenge is to create failover capability to media cache. Right now most of peoples are using intermediate software's like nginx,apache etc for failover.

Storage cloud >Nginx/Apache for Failover >Wowza Media cache >end viewer

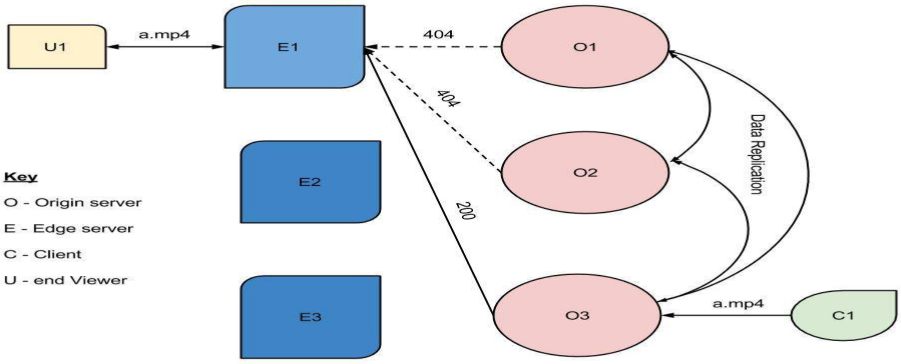

Fig 4:example for failover

So we have re configured media cache with failover over features. With this module we can avoid additional layers (Nginx) and will bring VOD delivery to high availability solutions.

Table1:Comparison test with mediacache failover and nginx failover results

Media cache failover	*Nginx Failover*	*Test result for 50 concurrent viewer*
5 to 10 sec	10 to 20 sec	*Delay for stream*
1 GB Ram	2 GB Ram	*Memory usage for 50 users*
2	3	*Load average for processor*
50	100	*Network connection*
2 (1935,80)	3 (1935,80,90)	*Number of Ports for servers*

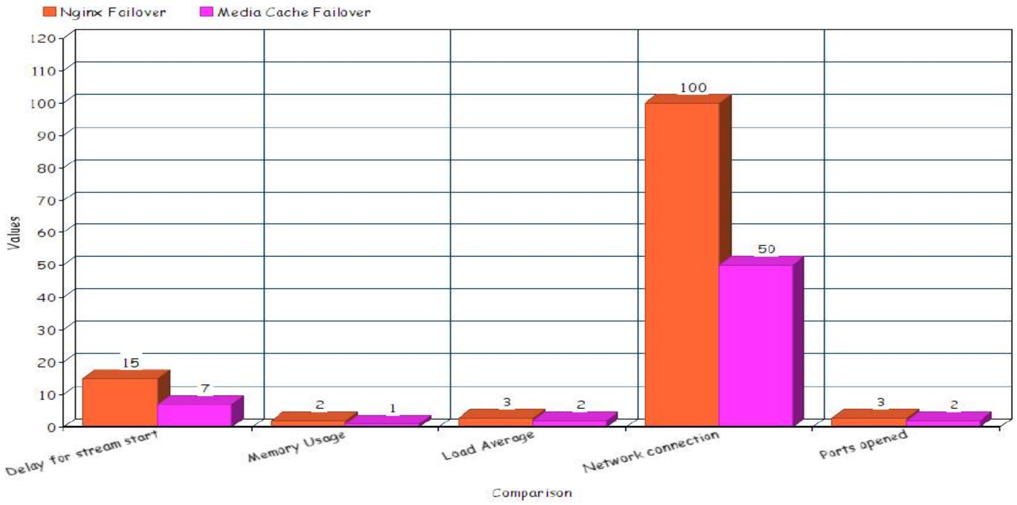

Fig 5:Comparison graph of Nginx failover and wowza cache failover

Apart from media cache AddOn, we have developed two more modules in wowza for stream security purpose. They are

1. ***Secure token***: This is a authentication based delivery system and protect the contents from Unauthorized viewers. *Apart from normal token security our token security has following features:*

a) *Option to exclude IP or IP block from secure token* .Which is very useful if we have another server and want to pull tokenized streams. If we exclude IP, then server will deliver the content if there is no token in requested URL.
b) *Token is based on Client IP basis*. So if the unauthorized user got a valid tokenized URL from an authorized viewer. then he/she can't use that valid tokenized URL. Because the token is based on Client IP. So unauthorized user can't use the valid tokenized URL from that IP.

*2. **Geo Block*** : Option to block and allow streams based on country. We have added whitelist IP and IP block in geo block also. Another option we added in this is to exclude domain from geo block. Which means, if client is requesting from a valid domain and country is blocked then we will deliver the content because of domain exclusion.

V. CONCLUSION

The paper investigated secure video streaming using CDN and cloud: a community Clustering method.That minimizing the averaged time delay. The video Distribution and community division method is based on the observation. we proposed our community-based Content replication and request routing architecture over the Cloud-centric CDN infrastructure. Wowza with wowza media cache used in edge Server, wowza for deliver video to variety of device and media cache for content Cache.open stack swift technology implement in cloud for High redundancy. Adaptive bitrates streaming, secure token Etc. technologies used in proposed system for secure and smooth streaming.

REFERENCES

1. F.Wang, J. Liu, and M. Chen, "Calms: Cloud-assisted live media streaming for globalized demands with time/region diversities," in *INFOCOM,2012 Proceedings IEEE*. IEEE, 2012, pp. 199–207.
2. G. Held, *A practical guide to content delivery networks*. CRC Press,2010.
3. A. Passarella, "A survey on content-centric technologies for the current internet: Cdn and p2p solutions," *Computer Communications*, vol. 35,no. 1, pp. 1–32, 2012.
4. K. Shuang and S. Su, "Paidd: a hybrid p2p-based architecture for improving data distribution in social networks," *Science China Information Sciences*, vol. 57, no. 4, pp. 1–11, 2014.
5. F. Chen, K. Guo, J. Lin, and T. La Porta, "Intra-cloud lightning: Building cdns in the cloud," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012,pp. 433–441.
6. Y. Wu, C. Wu, B. Li, X. Qiu, and F. C. Lau, "Cloudmedia: When cloud on demand meets video on demand," in *Distributed Computing Systems(ICDCS), 2011 31st International Conference on*. IEEE, 2011, pp. 268–277.
7. Y. Jin, Y. Wen, G. Shi, G. Wang, and A. Vasilakos, "Codaas: An experimental cloud-centric content delivery platform for user-generatedcontents," in *Computing, Networking and Communications (ICNC), 2012International Conference on*. IEEE, 2012, pp. 934–938.
8. B. Krishnamurthy, P. Gill, and M. Arlitt, "A few chirps about twitter,"in *Proceedings of the first workshop on Online social networks*. ACM,2008, pp. 19–24.
9. G. Liu, H. Shen, and H. Chandler, "Selective data replication for online social networks with distributed atacenters," in *Network Protocols(ICNP), 2013 21st IEEE International Conference on*, Oct 2013, pp. 1–10.
10. M. P. Wittie, V. Pejovic, L. Deek, K. C. Almeroth, and B. Y. Zhao,"Exploiting locality of interest in online social networks," in *Proceedingsof the 6th International Conference*. ACM, 2010, p. 25.
11. http://nginx.org/en/docs/http/ngx_http_secure_link_module.html
12. https://www.maxcdn.com/one/tutorial/pull-zone-security/
13. http://www.sparkmycloud.com/blog/whywowzastreamingengine/
14. https://www.wowza.com/docs/howtoscalevideoondemandstreamingwithmediacache

BIOGRAPHY

Subisha C B received her B.Tech. degree in Computer Science and engineering from Mahatma Gandhi University , and currently pursuing MTech inComputer Science at KMCT College of Engineering, Calicut. She has one year of IT industrial experience as a software Developer.

Hemand E P received his B.Tech. degree in Computer Science and engineering from Govt College of Engineering, Kannur. completed in 2008 and MTech degree in Computer Science from RV College of Engineering, Bangalore. completed in 2012.He has five years of experience in Teaching. He is currently working as Assistant professor at KMCT college of Engineering, Kozhikode.