ICN based Scalable Audio-Video Conferencing on Virtualized Service Edge Router (VSER) Platform

Asit Chakraborti, Syed Obaid Amin, Bin Zhao, Aytac Azgin,
Ravishankar Ravindran and Guoqiang Wang
Huawei Research Center, Santa Clara, CA, USA.
{asit.chakraborti, obaid.amin, bin.zhao1, aytac.azgin, ravi.ravindran, gq.wang}@huawei.com

ABSTRACT

This paper proposes an audio/video conferencing architecture based on the VSER platform. The proposed solution follows a hybrid approach, where consumers are initialized to current producer's media state through a media-dependent notification framework provided by the VSER; whereas the data exchange leverages ICN features, resulting in bandwidth efficiency. The hybrid design provides better scalability and reliability when compared to the ICN based P2P conferencing systems.

1. INTRODUCTION

Real-time conferencing applications gain from the bandwidth efficiency offered by CCN/NDN architecture [1, 2]. However, challenges exist when applications with stringent real-time requirements are operated in a PULL based architecture. For example, the latency requirements for audio and video are $\leq 150ms$ and $\leq 350ms$, respectively [3]; whereas relative audio/video sync requirements are $+45ms$ to $-125ms$ [4]. Another challenge in a PULL based architecture is for the consumers to learn the names of the dynamic media particularly considering dynamic events like join/leave and transient connectivity issues in case of mobility or network degradation. To meet these requirements, this paper presents a design of an audio/video conferencing tool that uses VSER platform for notifications; whereas relies on the ICN framework for data exchange. This hybrid approach is useful in providing synchronization after a transient disconnection or a network failure and leverages ICN features, such as multicasting, for data exchange.

1.1 Virtual Service Edge Router (VSER) Platform

The audio/video conferencing solution is realized over an ICN based service platform, referred to here as the Virtual Service Edge Router (VSER), to achieve efficient service delivery leveraging ICN features. Considering the need for service-scaling, contextualization, agility and low latency the VSER platform is suitable for the network edge, as shown in Fig. 1. The benefits of such a platform is VSER's service hosting feature (enabling computing) along with ICN features such as unified naming, multicasting, mobility, and content-based security. Compared to NDN-RTC [5], which is a peer based conferencing system, the proposed VSER based conferencing system provides higher reliability (i.e., faster recovery from the transient network conditions) and performance with respect to media name sync among participants.

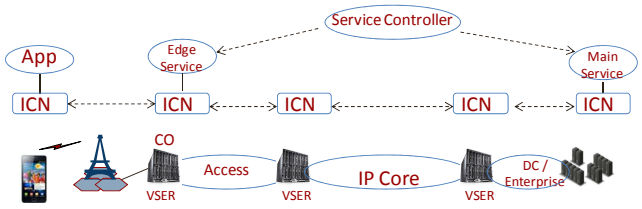

Figure 1: Deployment scenario.

2. CONFERENCING ARCHITECTURE

Video conferencing requires synchronization of media channels (audio/video/text) among multiple participants, which is handled by the service framework of VSER. The conferencing logic within VSER is responsible for connecting UE to the closest VSER, managing participant join/leave events and mobility, syncing media status, and conference state recovery. The conferencing architecture is summarized with respect to naming schema, name sync mechanism, consumer and producer design for a participant producing and consuming audio/video/text content.

2.1 Naming Convention

The proposed naming schema comprises of three components, (i)*network prefix*: which represents the routeable name component; (ii)*application identity*: which includes the conference and user ID; and (iii)*application metadata*: which includes the media-type and media specific attributes. More precisely, the naming convention is: */<**vser-id**>/<**conf-id**>/<**ue-id**>/<**media-type**>/<**session-id**>/<**frame-id**>/[<**chunk-id**>]*. Here the <*vser-id*> is a routeable name prefix of the VSER attached to the producer, whereas <*conf-id*>/<*ue-id*> identifies the participator context. The <*media-type*> can be audio, video or text. The <*session-id*> identifies a particular session and <*frame-id*> identifies a par-

ICN'15, September 30–October 2, 2015, San Francisco, CA, USA.
 ISBN 978-1-4503-3855-4/15/09
DOI: http://dx.doi.org/10.1145/2810156.2812615.

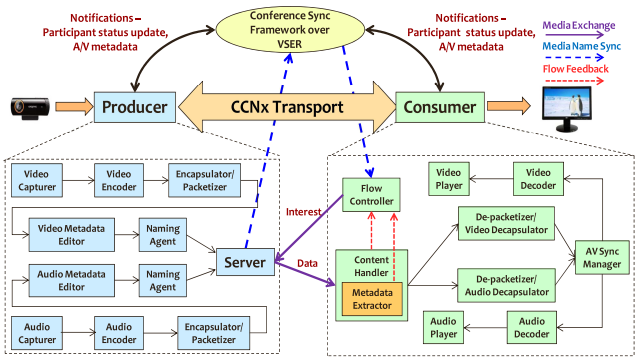

Figure 2: Audio-Video conferencing architecture.

ticular frame. The last component <*chunk-id*> is required if the media type is video, and identifies individual chunks in a video frame.

2.2 Notification Framework

As discussed in [6], the notification framework allows name sync and recovery from network failures. Once a producer selects a media type and generates a piece of content, it notifies the VSER's service agent of the new content, which is pushed to all the participants of a conference. The consumer then pulls that content by expressing Interest for it. The frequency of the notifications depends on the media type. For text, this notification is for every chat text committed by the participant. For real-time content, the notification traffic is minimized by transmitting the notifications at a configurable periodic interval. For video, the local agent sends a notification for every Group of Pictures (GOP), equivalent to $1/second$ in our setup. There is an inverse relationship between the volume of notification traffic and processing and communication overhead incurred by the consumer to achieve producer rate. For audio we set the notification frequency to $1/second$ as well.

2.3 Producer Design

Media generation at the producer end is driven by the participant behavior, i.e. the media is generated based on what the participant chooses to enable at a given point of time. Video producer uses the open-source Xuggler Java library to encode audio and video, using the MP3 and H.263 codec respectively. Audio is sampled at $44.1KHz$ and an audio frame roughly captures $14ms$ of audio. Video uses a frame rate of $25fps$ and also a GOP size of 25. The encoded frames are then adapted to CCN transport by chunking if required and named, and synced with the consumers using the notification framework. An application level PIT is also introduced that is used to store the Interests for near-future contents, sent by the clients in pre-fetching mode (discussed in Section 2.4). This allows producer to push out the data to the network as soon as it is generated, instead of waiting for the corresponding Interest.

2.4 Consumer Design

The conference service framework allows the local agent to discover and join active chatrooms and learn about the participants. This allows consumers to select a subset of audio/video feeds. The action to view a participant is conveyed to the conference service framework, which allows the consumer to learn producers' latest video/audio name states through the notification framework. This information is enough for the consumer to start expressing Interests and sync with the real-time content generated by a participant. To keep up with the producer's data generation rate, the consumer always sends out Interests in advance for future audio/video frames (pre-fetching). The frequency of these Interests primarily depends on the data generation rate of the producer. The state and data generation rate of the producer, can be derived by the consumer from the notifications.

In Fig. 2, the *Flow Controller* manages the Interest expression rate, based on feedback from notifications and *Content Handler*. Because of variable sized video frame data, the challenge is to learn the variable number of chunks comprising a video frame. This is addressed by using a frame descriptor metadata in each video chunk, which is extracted and fed back to the *Flow Controller* by the *Metadata Extractor*. The *Flow Controller* uses the periodic notifications from the conference notification framework to sync stream information. This allows us to handle issues such as new join events and network disruptions. During network disruptions, *Flow-Controller* starts from the latest conference state clearing the outstanding Interests for stale video frames. Based on the feedback from the *Content Handler*, the *Flow Controller* can also reissue Interests for the missing frames. The audio-video sync manager (*AVSyncManager*) module uses the timestamp information from the audio/video frames to sync audio/video playback before handing the content to the respective decoders.

3. DEMO SCENARIO

We demonstrate several features of the conferencing application: 1) VSER based conference service framework, by provisioning multiple conferences dynamically that participants can choose to join at will; 2) seamless interaction among participants in a conference session using audio/video/text media; 3) content sync as participants join/leave at random intervals, dynamic participant discovery feature, usefulness of the notifications for real-time content sync towards seamless experience even after a network disruption. The CCN router in VSER is a multi-threaded TLV implementation, while the service orchestration and SDN features are enabled using OpenStack and FloodLight.

4. REFERENCES

[1] Van Jacobson, et al. Networking named content. *Commun. ACM*, 55(1):117–124, January 2012.

[2] Lixia Zhang, et al. Named data networking. *SIGCOMM Comput. Commun. Rev.*, 44(3):66–73, July 2014.

[3] Yan Chen, Toni Farley, and Nong Ye. Qos requirements of network applications on the internet. *Inf. Knowl. Syst. Manag.*, 4(1):55–76, January 2004.

[4] ITU-R BT.1359-1 1, `https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.1359-1-199811-I!!PDF-E.pdf`.

[5] Peter Gusev. Ndn real time conferencing library. In *NDN Community Meeting*, Jun 2014.

[6] R. Ravindran, et al. Towards software defined icn based edge-cloud services. In *IEEE Cloud Networking (CloudNet)*, pages 227–235, Nov 2013.