Cloud-Based Enhanced Mobile Video Streaming

Rajshekhar Targar[1], Ziaur Rahman[2]

[1, 2] Acharya Institute of Technology, Bangalore 107, India

Abstract: ***The recent cloud computing technology, with its rich resources to compensate for the limitations of mobile devices and connections, can potentially provide an ideal platform to support the desired mobile services. In this paper, Cloud-based Enhanced Mobile Video Streaming (CloudEMVS) is proposed. This system effectively utilizes both PaaS (Platform-as-a- Service) and IaaS (Infrastructure-as-a-Service) cloud services to offer the living-room experience of video watching to a group of disparate mobile users who can interact socially while sharing the video. A surrogate, performs efficient stream transcoding that matches the current connectivity quality of the mobile user is introduced.***

Keywords: Cloud Computing, IaaS, PaaS, Surrogate, Video Streaming.

1. Introduction

The rapidly increasing power of personal mobile devices (smartphones, tablets, etc.) is providing much richer contents and social interactions to users on the move. Tough challenges arise on how to effectively exploit cloud resources to facilitate mobile services, especially those with stringent interaction delay requirements. The design of a novel enhanced mobile video streaming, which can effectively utilize the cloud computing paradigm to offer a living- room experience of video watching to disparate mobile users with spontaneous social interactions. In CloudEMVS, mobile users can import a live or on-demand video (VoD) to watch from any video streaming site, invite their friends to watch the video concurrently, and chat with their friends while enjoying the video. CloudEMVS is designed to seamlessly utilize agile resource support and rich functionalities offered by both an IaaS (Infrastructure-as-a-Service) cloud and a PaaS (Platform-as-a-Service) cloud. The remainder of this paper is organized as follows. In Sec. II, Proposed system is compared with the existing system. In Sec. III architecture of CloudEMVS

2. Related Work

A number of mobile TV systems have sprung up in recent years, driven by both hardware and software advances in mobile devices. Some early systems [1] bring the "living-room" experience to small screens on the move. But they focus more on barrier clearance in order to realize the convergence of the television network and the mobile network, than exploring the demand of "social" interactions among mobile users. There is another trend in which efforts are dedicated to extending social elements to television systems [2]. In past, efforts are made [2] to add rich social interactions to TV but their design is limited to traditional broadcast program channels. Existing systems [3] are designed in a way; it has a mobile social TV system, which is customized for DVB-H networks and Symbian devices as opposed to a wider audience. Compared to these prior work and systems, a design for a generic, portable mobile social TV framework, featuring co viewing experiences among friends over geographical separations through mobile devices is targeted. This framework is open to all Internet-based video programs, either live or on-demand, and supports a wide range of devices with HTML5 compatible browsers installed, without any other mandatory component on the devices. For any application targeted at mobile devices, reducing power consumption is perennially one of the major concerns and challenges. Most of the existing systems exploit collaborations between the mobile OS and the mobile applications to balance the energy conservation and application performance. They investigate mobile multimedia streaming, similar to most of the other work, by adjusting the CPU power for energy saving; however, according to the recent measurement work, displays the wireless network card (including the cellular module) and not the CPU consume more than half of the overall power consumption in smart phones nowadays.

Cloud computing provides offloading mobile devices' computation workload to a nearby resource- rich infrastructure (i.e., Cloudlets) by dynamic VM synthesis. An elastic mobile application model by offloading part of the applications (weblets) to an IaaS cloud was introduced. Recently, attentions have been drawn to enabling media applications using the cloud, for both media storage [4] and processing.

Finally, there is a lack of a richly-featured cloud- based mobile social TV system in real life. The only system coming close to proposed system is Live Stream on the iOS platform. This iOS-locked application only supports live video channels, and all its social functions are bound to Facebook open APIs. Conversely, the prototype implemented is browser-based and platform independent; it supports both live channels, VoD channels and even personal channels hosted by any user, with wider usage ranges and flexible extensibility. The framework proposed can be readily applied to other cloud-assisted mobile media applications as well.

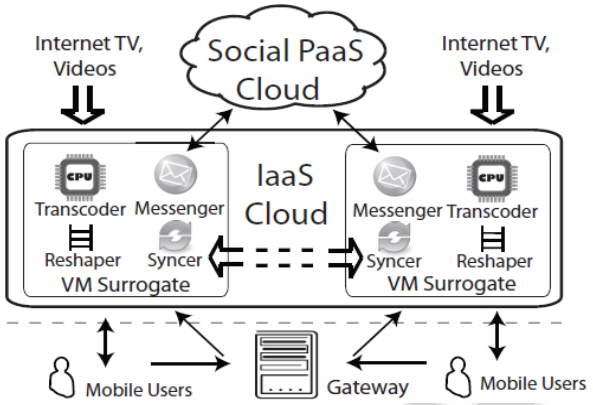

Figure 1: Architecture of Cloud EMVS

3. CloudEMVS: Architecture and Design

As a novel Mobile-Social TV system using cloud computing (CloudMEMVS), provides two major functionalities to participating mobile users: (1) Universal streaming: A user can stream a live or on- demand video from any video sources he chooses, such as a TV program provider or an Internet video streaming site, with tailored encoding formats and rates for the device each time. (2) Co-viewing with social exchanges: A user can invite multiple friends to watch the same video, and exchange text messages while watching. The group of friends watching the same video is referred to as a session. The mobile user who initiates a session is the host of the session. The architecture of CloudEMVS and the detailed designs of the different modules are presented in the following.

3.1 Key Modules

Fig. 1 gives an overview of the architecture of CloudEMVS. A surrogate (i.e., a virtual machine (VM) instance), or a VM surrogate equivalently, is created for each online mobile user in an IaaS cloud infrastructure. The surrogate acts as a proxy between the mobile device and the video sources, providing transcoding services as well as segmenting the streaming traffic for burst transmission to the user. Besides, they are also responsible for handling frequently exchanged social messages among their corresponding users in a timely and efficient manner, shielding mobile devices from unnecessary traffic and enabling battery efficient, spontaneous social interactions. The surrogates exchange social messages via a back-end PaaS cloud, which adds scalability and robustness to the system. There is a gateway server in CloudEMVS that keeps track of participating users and their VM surrogates, which can be implemented by a standalone server or VMs in the IaaS cloud. The design of CloudEMVS can be divided into the following major functional modules.

3.1.1 Transcoder

It resides in each surrogate, and is responsible for dynamically deciding how to encode the video stream from the video source in the appropriate format, dimension, and bit rate. Before delivery to the user, the video stream is further encapsulated into a proper transport stream. In this implementation, each video is exported as MPEG-2 transport streams, which is the de facto standard nowadays to deliver digital video and audio streams over lossy medium.

3.1.2 Reshaper

The reshaper in each surrogate receives the encoded transport stream from the transcoder, chops it into segments, and then sends each segment in a burst to the mobile device upon its request (i.e., a burst transmission mechanism), to achieve the best power efficiency of the device. The burst size, i.e., the amount of data in each burst, is carefully decided according to the 3G technologies implemented by the corresponding carrier.

3.1.3 Social Cloud

It is built on top of any general PaaS cloud services with BigTable-like data store to yield better economies of scale without being locked down to any specific proprietary platforms. Despite its implementation on Google App Engine (GAE) as a proof of concept, our prototype can be readily ported to other platforms. It stores all the social data in the system, including the online statuses of all users, records of the existing sessions, and messages (invitations and chat histories) in each session. The social data are categorized into different kinds and split into different entities (in analogy to tables and rows in traditional relational database, respectively). The social cloud is queried from time to time by the VM surrogates.

3.1.4 Messenger

It is the client side of the social cloud, residing in each surrogate in the IaaS cloud. The Messenger periodically queries the social cloud for the social data on behalf of the mobile user and pre-processes the data into a light-weighted format (plain text files), at a much lower frequency. The plain text files (in XML formats) are asynchronously delivered from the surrogate to the user in a traffic-friendly manner, i.e., little traffic is incurred. In the reverse direction, the messenger disseminates this user's messages (invitations and chat messages) to other users via the data store of the social cloud.

3.1.5 Syncer

The syncer on a surrogate guarantees that view-ing progress of this user is within a time window of other users in the same session (if the user chooses to synchronize with others). To achieve this, the syncer periodically retrieves the current playback progress of the session host and instructs its mobile user to adjust its playback position. In this way, friends can enjoy the "sitting together" viewing experience. Different from the design of communication among messagers, syncers on different VM surrogates communicate directly with each other as only limited amounts of traffic are involved.

3.1.6 Mobile Client

The mobile client is not required to install any specific client software in order to use CloudEMVS, as long as it has an HTML5 compatible browser (e.g., Mobile Safari, Chrome, etc.) and supports the HTTP Live Streaming protocol. Both are widely supported on most state-of-the-art smartphones.

3.1.7 Gateway

The gateway provides authentication services for users to log in to the CloudEMVS system, and stores users' credentials in a permanent table of a MySQL database it has installed. It also stores information of the pool of currently available VMs in the IaaS cloud in another in-memory table. After a user successfully logs in to the system, a VM surrogate will be assigned from the pool to the user. The in-memory table is used to guarantee small query latencies, since the VM pool is updated frequently as the gateway reserves and destroys VM instances according to the current workload. In addition, the gateway also stores each user's friend list in a plain text file (in XML formats), which is immediately uploaded to the surrogate after it is assigned to the user. The key designs in CloudEMVS are described as following.

3.2 Loosely Coupled Interfaces

Similar in spirit to web services, the interfaces between different modules in CloudEMVS, i.e., mobile users, VM surrogates, and the social cloud, are based on HTTP, a universal standard for all Internet- connected devices or platforms. Loose coupling between users and the infrastructure, almost any mobile device is ready to gain access to the CloudEMVS services, as long as it is installed with an HTTP browser. The VM surrogates provisioned in the IaaS cloud cooperate with the social cloud implemented on a PaaS cloud service via HTTP as well, with no knowledge of the inner components and underlying technologies of each other, which contributes significantly to the portability and easy maintenance of the system.

For social message exchanges among friends, CloudEMVS employs asynchronous communication. All the exchanged messages are routed via the surrogates to the social cloud, which efficiently organizes and stores the large volumes of data in a BigTable-like data store. The VM surrogates query the social cloud frequently and processes the retrieved data into XML files, for later retrieval by users in an asynchronous fashion. Such a design effectively separates the mobile users from the social cloud to significantly simplify the architecture, while the extra delay introduced at the VM surrogates is ignorable.

3.3 Pipelined Video Processing

Both live streaming of real-time contents and on demand streaming of stored contents are supported in CloudEMVS. Video processing in each surrogate is designed to work on the fly, i.e., the transcoder conducts real-time encoding from the video source, the encoded video is fed immediately into the reshaper for segmentation and transmission, and a mobile user can start viewing the video as soon as the first segment is received. To support dynamic bit rate switch, the transcoder launches multiple threads to transcode the video into multiple bit rates once the connection speed between the surrogate and the mobile user changes. The IaaS cloud where the surrogates are deployed represents an ideal platform for implementing such computation intensive jobs.

3.4 Burst Transmissions

First, 3G power states, different from Wi-Fi which is more similar to the LANed Internet access, 3G cellular services suffer from the limited radio resources, and therefore each user equipment (UE) needs to be regulated by a Radio Resource Control (RRC) state machine. Different 3G carriers may customize and deploy complex states in their respective cellular networks. Different states indicate different levels of allocated radio resources, and hence different levels of energy consumptions.

Second, Transmission mechanism, In CloudEMVS, maximum conservation of the battery capacity of the mobile device, and design a burst transmission mechanism for streaming between the surrogate and the device is aimed. Using the HTTP live streaming protocol, the mobile device sends out requests for the next segment of the video stream from time to time. The surrogate divides the video into segments, and sends each segment in a burst transmission to the mobile device, upon such a request. When the mobile device is receiving a segment, it operates in the high-power state when there is nothing to receive, it transfers to the low- power state via the intermediate state and remains there until the next burst (segment) arrives.

Third, to decide the burst size, i.e., the size of the segment transmitted in one burst, it is necessary to take into consideration characteristics of mobile streaming and energy consumption during state transitions.

3.5 VM Surrogates

All the VM surrogates are provisioned from Amazon EC2 web services and tracked by the gateway. To implement all the video processing related tasks using ANSI C, to guarantee the performance is proposed.

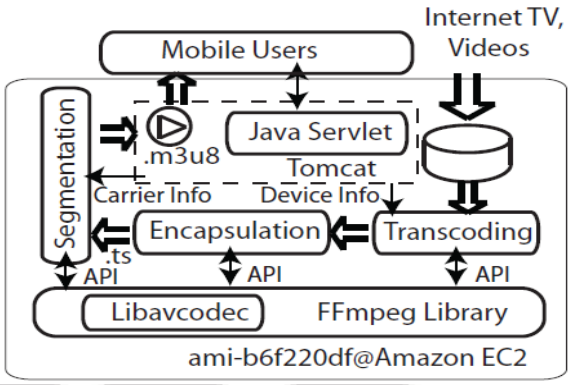

Figure 2: Streaming architecture in each customized VM image

In particular, FFmpeg together with libav-codec as the groundsill library to develop the transcoding, segmentation and reshaping modules on the VM surrogates is needed and also a Tomcat web server (version 6.5) to serve as a Servlet container and a file server on each surrogate is needed. Once a VM surrogate receives a video subscription request from the user, it downloads the video from the source URL, and processes the video stream by transcoding and segmentation, based

on the collected device configurations by the portal. Fig. 2 shows the streaming architecture in our customized VM image.

3.6 Data Models in the Social Cloud

GAE (Google Application Engine) is mainly used as the back-end data store to keep the transient states and data of CloudEMVS, including users' online presence status, social messages (invitation and chat messages) in all the sessions. With Jetty as the underlying Servlet container, most Java-based applications can be easily migrated to GAE, under limited usage constraints, where no platform-specific APIs are enforced for the deployment. GAE provides both its Java Persistence adapter and a set of proprietary low-level APIs to map the relational data.

Once a user logs in to the system and enters the URL of a video to watch, a session ID is generated for the new session (corresponding to vie wing of this video), by combining the user's "username" in the system with the time stamp when the session is created. The gateway delivers an HTTP request to a Servlet listener running on GAE, to notify it that an entry for the newly joined user should be added, with the user's "username" as the key and other information (URL of the subscribed video, the session ID, etc.) as the value.

Whenever a user decides to join a session hosted by his friend upon invitation, his VM surrogate switches to download the video of the session, and at the same time sends an HTTP request to the social cloud, for updating the session ID in this user's entry to the new one. If the user wishes to synchronize his playback progress with that of the session host, his VM surrogate synchronizes with the session host to maintain the playback "current time" value (HTML5 property).

The social cloud maintains a "Logs" entry for each existing session in CloudEMVS, with the session ID as the primary key and an array list as the value, which corresponds to individual messages in this session. When a user in a session posts a comment, this message is first sent to his VM surrogate, which further injects the message into the social cloud via another Servlet listener. The message is stored as a "Message" entry in the social cloud, with the message content as the value, and an auto-generated integer as the key. Entries "Logs" and "Message" are annotated by a @OneToMany relationship, to facilitate the data management. VM surrogates of users in the same session send periodical HTTP query requests to the social cloud for the latest comments from others. The default interval for retrieval of new comments is 10 seconds. The retrieved messages are stored and updated on the surrogates, which process them into well-formed XML formats for efficient parsing at the user devices. The user devices retrieve the XML files from the surrogates at a lower frequency (with default interval 1 minute), in order to minimize the power consumption and the traffic. Fig. 3 presents social message exchanges among a mobile user, his VM surrogate, and the GAE. A large number of entries in the social cloud become outdated very soon, since users may switch from one session to another, quit the system, and so on. We launch a cron job behind the scene every 10 minutes to clear those outdated entries is launched. For example, for sessions of which everybody has left, their "Logs" entries and all the associated "Message" entries are deleted in a single transaction.

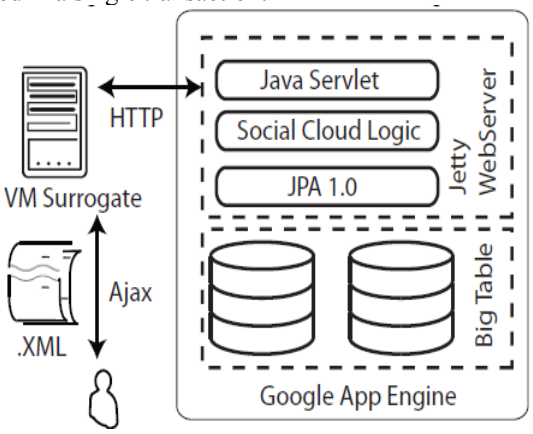

Figure 3: Social message exchanges via Google App Engine

4. Work flow of CloudEMVS

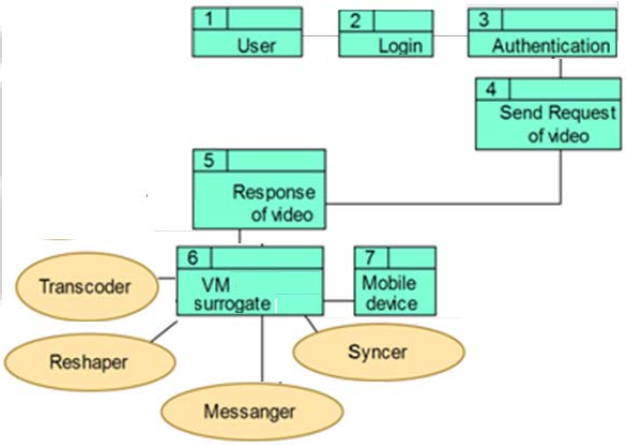

Figure 4: Flow Diagram of CloudEMVS

Figure 4 shows the flow diagram of CloudEMVS. The user enters the credentials and after authentication the user requests for the video either on demand or live. The VM surrogate processes the video and then sends it as the response to the user.

5. Sequence Diagram

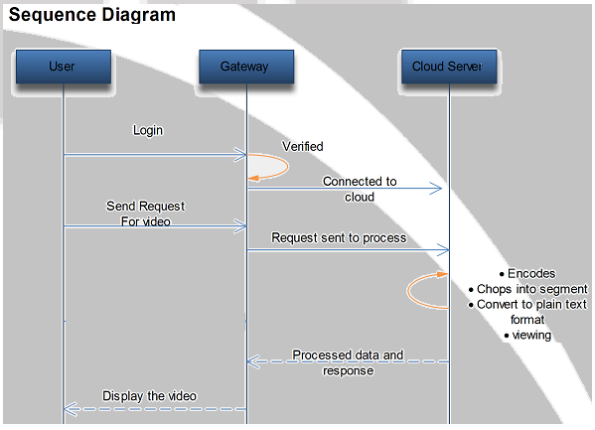

Figure 5: Sequence Diagram of CloudEMVS

Figure 5 shows the sequence diagram of CloudEMVS. The user logs in through the gateway. After which the user connects to the cloud where the user will be able to request the video either VOD or live. When the user sends the request for the video, the cloud encodes chops into segments, converts to plain format and finally sends the response to the user.

6. Conclusion and Future Work

In view of what might become a trend for mobile TV i.e., mobile social TV based on agile resources supports and rich functionalities of cloud computing services a generic and portal mobile social TV framework is introduced, CloudEMVS, that makes use of both an IaaS and a PaaS cloud. The framework provides efficient transcoding services for most platforms under various networks conditions and supports for co-viewing experiences through timely chat exchanges among the viewing users. By employing one surrogate VM for each mobile user, ultimate scalability of the system is achieved. Through an in-depth investigation of the power states in commercial 3G cellular networks, an energy-efficient burst transmission mechanism that can effectively increase the battery lifetime of user is also proposed.

Much more, however, can be done to enhance CloudEMVS to have product-level performance. In the current prototype, sharing of encoded streams (on the same format/bit rate) among surrogates of different users is not enabled. In future work, such sharing can be enabled and carried out in a peer-to-peer fashion, e.g., the surrogate of a newly joined user may fetch the transcoded streams directly from other surrogates, if they are encoded in the format/bit rate that the new user wants.

References

[1] J. Santos, D. Gomes, S. Sargento, R. L. Aguitar, N. Baker, M. Zatar, and A. Ikram, “Multicast/broadcast network convergence in next Generation mobile networks, “Computer Netw, vol 52, pp. 228-247, January 2008.
[2] T. Coppens, L. Trappeniners, and M. Godon, “AmigoTV: towards a social TV experience,” in Proc of EuroITV,2004.
[3] R. Schatz, S. Wagner, S. Egger, and N. Jordan, “Mobile TV becomes Social- Integrating Content with Communications,” in Proc. Of ITI, 2007.
[4] W. Zhu, C. Luo, J. Wang, and S . L i , “Multimedia cloud computing,” IEEE Signal Processing Magazine, vol 28, pp. 59-69, 2011.
[5] R. Pereira and K. Breitman, “A cloud b a s e d architecture for improving video compression time efficiency. The split & merge approach.
[6] Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Bur-rows, T. Chandra, A. Fikes, and R. E. Gruber, “Bigtable: A Distributed Storage System for Structured Data, in Proc of OSDI, 2006.

Author Profile

Rajshekhar Targar is currently pursuing his Master degree in Computer Science from Acharya Institute of Technology Bangalore, India, 2012-2014. He received B.E. degree in Computer Science from Gogte Institute of Technology in 2010. His research interest areas include cloud computing, mobile video streaming.