HAVS: Hadoop Based Adaptive Video Streaming by the Integration of Cloudlets and Stratus

Keerthika Janani.M and Sudhakar.G
Anna University, Chennai

Abstract

Video Streaming is the remarkable problem which is encountered in the recent cloud environment. We derive the strategies for traffic over mobile network on a cloud assisted mobile platform. This paper presents HAVS [Hadoop based Adaptive Video Streaming]. The new and general purpose methodologies for streaming is described on the parameters [Stratus, Cloudlets, Map Reduce, Video Coding]. We propose a platform to enhance dynamic adaption and optimization of video. The reduction of energy consumption by leveraging cloud resources to make the data communication on Smartphone's more efficient.

Keywords: Hadoop, Cloudlet, Stratus, Map Reduce.

I. Introduction

Recently, most of mobile network operators are facing a serious challenge due to mobile data (especially video traffic) explosion [1]. While video streaming services become more crucial for mobile users, their traffic may often exceed the bandwidth capacity of cellular networks. Meanwhile the video streaming is not so challenging in wired networks, mobile net-works have been suffering from video traffic transmissions over scarce bandwidth of wireless links. Despite network operators' desperate efforts to enhance the wireless link bandwidth (e.g., 3G and LTE), soaring video traffic demands from mobile users are rapidly overwhelming the wireless link capacity.

While receiving video streaming traffic via 3G/4G mobile networks, mobile users often suffer from long buffering time and intermittent disruptions due to the limited bandwidth and link condition fluctuation caused by multi-path fading and user mobility. Thus, it is crucial to improve the service quality of mobile video streaming while using the networking and computing resources efficiently.

In order to support adaptive video streaming services in HAVS architecture, we discuss following issues:

• **Adaptability:** The video streaming service should be aware of available connectivity and bandwidth by taking into consideration dynamic wireless link conditions and be able to adapt to the best quality of video depending the estimated bandwidth.

• **Video Enhancement:** Due to mobility, it always does not allow the best quality of video by only cellular networks. Thus, by exploiting different interfaces, e.g., WiMax and WiFi, a MS (Mobile Station) receives video streams via a cellular link and also opportunistically accesses local WiFi with other MSs to get video segments with a higher quality.

• **NDN Caching and Sharing:** In-network caching capability gives an opportunity that a MS retrieves a video segment from any nearby MS already caching that video segment, not via a BS (Base Station). Furthermore, each MS can freely move and share video contents with each other.

Recently there have been many studies on how to improve the service quality of mobile video streaming on two aspects:

Scalability: To address this issue, the Scalable Video Coding (SVC) technique (Annex G extension) of the H.264 AVC video compression standard de-fines a base layer (BL) with multiple enhance layers (ELs). These sub streams can be encoded by exploiting three scalability features: (i) spatial scalability by layering image resolution (screen pixels), (ii) temporal scalability by layering the frame rate, and (iii) quality scalability by layering the image compression. By the SVC, a video can be de-coded/played at the lowest quality if only the BL is delivered. However, the more ELs can be delivered, the better quality of the video stream is achieved.

Adaptability: Traditional video streaming techniques designed by considering relatively stable traffic links between servers and users perform poorly in mobile environments. Thus the fluctuating wireless link status should be properly dealt with to provide 'tolerable" video streaming services. To address this issue, we have to adjust the video bit rate adapting to the currently time-varying available link bandwidth of each mobile user. Such adaptive streaming techniques can effectively reduce packet losses and bandwidth waste.

Scalable video coding and adaptive streaming techniques can be jointly combined to accomplish effectively the best possible quality of video streaming services. That is, we can dynamically adjust the number of SVC layers depending on the current link status.

Cloud computing techniques are poised to flexibly provide scalable resources to content/service providers, and process offloading to mobile users. Thus, cloud data centers can easily provision for large-scale real-time video services as investigated. Several studies on mobile cloud computing technologies have proposed to generate personalized intelligent agents for servicing mobile users, e.g., Cloudlet and Stratus. This is because, in the cloud, multiple agent instances (or

ISSN: 2278 – 1323

threads) can be maintained dynamically and efficiently depending on the time-varying user demands.

We use Hadoop, HDFS and HBase for storing and indexing our data, and associate this storage with a Web server that lets users navigate through the archive and retrieve documents. In the present post, we focus on *videos* and detail the solution adopted to serve true streaming from HDFS storage. There are basically two ways to play a video. The simplest one is a two-steps process: first the whole file is downloaded from the Web server to the user's computer, and then displayed by the player running the local copy. It has the disadvantage that the download step may take a while is the file is big (hundreds of megabytes are not uncommon). The second one uses (true) *streaming*: the video file is split into fragments which are sent from the Web server to the player, giving the illusion of a continuous stream. From the user point of view, it looks as if a window is swept over the video content, saving the need of a full initial download of the whole file.

Obviously, streaming is a more involved method because it requires a strong coordination between the components involved in the process, namely the player, the Web server, and the file system from which the video is retrieved. We examine this technical issue in the context of a Hadoop system where files are stored in HDFS, a file system dedicated to large distributed storage.

II. Related Work

I. Adaptive Video Streaming

In adaptive video streaming, e.g., Microsoft's Smooth Streaming, with each chunk download, the client measures the network bandwidth and runs a Rate Determination Algorithm (RDA) to determine which bit rate to request next. Each request represents an opportunity for the client to change bit rates. When selecting a bit rate, the RDA must consider the available bandwidth, CPU processing power, screen size, and the fullness of its buffer. The RDA must balance the desire to request high-quality video with the need to prevent its buffer from draining in order to deliver the highest sustainable quality without stops or stutters. Some of commercially available RDAs were evaluated in [8] and a rate adaptation algorithm for conversational 3G video streaming was proposed by. In addition, a couple of cross-layer adaptation techniques were discussed which can acquire more accurate information of the session quality so that the rate adaptation can be more accurately made.

II. Mobile Cloud Computing Technique

The cloud computing has been well positioned to provide video streaming services, especially in the wired Internet because of its scalability and capability [13]. For example, the quality-assured bandwidth auto-scaling for VoD streaming based on the cloud computing is proposed [14], and is a cloud-assisted live media streaming service for globally distributed users. However, extending the cloud computing-based services to mobile environments requires more factors to consider: wireless link dynamics, user mobility, and the limited capability of mobile devices [34], [35]. More recently, new designs for users on top of mobile cloud computing environments are proposed, which virtualizes private agents that are in charge of satisfying the requirements (e.g., QoS) of individual users such as Cloudlets [21] and Stratus [22]. Thus, we are motivated to design the HAVS-Cloud framework by using cloudlets a virtual agent in the cloud to provide adaptive video streaming services.

III. Hadoop video Streaming Technique

Hadoop is a fault-tolerant distributed system for data storage which is highly scalable. The scalability is the result of a Self-Healing High Bandwidth Clustered Storage, known by the acronym of HDFS (Hadoop Distributed File System) and a specific fault-tolerant Distributed Processing, known as Map Reduce. The Hadoop cluster or cloud is disruptive in data center. By default, Hadoop distributions are configured to run on single machine and the Yahoo virtual machine is a good way to get going. However, the power of Hadoop comes from its inherent distributed nature and deploying distributed computing on a single machine misses its very point. It acts as a big initiative to create the streaming environment on the basis of the Hadoop with the integration of the Hadoop distributed file system.

III. HAVS-CLOUD FRAMEWORK

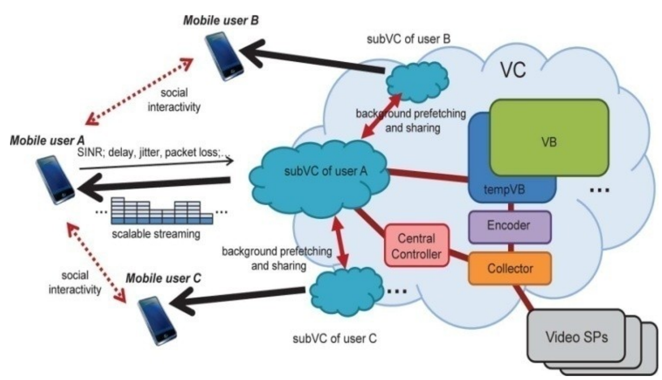


Fig 1. Illustration of the HAVS-Cloud framework with the Video Cloud (VC)

In this section we explain the HAVS-Cloud framework includes the Adaptive Mobile Video streaming and the Efficient Social Video sharing. As shown in Fig. 1, the whole video storing and streaming system in the cloud is called the Video Cloud (VC).In the VC, there is a large-scale video base (VB), which stores the most of the popular video clips for the video service providers (VSPs). A temporal video base (temp HVB) is used to cache new candidates for the popular videos, while temp HVB counts the access frequency of each video. The VC keeps running a collector to seek videos which are

ISSN: 2278 – 1323

already popular in VSPs, and will re-encode the collected videos into SVC format and store into temp HVB first. By this 2-tier storage, the HAVS-Cloud can keep serving most of popular videos eternally. Note that management work will be handled by the controller in the VC. Specialized for each mobile user, a sub-video cloud (Cloudlets) is created dynamically if there is any video streaming demand from the user. The sub-VC has a sub video base (sub VB), which stores the recently fetched video segments. Note that the video deliveries among the Cloudlets and the VC in most cases are actually not "copy", but just "link" operations on the same file eternally within the cloud data center [36]. There is also encoding function in Cloudlets (actually a smaller-scale encoder instance of the encoder in VC), and if the mobile user demands a new video, which is not in the sub VB or the VB in VC, the Cloudlets will fetch, encode and transfer the video. During video streaming, mobile users will always report link conditions to their corresponding Cloudlets, and then the Cloudlets offer adaptive video streams. Note that each mobile device also has a temporary caching storage, which is called local video base (local VB), and is used for buffering and prefetching. Note that as the cloud service may across different places, or even continents, so in the case of a video delivery and prefetching between different data centers, an transmission will be carried out, which can be then called "copy". And because of the optimal deployment of data centers, as well as the capable links among the data centers, the "copy" of a large video file takes tiny delay [36].

IV. HAVS: Hadoop Adaptive Video Streaming

A.SVC

In SVC, a combination of the three lowest scalability is called the Base Layer (BL) while the enhanced combinations are called Enhancement Layers (ELs). The BL is guaranteed to be delivered, while more ELs can be also obtained when the link can afford, a better video quality can be expected. By using SVC encoding techniques, the server doesn't need to concern the client side or the link quality. Even some packets are lost, the client still can decode the video and display. But this is still not bandwidth-efficient due to the unnecessary packet loss. So it is necessary to control the SVC-based video streaming at the server side with the rate adaptation method to efficiently utilize the bandwidth. An efficient adaptation framework using SVC and MPEG-21 Digital Item Adaptation (DIA) is integrated and it is shown that SVC can seamlessly be adapted using DIA. For protection of packet losses in an error prone environment an unequal erasure protection scheme for SVC is provided.

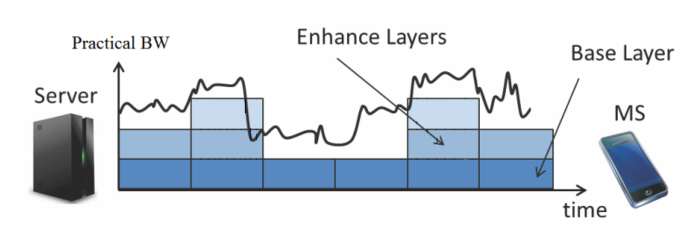
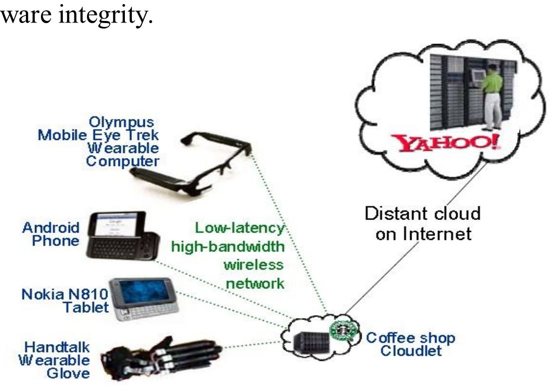


Fig 2. Scalable video streaming in the HAVS-Cloud Framework

B. How Cloudlets can Help

Rather than relying on a distant "cloud," the resource poverty of a mobile device can be addressed by using a nearby resource-rich cloudlet. The need for real-time interactive response can be met by low latency, one-hop, and high-bandwidth wireless access to the cloudlet. The mobile device functions as a thin client, with all significant computation occurring in the nearby cloudlet. Physical proximity of the cloudlet is essential: the end-to-end response time of applications executing in the cloudlet needs to be fast (few milliseconds) and predictable. If no cloudlet is available nearby, the mobile device can gracefully degrade to a fallback mode that involves a distant cloud or, in the worst case, solely its own resources. Full functionality and performance can return later, when a nearby cloudlet is discovered. As Figure 4(a) illustrates, cloudlets are decentralized and widely-dispersed Internet infrastructure whose compute cycles and storage resources can be leveraged by nearby mobile computers. A cloudlet can be viewed as a "data center in a box." It is self-managing, requiring little more than power, Internet connectivity, and access control for setup. This simplicity of management corresponds to an appliance model of computing resources, and makes it trivial to deploy on a business premises such as a coffee shop or a doctor's office. Internally, a cloudlet may be viewed as a cluster of multi-core computers, with gigabit internal connectivity and a high-bandwidth wireless LAN. For safe deployment in unmonitored areas, the cloudlet may be packaged in a tamper-resistant or tamper-evident enclosure with third-party remote monitoring of hardware integrity.



(a) Cloudlet Concept

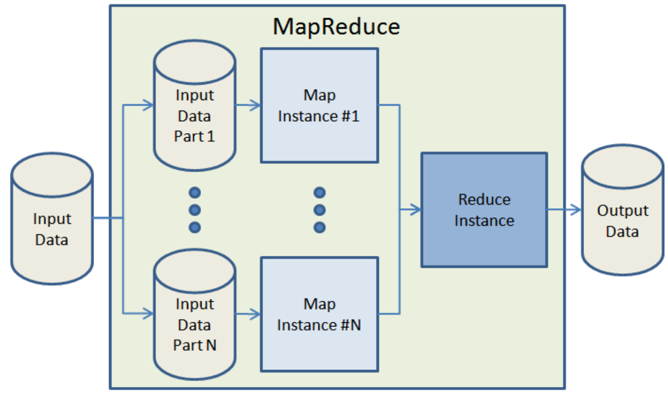
ISSN: 2278 – 1323

C. Energy Saving with Stratus

Stratus is a system to reduce this energy consumption by leveraging cloud resources to make data communication on Smartphone's more efficient. Using a cloud-based proxy, Stratus employs optimizations that adapt an application's incoming and outgoing traffic to better match the energy characteristics of the radio interface. The optimizations include (a) aggregation to bunch up sporadic transmissions, (b) asymmetric dictionary-based compression to reduce the number of bits transmitted over the air, and (c) opportunistic scheduling to avoid communication during periods of poor signal reception. These optimizations can be used individually, or in combination, subject to an application's delay tolerance. For example, using our Stratus prototype, the aggregation and compression optimizations together achieve up to 50% energy savings for web browsing, while the aggregation and scheduling optimizations together achieve up to 35% energy savings for a media streaming application.

D .Map Reduce Overview

The Map Reduce frame work [3] consists of a single master Job Tracker and one slave Task Tracker per cluster node. The master is responsible for scheduling the jobs' component tasks in the slaves, monitoring them, and re-executing any failed tasks. The slaves executed the tasks as directed by the master. As mentioned, Map Reduce applications are based on a master-slave model [6]. This part describes the various operations that are performed by a generic application to transform input data into output data according to that model. The user defined map and reduce functions [5]. The reduce function merges all intermediate values having the same intermediate. The Job Tracker will first determine the number of splits from the input path, and select some Task Tracker based on their network proximity to the data sources, then the Job Tracker send the task requests to those selected Task Trackers. Each Task Tracker will start the map phase processing by extracting the input data from the splits. For each record parsed by the "Input Format", it invokes the user provided "map" function, which emits a number of key/value pair in the memory buffer. A periodic wakeup process will sort the memory buffer into different reducer node by invoke the "combine" function. The key/value pairs are sorted into one of the R local files (suppose there are R reducer nodes). When the map task completes (all splits are done), the Task Tracker will notify the Job Tracker. When all the Task Trackers are done, the Job Tracker will notify the selected Task Trackers for the reduce phase. Each Task Tracker will read the region files remotely. It sorts the key/value pairs and for each key, it invokes the "reduce" function, which collects the key/aggregated Value into the output file (one per reducer node).



V. VIDEO SHARING IN HAVS

In the HAVS architecture, each node has a cache for storing and sharing. Thus in wireless HAVS, once an MS obtained video segments from the video server via the BS, other MSs nearby can opportunistically receive the video segment from the MS who holds the segments. In general, an MS always requests a segment over local WiFi connectivity, if any. If no MS that holds the segment is available, it will request the segment via 3G/4G. After the reception, the MS caches video segments at its own repository; from then on, any MS nearby can share the segments via the local WiFi connectivity. This sharing is easily facilitated in the HAVS framework. We substantiate the sharing concept in HAVS on the Android platform as follows. MSs Set up a CCNx overlay network on top of UDP/IP sockets. One MS, say the master, offers tethering service to the other MS using the WiFi Direct. The other MS can then obtain video segments directly with the same name from the cache of the master MS. MSs check WiFi link status, and if its link is poor, then they may switch to 3G/4G links. To achieve the 3G/4G offloading in wireless HAVS, every time an MS generates an interest, the CCNx will check whether there is any other MSs nearby. Using the Android SDK we can check whether there is already WiFi connectivity. Then we send the interest with the same segment name via the IP interface of the local connectivity.

VI. VIDEO STORAGE AND STREAMING FLOW BY CLOUDLETS AND STRATUS

The two parts, Cloudlets and Stratus, in HAVS-Cloud frame-work have tight connections and will together service the video streaming and sharing: they both rely on the cloud computing platform and are carried out by the private agencies of users; while prefetching in Stratus, the Cloudlets will still monitor and improve the transmission considering the link status; with a certain amount of prefetched segments by Stratus, Cloudlets can offer better video quality. Note that in order to exchange the videos among the local VBs, sub HVBs, temp VB and the VB, a video map (V Map) is used to indicate the required segments.

ISSN: 2278 – 1323

Once a mobile user starts to watch a video by a link, the local VB will first be checked whether there is any prefetched segment of the video so that it can directly start. If there is none or just some parts, the client will report a corresponding V Map to its Cloudlet. If the Cloudlet has prefetched parts in sub VB, the Cloudlet will initiate the segment transmission. But if there is also none in the sub HVB, the temp VB and VB in the center VC will be checked. For a non-existing video in HAVS-Cloud, the collector in VC will immediately fetch it from external video providers via the link; after re-encoding the video into SVC format, taking a bit longer delay, the Cloudlet will transfer to the mobile user. Also in HAVS-Cloud, if a video is shared among the Cloudlets at a certain frequency threshold (e.g., 10 times per day), it will be uploaded to the tempVB of the VC; and if it is further shared at a much higher frequency (e.g., 100 times per day), it will be stored with a longer lifetime in the VB. In such a manner, which is quite similar to the leveled CPU cache, the sub HVB and VB can always store fresh and popular videos in order to increase the probability of re-usage.

VII. IMPLEMENTATION AND EVALUATION

We evaluate the performance of the HAVS-Cloud framework by a prototype implementation. We choose the U-cloud server (premium) in the cloud computing service offered by Korean Telecom, and utilize the virtual server with 6 virtual CPU cores (2.66GHz) and 32GB memory, which is fast enough for encoding 480P (480 by 720) video with H.264 SVC format in 30 fps at real time [9]. In the cloud, we deploy our server application based on Java, including one main program handling all tasks of the whole VC, while the program dynamically initializes, maintains and terminates instances of another small Java application as private agents for all active users. We implement the mobile client at a mobile phone, Samsung Galaxy II, with android system version 4.0. The mobile data service is offered by LG U+ LTE network, while in some uncovered area the 3G network is used. Note that we still use “3G” to indicate the general cellular network. We test in the downtown area, so the practical bandwidth of the mobile link is not as high as we expected, but this won’t impact our experiment results. The test video is the “Gangnam style”, in H.264 format with 480P resolution downloaded from YouTube. Its size is 13.849 Mbytes and with a duration of 180 seconds. We first decode it by the x264 decoder into the YUV format, and re-encode it by the H.264 SVC encoder, the Joint Scalable Video Model (JSVM) software of version 9.1 [40]. We just use default settings for the decoding and encoding, and do the H.264 SVC encoding at the virtual server in the cloud. We split the video into segments by 1 second to 5 seconds that is to vary with values 1s, 2s, 3s, 4s and 5s. By JSVM, besides the base layer, we further make five temporal layers (1.875, 3.75, 7.5, 15, and 15 fps), two spatial layers (240 by 360 and 120 by 180) and two more quality layer (low and high), referring to [12] and [40]. Thus we define the best resolution configuration as “1+5+2+2”. And we also test different resolution configurations, including “1+1+1+1”, “1+2+2+2”, “1+3+2+2” and “1+4+2+2”.

Fig 2. Screenshot of HAVS application

VIII. CONCLUSION

In this paper, we discuss the impact of network on the performance of adaptive video streaming in wireless mobile environment, and design an adaptive mobile video streaming with offloading and sharing, HAVS, in the wireless architecture, along with the functionality for sharing among mobile users by Hybrid cloud. It is proved that HAVS outperforms pure streaming via Cloudlet in terms of the average video quality and traffic load reduction. Our current work still has some space for improvement: a) we haven’t took into account energy consumption yet. However, in some cases total energy consumption may be reduced due to high energy efficiency of WiFi. b) The benefit of WiFi sharing may be not so high in practical when various videos are requested for a short interval. In this case, we encourage learning from other domains to improve the sharing probability, e.g., social influence from social network services.

REFERENCES

[1] Project CCNx, “http://www.ccnx.org, ” Sep., 2009.
[2] D. Kulinski, J. Burke, “NDNVideo: Random-access Live and Prerecorded Streaming using NDN,” *Technical Report NDN-0007*, 2012.
[3] Z. Zhenkai, W. Sen Y. Xu, V. Jacobson and Z. Lixia, “ACT: Audio Conference Tool Over Named Data Networking,” *ICN*, 2011.
[4] S. Akhshabi, A. C. Begen, and C. Dovrolis, “An Experimental Evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP,” *MMSys*, 2011
[5] K. Tappayuthpijarn, G. Liebl, T. Stockhammer, and E. Steinbach, “Adaptive video streaming over a mobile network with TCP-Friendly Rate Control,” *ACM IWCMC*, 2009
[6] X. Wang, M. Chen, B. Pan, T. Kwon, L. T. Yang, V. C. M. Leung, “AMES-Cloud: A Framework of Adaptive Mobile Video Streamingand Efficient Social Video Sharing in the Clouds,” *IEEE Transaction of Multimedia*, 2013.
[7] T. Taleb and K. Hashimoto, “MS2: A novel multi-source mobile-streaming architecture,” *IEEE Trans. Broadcasting*, vol. 57, no.3, pp. 662–673, Sep. 2011.
[8] J. Fernandez, T. Taleb, M. Guizani, and N. Kato, “Bandwidth aggregation-aware dynamic QoS negotiation for real-time video

ISSN: 2278 – 1323

applications in next-generation wireless networks," *IEEE Trans. Multimedia*, vol. 11, no. 6, pp. 1082–1093, Oct. 2009.

[9] M. Wien, R. Cazoulat, A. Graffunder, A. Hutter, and P. Amon, "Real-time system for adaptive video streaming based on SVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1227–1237, Sep. 2007.

[10] P. McDonagh, C. Vallati, A. Pande, and P. Mohapatra, "Quality-oriented scalable video delivery using H. 264 SVC on an LTE network," in *Proc. WPMC*, 2011.

[11] Y. G. Wen, W. W. Zhang, K. Guan, D. Kilper, and H. Y. Luo, "Energy-optimal execution policy for a Cloud-assisted mobile application platform," Sep. 2011.

[12] J. Li, M. Qiu, Z. Ming, G. Quan, X. Qin, and Z. Gu, "Online optimization for scheduling preemptable tasks on IaaS cloud systems," *J. Parallel Distrib. Computing*, vol. 72, no. 5, pp. 666–677, 2012.

[13] N. Davies, "The case for VM-Based Cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[14] Z. Wang, L. Sun, C. Wu, and S. Yang, "Guiding Internet-scale video service deployment using microblog-based prediction," in *Proc. IEEE INFOCOM Mini-conf.*, 2012, pp. 2901–2905.

[15] A. Zambelli, "IIS smooth streaming technical overview," Microsoft Corp., 2009.

[16] V. Singh and I. D. D. Curcio, "Rate adaptation for conversational 3G video," in *Proc. IEEE INFOCOM Workshop*, 2009, pp. 205–211.

[17] F. Wang, J. Liu, and M. Chen, "CALMS: Cloud-assisted live media Streaming for globalized demands with time/region diversities," in *Proc. IEEE INFOCOM*, 2012, pp. 199–207.

[18] G.Wang and T. E.Ng, "The impact of virtualization on network performance of amazon EC2 data center," in *Proc. IEEE INFOCOM*, 2010, pp. 1163–1171.

[19] J. M. Kang, S. S. Seo, and J. W. Hong, "Personalized battery lifetime prediction for mobile devices based on usage patterns," *J. Computing Sci. Eng.*, vol. 5, no. 4, pp. 338–345, 2011.

Keerthika Janani.M is a M.E candidate in the Computer Science and Engineering, Ranganathan Engineering College, Anna University, Chennai. She received the B.Tech. degree in the Department of Information Technology of Ranganathan Engineering College, Anna University, Chennai in 2008. Her current research interests are in the areas of Cloud Computing and Cloud Security.

G.Sudhakar is a Ph.D Candidate in Software Metrics, Anna University Chennai, India. He received B.Tech Degree in the Dept of Information Technology at Mountzion College of Engineering in 2006 and M.E degree in the Dept of CSE at Anna University, Trichy in 2010. His Current research interests are in the area of software Engg, Network Security, Cloud Computing.

ISSN: 2278 – 1323